# 及早准备，终有回报：类增量语义分割的新分类器预调方法

Zhengyuan Xie[1], Haiquan Lu[1], Jia-wen Xiao[1], Enguang Wang[1], Le Zhang[3], and Xialei Liu[1,2(✉)]

[1] VCIP, CS, Nankai University
[2] NKIARI, Shenzhen Futian
{xiezhengyuan}@mail.nankai.edu.cn
{xialei}@nankai.edu.cn
[3] SICE, UESTC

**摘要** 类增量语义分割力求在学习新任务的同时保留旧知识，但它受制于灾难性遗忘和背景漂移问题。先前的研究表明，初始化新分类器至关重要，这些研究主要关注从背景分类器中传递知识或为未来的新类别准备分类器，但他们忽视了新分类器的灵活性与差异性。在本文中，我们提出了一种在正式训练过程之前应用的新分类器预调方法（NeST），通过学习从旧分类器到新分类器的变换来进行初始化，而不是直接调整新分类器的参数。我们的方法可以使新分类器与骨干网络对齐并适应新数据，从而防止在学习新类别时特征提取器发生剧烈变化。此外，我们设计了一种策略，考虑了跨任务类别的相似性，以初始化变换中使用的矩阵，从而帮助实现稳定性和可塑性之间的平衡。在 Pascal VOC 2012 和 ADE20K 数据集上的实验表明，该策略可以显著提高以往方法的性能。本文的代码现已发布在https://github.com/zhengyuan-xie/ECCV24_NeST.

**关键词:** 类增量学习 · 语义分割

## 1 引言

如今，由深度神经网络驱动的分割模型在各个领域取得了显著成功 [9, 10, 33]。然而，这些模型通常在静态环境中进行训练和测试 [20]，这与真实世界的场景相悖 [47]。当面对连续的数据流 [24] 时，模型需要处理多个新类。将新数据与先前数据结合起来重新训练模型（即联合训练），可以确保模型的性能。然而，在处理大规模数据集时，计算成本变得极其高昂。简单地微调模型可能会导致先前获得的知识丧失，这种现象被称为灾难性遗忘 [29, 36]。

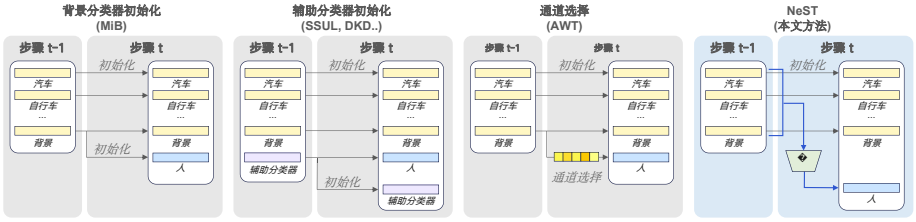



**图 1:** 用于类增量语义分割的不同分类器初始化方法。MiB [5] 直接使用背景分类器来初始化新分类器。一些方法 [2,6] 为未来的类别训练辅助分类器。AWT [21] 通过基于梯度的归因方法从背景分类器中选择最相关的权重用于新分类器的初始化。我们的新分类器预调方法通过学习从所有旧分类器的变换来生成新分类器进行初始化。

特别是对于语义分割，还面临着背景漂移问题 [5] （即当前步骤中标记为背景的像素可能属于先前或未来的类别）。

类增量语义分割（CISS）[5,6,16,55] 被提出以应对灾难性遗忘和背景漂移的挑战。它要求分割模型在学习新类别概念的同时保留旧知识。如果不加以限制，与旧类别相关的重要参数可能会被错误地更新 [28]，导致模型忘记先前学到的知识。相反，仅专注于记住旧知识可能会限制模型对新类别的学习。因此，持续学习的关键问题实际上是稳定性和可塑性之间的平衡 [22,28]。

在稳定性方面，现有的方法依赖于从前一步学习的旧模型进行知识迁移 [5,16,50,55]，例如*权重初始化*。然而，旧模型没有识别新类别的能力，因此找到合适的方法来初始化新添加的分类器至关重要。随机初始化可能会导致新分类器与特征之间的不匹配，从而导致训练不稳定 [5]。针对这种情况，如 图 1所示，MiB [5] 利用背景分类器来初始化新分类器，因为未来的类别可能会出现在当前的数据中，并被标记为背景。然而，这可能导致模型对真实背景像素做出错误的预测 [21]。AWT [21] 通过基于梯度的归因技术，从旧的背景分类器中选择最相关的权重进行初始化，但它忽略了其他旧分类器，并带来了巨大的内存成本。一些方法训练辅助分类器来初始化新分类器 [2,6]，但由于没有任何带有真实标签的未来数据，辅助分类器和真正的新分类器之间仍然存在偏差。此外，上述初始化方法对每个新分类器一视同仁，忽略了新类别之间的差异。

根据上述观察，我们提出了新分类器预调方法 (NeST)，以使新分类器更好地与骨干网络对齐并适应训练数据，并防止因训练过程不稳定而导致特征提取器发生剧烈变化。为了更好地利用旧知识，我们没有直接调整新分类器的参数，而是学习从所有旧分类器生成新分类器的线性变换。具体来说，

在当前步骤的正式训练过程之前，我们为每个新类别分配两个变换矩阵，即一个重要性矩阵和一个投影矩阵。作为可训练参数，重要性矩阵学习与新类别相关的旧分类器每个通道的加权得分，而投影矩阵则从加权的旧分类器中学习线性组合以生成新分类器。随后，我们使用当前步骤的数据来学习从旧分类器到每个新分类器的变换。最后，我们利用学习到的重要性矩阵和投影矩阵从旧分类器生成新分类器，然后利用得到的权重进行正式训练过程中的分类器初始化。通过使用这两个矩阵，新的分类器由旧知识生成，并且彼此不同。

为了实现稳定性与可塑性之间的平衡，我们还发现这些变换矩阵的初始值至关重要，并提出了一种初始化策略，考虑新旧类别之间的跨任务类别相似性，从而帮助新分类器的学习。

综上，本论文的主要贡献有以下三点：

1. 我们提出了一种新分类器预调方法（NeST），通过从所有旧分类器中学习变换，在正式训练过程之前生成带有旧知识的新分类器。
2. 我们进一步优化了变换矩阵的初始化，在稳定性和可塑性之间取得平衡。
3. 我们在 Pascal VOC 2012 和 ADE20K 数据集上进行了实验。结果表明，NeST 可以轻松地应用到其他方法中，并显著提升其性能。

## 2 相关工作

**语义分割** 近年来，深度学习技术的发展使得语义分割模型的性能得到了提升。全卷积网络 (FCN) [34] 开创了语义分割的先河，并且一系列基于卷积的分割网络在许多基准测试中取得了优异的表现 [8, 19, 23, 41, 48, 57]。最近，Transformer 架构变得越来越流行，并且性能效果十分显著 [11, 13, 32, 40, 44, 51, 59]。因此，我们在 ResNet 和 Swin-Transformer 这两个不同的骨干网络上进行了实验。

**类增量学习** 类增量学习致力于解决具有不断增加类别的任务中的灾难性遗忘问题。整个训练过程被分为多个步骤，在每个步骤中，模型需要学习一个或多个类别，这是最具挑战性的部分。现有的方法可以分为三类。*模型扩展方法* [26, 27, 52, 54] 在增量步骤中扩大模型规模，以学习新知识，同时在固定参数中保留旧知识。*基于重放的方法*存储一系列样本 [3, 45] 或沿着任务序列的原型 [61], 或使用生成网络来保持旧知识 [35, 43]。*参数正则化方法*则是限制某些参数的学习 [7, 30]，或是使用知识蒸馏技术 [1, 16, 17, 49] 来保留旧知识。

**类增量语义分割** CISS 要求分割模型在连续学习的步骤中识别所有已学习的类别。与分类任务不同，语义分割有其独特的背景漂移 [5] 问题。MiB [5] 首次提出了无偏交叉熵和蒸馏方法来解决背景漂移问题。PLOP [16] 使用伪标签和中间特征来传递旧知识。SDR [38] 提出了一种基于对比学习的方法，最小化类内特征距离。RCIL [55] 引入了具有互补网络结构的重参数化方法用于持续语义分割。GSC [14] 探索了考虑梯度和语义补偿的增量语义分割方法。EWF [50] 使用权重融合策略将新旧知识融合。许多方法已经在 CISS 中探索了分类器的初始化。SSUL [6] 利用辅助数据训练 "未知" 分类器，并使用它来初始化新分类器。DKD [2] 提出了一种分解的知识蒸馏方法，改进了模型的刚性和稳定性。AWT [21] 应用基于梯度的归因技术，将旧分类器的相关权重传递到新分类器。与上述方法不同，我们提出了一种新分类器预调方法，实现了可塑性与稳定性之间的平衡。

## 3 方法

### 3.1 背景知识

**问题定义** 根据 [5, 16]，CISS 包含若干步骤 $\{t\}_{t=1}^{n}$ 来接收顺序的数据流 $\{\mathcal{D}_t\}_{t=1}^{n}$ 及类别 $\{\mathcal{C}_t\}_{t=1}^{n}$。在每一步 $t$ 中，模型需要学习当前步骤中的 $|\mathcal{C}_t|$ 新类别，而旧的训练数据则不可用。对于当前步骤中的图像，属于 $\mathcal{C}_t$ 的像素会被标记为其真实类别，其余像素则被标记为背景。最后，在最后一步之后，模型将在所有已学习类别的数据上进行测试。

在步骤 $t$ 中，分割模型由特征提取器 $f_\theta^t$ 和分类器 $h_\phi^t$ 组成。新的分类器 $\phi_t$ 是新添加的，而前一步的分类器权重为 $\phi_{1:t-1}$。在本部分中，我们的主要关注点是分类器 $\phi_t$ 的初始化。

**现有方法** 在这一部分中，我们将阐述之前初始化方法的不足，并进一步明确我们的研究动机。如前所述，在之前的步骤中，属于 $\mathcal{C}_t$ 的类别可能会被标记为背景，这被称为*背景漂移*。基于这一前提，之前的方法要么试图找到更好的方式利用背景分类器 [5, 21]，要么为未来的类别训练辅助分类器 [2, 6]，但这两种方法都忽略了新类别之间的差异。此外，由于上述方法没有使用新类别数据进行训练，新分类器与特征提取器之间也会存在不匹配问题。当使用新数据进行训练时，这可能导致特征提取器的参数发生剧烈变化，从而损害先前的知识。

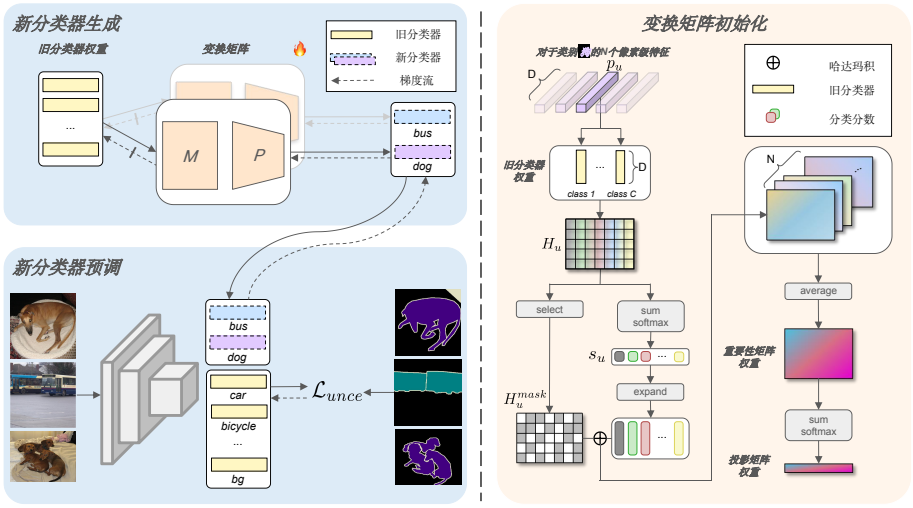



**图 2:** 我们的新分类器预调方法 (NeST) 的示意图。图的左侧展示了新分类器预调过程中的一个迭代步骤。图的右侧展示了在预调过程之前，基于跨任务类别相似性的初始化重要性矩阵和投影矩阵的过程。

### 3.2 新分类器预调

考虑到上述观察，我们提出了一种简单但有效的方法 NeST，以提升 CISS 的性能。在正式训练过程之前的预调阶段，我们首先通过旧分类器生成每个新分类器，然后在每次前向传递时使用新数据进行微调。

**新分类器生成** 如 图 2所示，对于每个新类 $c \in \mathcal{C}_t$，我们分配一个重要性矩阵和一个投影矩阵，通过学习从所有旧分类器到新分类器的变换，在每次前向传递中生成新分类器的权重，如下所示：

$$\mathbf{w}_c = (\mathbf{M}_c \odot \mathbf{W}_{old})\mathbf{P}_c\,, \tag{1}$$

其中，$\mathbf{w}_c \in \mathbb{R}^d$ 表示新类别 $c$ 的权重，$\mathbf{M}_c \in \mathbb{R}^{d\times n_{old}}$ 表示类别 $c$ 的重要性矩阵，$n_{old} = \left|\bigcup_{i=1}^{t-1}\mathcal{C}_i\right| + 1$ 表示包括背景在内的旧类别数，$\odot$ 表示 Hadamard 乘积操作，$\mathbf{W}_{old} \in \mathbb{R}^{d\times n_{old}}$ 表示所有旧分类器的权重矩阵，$\mathbf{P}_c \in \mathbb{R}^{n_{old}\times 1}$ 表示类别 $c$ 的投影矩阵，$d$ 表示 $f_\theta^{t-1}$ 的输出维度。$\mathbf{M}_c$ 反映了与新类别 $c$ 相关的每个旧分类器通道的重要性，而 $\mathbf{P}_c$ 则将旧分类器组合在一起，完成新类别的权重生成。在生成新权重后，我们将它们连接起来作为 $\phi_t$ 的参数，如下所示：

$$\phi_t = \mathrm{concat}[\mathbf{w}_{\mathrm{n_{old}}}, ..., \mathbf{w}_{\mathrm{n_{old}}+|\mathcal{C}_t|-1}]\,, \tag{2}$$

其中 concat[] 是连接操作。具体而言，在之前的步骤中新类别可能会随着旧背景类别被一起学习到，并且在预调过程中的背景分数可能较高，这阻碍了模型学习新类别。我们还在预调过程中从旧背景分类器学习到新的背景分类器的变换，如下所示：

$$\hat{\mathbf{w}}_0 = (\mathbf{M}_0 \odot \mathbf{w}_0)\mathbf{P}_0\,, \tag{3}$$

其中 $\mathbf{w}_0 \in \mathbb{R}^d$ 表示先前的背景分类器，$\mathbf{M}_0 \in \mathbb{R}^{d\times 1}$ 和 $\mathbf{P}_0 \in \mathbb{R}^{1\times 1}$ 是专门为背景分配的矩阵，$\hat{\mathbf{w}}_0 \in \mathbb{R}^d$ 是为当前任务准备的新背景分类器。

**预调与初始化** 在每次前向传递过程中，如 图 2所示，生成新分类器的权重后，我们将当前步骤的数据输入到模型中，使用生成的新分类器来计算损失，然后反向传播损失以更新可学习的参数。在预调过程中使用的损失函数是无偏的交叉熵 $\mathcal{L}_{unce}$ [5]，因为它有助于避免过拟合问题 [39]：

$$\mathcal{L}_{unce} = -\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \log \tilde{q}_x^t(i, y_i)\,, \tag{4}$$

其中 $\mathcal{I}$ 表示图像中的像素集，$y_i$ 表示当前步骤中像素 $i$ 的真实标签，$\tilde{q}_x^t$ 表示当前模型的更新输出。需要注意的是，在预调过程中，重要性矩阵和投影矩阵是可学习的，并且会被更新，而其他组件（如特征提取器）保持冻结状态。在预调过程结束后，我们使用重要性矩阵和投影矩阵生成每个新分类器的权重进行初始化，随后移除附加参数。

### 3.3 基于跨任务类别相似性的变换矩阵初始化

我们发现初始化重要性矩阵和投影矩阵是关键。随机初始化忽略了旧分类器中不同通道的重要性，导致性能不佳。AWT [21] 揭示了仅有少数背景分类器的通道具有对新分类的贡献，而冗余通道可能阻碍新分类器的学习。同时，对于不同的新类别，旧分类器的重要通道应该有所不同。因此，我们引入了基于跨任务类别相似性的变换矩阵初始化方法。核心思想是，如果旧类别与新类别越相似，那么在预调过程中，该旧类别的贡献应更为显著，因此在重要性矩阵和投影矩阵中应为对应的位置分配更大的初始权重。

因此，我们使用旧模型预测的结果作为每个新类别像素的跨任务类别相似性分数，目的是评估新旧类别之间的相似性。如图 2所示，在学习变换的过程前，我们将当前步骤中的每个训练图像输入到旧模型中，并获取层 $h_\phi^{t-1}$ 前的输出。然后考虑新类别像素嵌入 $\mathbf{p}_u \in \mathbb{R}^d$ 的分类过程，该嵌入从旧模型中提取，假设旧类别的总数量为 $n_{old}$，则矩阵乘法可以分解为 $\mathbf{p}_u$ 与旧类

别权重 $\mathbf{W}_{old} \in \mathbb{R}^{d\times n_{old}}$ 的逐元素乘积 （也称为 Hadamard 乘积）和一个 sum-softmax 操作，如下所示：

$$\begin{aligned}\mathbf{H}_u &= (\mathbf{W}_{old} \odot \mathbf{p}'_u),\\ \mathbf{s}_u &= \mathrm{softmax}(\mathrm{sum}(\mathbf{H}_u))^\top,\end{aligned} \tag{5}$$

其中 $\mathbf{p}'_u \in \mathbb{R}^{d\times n_{old}}$ 表示通过 $n_{old}$ 次传播扩展的像素嵌入 $\mathbf{p}_u$，$\mathbf{H}_u \in \mathbb{R}^{d\times n_{old}}$ 表示 Hadamard 乘积的结果，sum 表示沿通道维度求和 $\mathbf{H}_u$ 的结果，$\mathbf{s}_u \in \mathbb{R}^{n_{old}}$ 表示 $\mathbf{p}_u$ 的分类分数。我们考虑到，对于每一列 （对应旧类别 $i$）的 $\mathbf{H}_u$，如果 $\mathbf{p}_u$ 中的正值对应的是属于类别 $i$ 的正类，则将该位置设置为 1，并获得 $\mathbf{H}_u^{mask}$，它可以被视为一个二进制矩阵，如下所示：

$$\mathbf{H}_u^{mask}(i,j) = \begin{cases} 1, \mathbf{H}_u(i,j) > 0 \\ 0, otherwise\,. \end{cases} \tag{6}$$

最后，利用当前步骤的真实标签掩码，对于属于新类别 $\boldsymbol{c_{new}}$ 的每个像素嵌入，我们计算其对应的 $\mathbf{H}_{mask}$ 与预测分数的 Hadamard 乘积，然后对结果取平均值以获得新类别 $c_{new}$ 的重要性矩阵权重 $\mathbf{M}_{c_{new}}$，如下所示：

$$\mathbf{M}_{c_{new}} = \frac{1}{N}\sum_{\mathbf{p}_u} \mathbf{H}_u^{mask} \odot \mathbf{s}'_u\,, \tag{7}$$

其中 $\mathbf{p}_u$ 表示属于新类别 $c_{new}$ 的像素嵌入，$\mathbf{s}'_u \in \mathbb{R}^{d\times n_{old}}$ 表示通过将预测分数 $\mathbf{s}_u$ 以 $\mathbf{p}_u$ 以 $d$ 维扩展后的矩阵，$N$ 表示属于新类别 $c_{new}$ 的像素数目。这里我们使用 $\mathbf{s}'_u$ 是因为分数较小的旧类别不会对新类别的初始化贡献太大，因为这两个类别之间的相似性相对较低。对于投影矩阵权重 $\mathbf{P}_{c_{new}}$，我们沿通道维度求和 $\mathbf{M}_{c_{new}}$，并应用 softmax 函数以获得旧类别的权重得分，如下所示：

$$\mathbf{P}_{c_{new}} = (\mathrm{softmax}(\mathrm{sum}(\mathbf{M}_{c_{new}})))^\top. \tag{8}$$

然后我们使用 $\mathbf{P}_{c_{new}} \in \mathbb{R}^{n_{old}\times 1}$ 来初始化新类别 $c_{new}$ 的投影矩阵，因为这些得分反映了新旧类别之间的相似程度，决定了旧类别中应该重点关注什么部分以进行知识传递。我们在 算法 1中提供了 NeST 方法的伪代码。

## 4 实验

### 4.1 实验设置

**协议** 在 CISS 中，整个训练过程包含 $T$ 个步骤，每个步骤的任务可能包含一个或多个类别。在步骤 $t$ 中，属于先前步骤的像素会被标记为背景。在评

**算法 1** NeST 方法的伪代码

**输入:** 当前任务 $t$ 的训练样本 $\mathcal{D}_t = \{(x_i, y_i)\}_t$，特征提取器 $f_\theta^0$，分类器 $h_\phi^0$，任务数 $T$，新类集合 $\mathcal{C}_t$。

**输出:** 新分类器 $\phi_t$ 的初始权重。

**for** $t \in \{1, 2, ..., T\}$ **do**
  **for** $i \in \mathcal{C}_t$ **do** ▷ 初始化变换矩阵。节 3.3
    $(\mathbf{M}_i, \mathbf{P}_i) \leftarrow Init(\mathcal{D}_t, h_\phi^{t-1}, f_\theta^{t-1})$
  **end for**
  **while** 未收敛 **do** ▷ 预调新分类器 节 3.2
    通过最小化 $\mathcal{L}_{unce}$ 训练 $(\mathbf{M}, \mathbf{P})$
  **end while**
  $\phi_t \leftarrow \text{Transform}(\mathbf{M}, \mathbf{P}, \phi_{1:t-1})$ ▷ 初始化新分类器
  $f_\theta^t \leftarrow f_\theta^{t-1}$
  $h_\phi^t \leftarrow (h_\phi^{t-1}, \phi_t)$
  **while** 未收敛 **do** ▷ 正式训练过程
    训练 $f_\theta^t$ and $h_\phi^t$
  **end while**
**end for**

估中，模型需要识别所有已学习的类别。一共有两种设置：不相交和相交 [5]。不相交设置中，当前步骤中的图像不包含未来要学习的类别的任何像素。相交设置更为现实，因为未来的类别可能出现在当前步骤中的图像中，我们主要在相交设置下评估我们的方法。

**数据集** Pascal VOC 2012 数据集 [18] 包含 20 个类别，其中包括 10,582 张图像作为训练集和 1,449 张图像作为验证集。ADE20K 数据集 [60] 包含 150 个类别，其中包括 20,210 张图像作为训练集和 2,000 张图像作为验证集。在 CISS 中，训练设置可以描述为 $X - Y$，其中 $X$ 表示初始步骤中训练的类别数，$Y$ 表示增量步骤中训练的类别数。我们在 Pascal VOC 2012 数据集 [18] 上进行了 *10-1*、*15-1*、*15-5* 和 *19-1* 设置的实验。对于 ADE20K 数据集 [60]，我们使用 *100-50*、*100-10*、*100-5* 和 *50-50* 的设置来验证我们的方法。

**实现细节** 根据之前的方法 [5, 16, 55]，我们使用 [8] 搭配 ResNet-101 [25] 作为分割模型。我们还使用 Swin-B [32] 作为基于 Transformer 的骨干网络。按照 [5,16] 的方法，我们使用随机裁剪和水平翻转来进行数据增强。我们在 4 个 GPU 上以 24 的批次大小训练模型。实验中使用 SGD 作为优化器。在 Pascal VOC 2012 和 ADE20K 的实验中，我们都将初始步骤的学习率设置

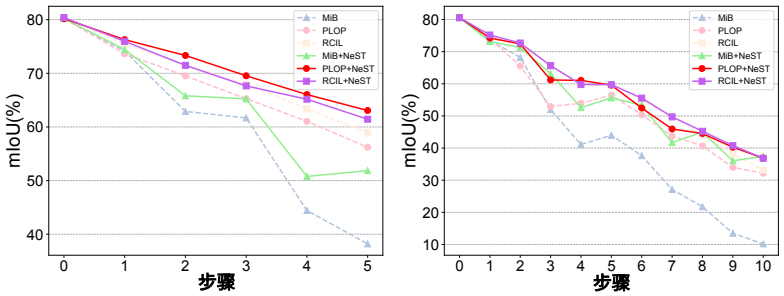
**表 1:** Pascal VOC 数据集上四种不同 CISS 场景的最后一步的 mIoU (%)。* 表示官方代码的复现结果。

| 方法 | 骨干网络 | 10-1(11 个步骤) | | | 15-1(6 个步骤) | | | 15-5(2 个步骤) | | | 19-1(2 个步骤) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0-10 | 11-20 | all | 0-15 | 16-20 | all | 0-15 | 16-20 | all | 0-19 | 20 | all |
| Joint | Res101 | 78.8 | 77.7 | 78.3 | 79.8 | 73.4 | 78.3 | 79.8 | 73.4 | 78.3 | 78.2 | 80.0 | 78.3 |
| ILT [37] | Res101 | 7.2 | 3.7 | 5.5 | 9.6 | 7.8 | 9.2 | 67.8 | 40.6 | 61.3 | 68.2 | 12.3 | 65.5 |
| SDR [38] | Res101 | 32.4 | 17.1 | 25.1 | 44.7 | 21.8 | 39.2 | 75.4 | 52.6 | 69.9 | 69.1 | 32.6 | 67.4 |
| PLOP+UCD [53] | Res101 | 42.3 | 28.3 | 35.3 | 66.3 | 21.6 | 55.1 | 75.0 | 51.8 | 69.2 | 75.9 | 39.5 | 74.0 |
| SPPA [31] | Res101 | - | - | - | 66.2 | 23.3 | 56.0 | 78.1 | 52.9 | 72.1 | 76.5 | 36.2 | 74.6 |
| MiB+AWT [21] | Res101 | 33.2 | 18.0 | 26.0 | 59.1 | 17.2 | 49.1 | 77.3 | 52.9 | 71.5 | - | - | - |
| ALIFE [39] | Res101 | - | - | - | 64.4 | 34.9 | 57.4 | 77.2 | 52.5 | 71.3 | 76.6 | 49.4 | 75.3 |
| GSC [14] | Res101 | 50.6 | 17.3 | 34.7 | 72.1 | 24.4 | 60.8 | 78.3 | 54.2 | 72.6 | 76.9 | 42.7 | 75.3 |
| MiB [5] | Res101 | 12.2 | 13.1 | 12.6 | 38.0 | 13.5 | 32.2 | 76.4 | 49.4 | 70.0 | 71.2 | 22.1 | 68.9 |
| MiB* [5] | Res101 | 10.4 | 9.9 | 10.2 | 45.2 | 15.7 | 38.2 | 76.8 | 49.1 | 70.2 | 71.6 | 28.6 | 69.6 |
| MiB+NeST (Ours) | Res101 | 52.3 | 21.0 | 37.4 | 61.7 | 20.4 | 51.8 | 77.1 | 50.1 | 70.7 | 71.7 | 28.2 | 69.7 |
| PLOP [16] | Res101 | 44.0 | 15.5 | 30.5 | 65.1 | 21.1 | 54.6 | 75.7 | 51.7 | 70.1 | 75.4 | 37.4 | 73.5 |
| PLOP* [16] | Res101 | 45.9 | 17.1 | 32.2 | 66.8 | 22.3 | 56.2 | 77.0 | 50.9 | 70.8 | 75.7 | 39.4 | 74.0 |
| PLOP+NeST (Ours) | Res101 | 54.2 | 17.8 | 36.9 | 72.2 | 33.7 | 63.1 | 77.6 | 55.8 | 72.4 | 77.0 | 49.1 | 75.7 |
| RCIL [55] | Res101 | 55.4 | 15.1 | 34.3 | 70.6 | 23.7 | 59.4 | 78.8 | 52.0 | 72.4 | 77.0 | 31.5 | 74.7 |
| RCIL* [55] | Res101 | 47.8 | 17.0 | 33.1 | 69.9 | 23.9 | 58.9 | 78.8 | 52.4 | 72.5 | 76.8 | 28.9 | 74.5 |
| RCIL+NeST (Ours) | Res101 | 51.4 | 20.9 | 36.8 | 71.9 | 28.0 | 61.4 | 79.0 | 52.8 | 72.8 | 77.0 | 33.3 | 74.9 |
| Joint | Swin-B | 80.4 | 79.7 | 80.1 | 81.1 | 76.7 | 80.1 | 81.1 | 76.7 | 80.1 | 80.0 | 80.7 | 80.1 |
| MiB* [5] | Swin-B | 11.4 | 18.9 | 15.0 | 35.0 | 43.2 | 36.9 | 80.7 | 66.5 | 77.3 | 79.2 | 60.2 | 78.3 |
| MiB+NeST (Ours) | Swin-B | 65.2 | 35.8 | 51.2 | 77.0 | 53.3 | 71.4 | 81.2 | 67.4 | 77.9 | 79.7 | 60.0 | 78.8 |
| PLOP* [16] | Swin-B | 37.8 | 23.1 | 30.8 | 74.1 | 52.1 | 68.9 | 80.1 | 68.1 | 77.2 | 77.0 | 65.8 | 76.4 |
| PLOP+NeST (Ours) | Swin-B | 64.3 | 28.3 | 47.2 | 76.8 | 57.2 | 72.2 | 80.5 | 70.8 | 78.2 | 79.6 | 70.2 | 79.1 |

为 0.02，增量步骤的学习率设置为 0.001。我们还使用 *poly* 调度作为权重衰减策略。对于 Pascal VOC 2012，我们在预调过程中训练了 5 个 epoch，学习率设置为 0.001。对于 ADE20K，在涉及 RCIL 和 Swin-B 的实验中，我们预调了 15 个 epoch，学习率设置为 0.1，其他实验则将学习率设置为 0.5。更多的细节已在补充材料中提供。

### 4.2 实验结果

在本节中，我们将 NeST 应用于三种经典方法 MiB [5], PLOP [16] 与 RCIL [55]，并结合不同的骨干网络。




**(a)** PASCAL VOC 2012 的 *15-1* 设置 **(b)** PASCAL VOC 2012 的 *10-1* 设置

**图 3:** *15-1* (a) 和 *10-1* (b) 设置下每个步骤的 mIoU (%)。

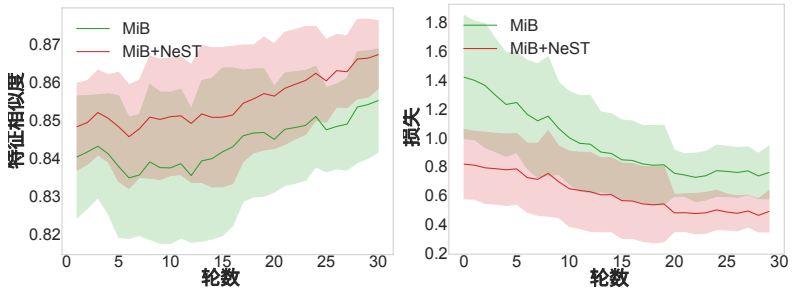
**Pascal VOC 2012** 我们在 *10-1*、*15-1*、*15-5* 和 *19-1* 设置下进行了实验。如 表 1所示，使用 NeST 的情况下，MiB [5]、PLOP [16] 和 RCIL [55] 的性能均有显著提升。在 *15-1* 设置下，将我们的方法结合 Deeplab，仅通过使用我们提出的新分类器策略，就能提升 MiB、PLOP 和 RCIL 的性能，分别为 13.6%、6.9% 和 2.5%。在更为困难的 *10-1* 场景中，NeST 可以大幅提升 MiB 的性能，达到 27.2% 的提升。同时，使用 Swin-B 的结果表明 NeST 也适用于基于 Transformer 的模型。在使用 Swin-B 的 MiB 上，NeST 可以在 *15-1* 设置下提升 34.5% 的性能，在 *10-1* 设置下提升 36.2% 的性能。在 表 1的结果中，展示了 NeST 在旧类别和新类别设置中显著增强了模型性能，这通过预调过程中将新分类器与现有骨干网络对齐来实现，从而减轻了旧知识遗忘的影响。我们还记录了我们的方法和基线方法在每个步骤的 mIoU。如 图 3所示，我们的方法在整个训练过程中表现均优于基线方法。

**ADE20K.** 为了进一步评估 NeST，我们在 ADE20K 数据集更具挑战性的设置下进行了实验。*100-50*、*100-10*、*100-5* 和 *50-50* 设置下的评估结果如 表 2所示。NeST 优于其他的竞争方法。在最为困难的 *100-5* 设置下，NeST 分别提升了 MiB [5]、PLOP [16] 和 RCIL [55] 的性能，分别为 6.8%、2.8% 和 1.3%，这表明 NeST 也适用于类别数较多的场景。

**有效性分析** 在正式训练步骤之前预调新分类器至关重要，这不仅有助于学习新类别，还能够弥补稳定性差距 [15]，这意味着模型在任务变化时将经历短暂的遗忘期，然后逐渐恢复。这一现象在 CISS 场景中也被发现 [2,50]。在每个正式训练步骤开始时，一个调整良好的新分类器可以使损失更小，因此对其他模型参数的影响也会减少，因为梯度值变得更小。由于模型的参数从旧模型继承而来，先前步骤中学到的旧知识可以得到保留。我们在 *15-1* 设

**表 2:** 在 ADE20K 数据集上，四种不同 CISS 场景的最后一步的 mIoU (%)。* 表示官方代码的复现结果。

| 方法 | 骨干网络 | 100-50(2 个步骤) | | | 100-10(6 个步骤) | | | 100-5(11 个步骤) | | | 50-50(3 个步骤) | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0-100 | 101-150 | all | 0-100 | 101-150 | all | 0-100 | 101-150 | all | 0-50 | 51-150 | all |
| Joint [55] | Res101 | 44.3 | 28.2 | 38.9 | 44.3 | 28.2 | 38.9 | 44.3 | 28.2 | 38.9 | 51.1 | 33.3 | 38.9 |
| ILT [37] | Res101 | 18.3 | 14.8 | 17.0 | 0.1 | 2.9 | 1.1 | 0.1 | 1.3 | 0.5 | 13.6 | 6.2 | 9.7 |
| PLOP+UCD [53] | Res101 | 42.1 | 15.8 | 33.3 | 40.8 | 15.2 | 32.3 | - | - | - | 47.1 | 24.1 | 31.8 |
| SPPA [31] | Res101 | 42.9 | 19.9 | 35.2 | 41.0 | 12.5 | 31.5 | - | - | - | 49.8 | 23.9 | 32.5 |
| MiB+AWT [21] | Res101 | 40.9 | 24.7 | 35.6 | 39.1 | 21.3 | 33.2 | 38.6 | 16.0 | 31.1 | 46.6 | 26.9 | 33.5 |
| ALIFE [39] | Res101 | 42.2 | 23.1 | 35.9 | 41.0 | 22.8 | 35.0 | - | - | - | 49.0 | 25.7 | 33.6 |
| GSC [14] | Res101 | 42.4 | 19.2 | 34.8 | 40.8 | 17.6 | 32.6 | 39.5 | 11.2 | 30.2 | 46.2 | 26.2 | 33.0 |
| MiB [5] | Res101 | 40.5 | 17.2 | 32.8 | 38.2 | 11.1 | 29.2 | 36.0 | 5.6 | 25.9 | 45.6 | 21.0 | 29.3 |
| MiB* [5] | Res101 | 40.5 | 23.5 | 34.9 | 37.8 | 12.1 | 29.3 | 35.8 | 6.0 | 25.9 | 45.9 | 23.9 | 31.3 |
| MiB+NeST (Ours) | Res101 | 40.3 | 24.6 | 35.1 | 40.2 | 20.6 | 33.7 | 39.9 | 18.0 | 32.7 | 45.6 | 26.8 | 33.2 |
| PLOP [16] | Res101 | 41.9 | 14.9 | 32.9 | 40.5 | 13.6 | 31.6 | 39.1 | 7.8 | 28.7 | 48.8 | 21.0 | 30.4 |
| PLOP* [16] | Res101 | 42.2 | 15.3 | 33.3 | 41.0 | 13.7 | 32.0 | 39.6 | 8.2 | 29.2 | 48.5 | 20.8 | 30.2 |
| PLOP+NeST (Ours) | Res101 | 42.2 | 24.3 | 36.3 | 40.9 | 22.0 | 34.7 | 39.3 | 17.4 | 32.0 | 48.7 | 27.7 | 34.8 |
| RCIL [55] | Res101 | 42.3 | 18.8 | 34.5 | 39.3 | 17.6 | 32.1 | 38.5 | 11.5 | 29.6 | 48.3 | 25.0 | 32.5 |
| RCIL* [55] | Res101 | 42.3 | 16.5 | 33.8 | 39.9 | 15.7 | 31.9 | 39.1 | 12.2 | 30.2 | 48.2 | 23.6 | 31.9 |
| RCIL+NeST (Ours) | Res101 | 42.3 | 22.8 | 35.8 | 40.7 | 19.0 | 33.5 | 39.4 | 15.5 | 31.5 | 48.2 | 27.4 | 34.4 |
| Joint | Swin-B | 43.4 | 31.9 | 39.6 | 43.4 | 31.9 | 39.6 | 43.4 | 31.9 | 39.6 | 50.7 | 33.9 | 39.6 |
| MiB* | Swin-B | 42.7 | 26.1 | 37.2 | 40.2 | 15.0 | 31.8 | 39.1 | 8.6 | 29.0 | 48.3 | 26.8 | 34.1 |
| MiB+NeST (Ours) | Swin-B | 42.8 | 27.8 | 37.9 | 41.8 | 23.8 | 35.9 | 40.5 | 19.9 | 33.7 | 49.7 | 29.3 | 36.2 |
| PLOP* | Swin-B | 43.4 | 17.1 | 34.7 | 41.4 | 17.7 | 33.6 | 39.7 | 13.6 | 31.0 | 50.5 | 24.1 | 33.0 |
| PLOP+NeST (Ours) | Swin-B | 43.5 | 26.5 | 37.9 | 41.7 | 24.2 | 35.9 | 39.7 | 18.3 | 32.6 | 50.6 | 28.9 | 36.2 |




**(a)** 特征图相似度 **(b)** 训练损失

**图 4:** Pascal VOC 2012 *15-1* 相交设置下的特征图相似度和训练损失。

置下对基线方法 MiB [5] 和 MiB+NeST 进行了实验。为了公平比较，我们绘制了第二步正式训练过程中每个 epoch 的平均损失和余弦特征相似度的

标准差，而在第一步中，所有实验均使用原始 MiB。如 图 4a所示，在整个恢复过程中，NeST 的特征相似性超过了 MiB，展示了 NeST 弥合模型稳定性差距的能力。图 4b表明，在正式训练过程中，NeST 的损失低于 MiB 的损失，有助于模型更快收敛并更好地学习。

### 4.3 消融实验

**不同的分类器初始化方法** 我们将 NeST 与其他初始化方法进行了比较。如 表 3所示，由于在训练早期阶段的不匹配，随机初始化表现效果最差。通过背景分类器进行初始化可以获得更好的性能。在正式训练步骤之前，直接微调通过背景分类器初始化的新分类器权重可以带来轻微的性能提升。与上述方法不同，NeST 使新分类器能够适应训练数据，并将基线方法在旧类别上的性能提升了 16.5%，在新类别上提升了 4.7%。

**表 3:** 不同分类器初始化策略的消融实验。在 Pascal VOC 2012 15-1 相交设置下测试 MiB [5] 的性能结果。

| **方法** | *0-15* | *16-20* | *all* |
|---|---|---|---|
| Random | 43.5 | 4.2 | 34.1 |
| Background [5] | 45.2 | 15.7 | 38.2 |
| Two-Stage | 46.0 | 15.3 | 38.7 |
| NeST (Ours) | 61.7 | 20.4 | 51.8 |

**表 4:** 不同变换矩阵初始化方法的比较。在 Pascal VOC 2012 15-1 相交设置下测试的性能结果。

| **方法** | *0-15* | *16-20* | *all* |
|---|---|---|---|
| Random Matrix Initialization | 53.8 | 7.5 | 42.8 |
| NeST (Ours) | 61.7 | 20.4 | 51.8 |

**变换矩阵初始化** 根据 表 4，如果使用随机初始化而不是我们设计的策略，整体性能仍然会高于基线方法 [5] 的性能。然而，在增量步骤中学习的新类别的 mIoU 显著低于基线方法的性能。这意味着如果我们随机初始化重要性

矩阵和投影矩阵，这只能帮助保持模型的稳定性。表 4的第二行显示，在使用我们提出的策略初始化重要性矩阵和投影矩阵后，新类别的性能提升到了20.4%，同时也提升了旧类别的性能。变换中使用的矩阵初始值设计可以促进模型对新类别的学习。因此，通过使用我们的策略来初始化矩阵，NeST可以在稳定性和可塑性之间实现平衡。

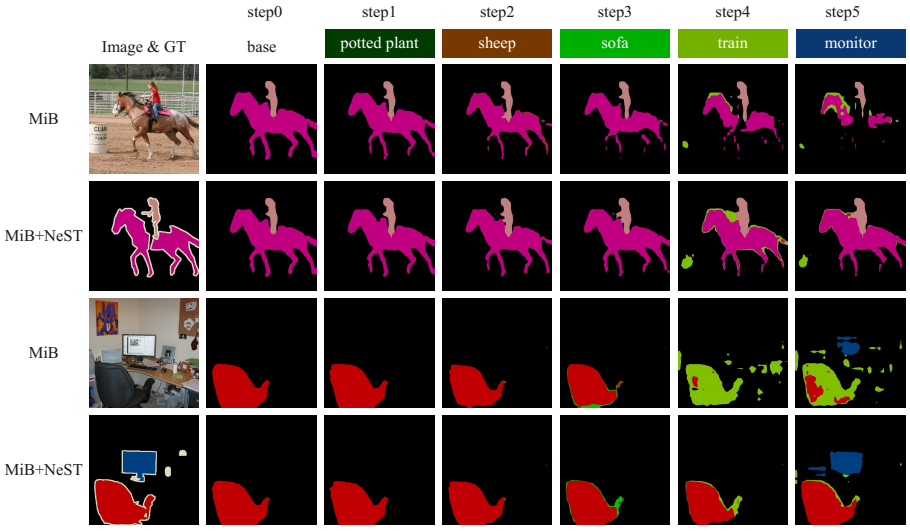



**图 5:** Pascal VOC 2012 *15-1* 相交设置下的定性比较。

**组件级别的消融实验** 我们对两种矩阵进行了组件级别的消融实验，如 表 5所示。消除重要性矩阵意味着我们将重要性矩阵中的值设置为 1 并冻结它，即仅学习旧分类器的组合来获得新分类器。同样，消除投影矩阵表明我们将加权的旧分类器进行平均以生成新分类器。实验结果表明，在对通道一视同仁的情况下，由于忽略了不同通道的差异，性能显著下降。

**计算成本** 额外参数的数量可以表示为 $n_{new} \times n_{old} \times (d+1) + d + n_{new} + 1$，其中 $n_{old}$ 随步骤线性增长，而 $n_{new}$ 始终固定。预调过程中额外的 FLOPs（约 15.6K）和参数（约 5.4K）在 *15-1* 的设置下是可以忽略不计的。MiB+NeST 的整个训练过程仅增加了 2.7% （3.7 分钟）的总训练时间 （2.3 小时）。请注意，在预调结束后，额外的参数将被移除，正式训练过程与 MiB 相同。

**关于 AWT 与 NeST 的讨论** AWT [21] 利用新的训练数据和旧背景分类器生成新分类器。我们在此讨论 AWT 和 NeST 之间的差异。AWT 采用

**表 5:** 在 Pascal VOC 2012 *15-1* 相交设置下的组件级消融实验。

| 方法 | 投影矩阵 | 重要性矩阵 | *0-15* | *16-20* | *all* |
|---|---|---|---|---|---|
| 基线方法 | | | 45.2 | 15.7 | 38.2 |
| 变量 | ✓ | | 53.3 | 15.4 | 44.3 |
| | | ✓ | 59.7 | 19.2 | 50.1 |
| | ✓ | ✓ | **61.7** | **20.4** | **51.8** |

**表 6:** Pascal VOC 2012 *15-1* 相交设置下的内存成本和额外训练时间。

| 方法 | 每个 GPU 的内存 | 额外时间 |
|---|---|---|
| MiB+AWT | $>$ 24GB | 6% |
| MiB+NeST (Ours) | **6.47GB** | **2.7%** |

**表 7:** Pascal VOC 2012 *15-1* 相交设置下五种不同类别顺序的平均性能。

| **方法** | *A* | *B* | *C* | *D* | *E* | avg |
|---|---|---|---|---|---|---|
| MiB | 38.2 | 28.2 | 38.5 | 38.0 | 52.0 | 39.0 |
| MiB+NeST (Ours) | 51.8 | 43.4 | 50.4 | 60.0 | 59.8 | **51.5** |

基于梯度的归因方法将最相关的权重传递给新分类器。然而，它对每个新类别一视同仁，忽略了旧分类器的作用和新类别之间的差异。同时，基于梯度的归因技术引入了巨大的显存开销，即使批次大小设置为 1，AWT 也无法在 RTX 3090 上运行。NeST 基于学习能够更好地使生成的分类器权重与骨干网络对齐。此外，如 表 6所示，NeST 的内存成本是可以接受的，并且比 AWT 略快。

**不同类别顺序的鲁棒性** 在 CISS 场景中，类别顺序至关重要，因为某一类别的学习进度可能会提升或损害另一类别的性能。为了验证 NeST 的鲁棒性，我们在 *15-1* 设置下对五种不同的类别顺序进行了实验。如 表 7所示，NeST 的性能要高于基线方法。

### 4.4 结果可视化

在 图 5中，我们展示了基于 MiB [5] 的本文方法测试的可视化结果。样本从步骤 0 和步骤 5 中选取，以展示 NeST 的稳定性和可塑性。在前两行中，类别 *person* 和 *horse* 在步骤 0 中学习，然后在接下来的步骤中，基线

方法 MiB [5] 逐渐忘记了步骤 0 中学习的概念，而 NeST 帮助模型保留了旧知识。在最后两行中，类别 *chair* 在步骤 0 中学习，类别 *tv/monitor* 在步骤 5 中学习。在学习类别 *train* 后，MiB 几乎忘记了类别 *chair* 的知识而 NeST 可以对大多数像素做出正确的预测。在最后一步中，NeST 可以帮助模型比 MiB 更好地学习新类别 *tv/monitor* 展示了我们方法的可塑性。

## 5 结论

在本文的工作中，我们提出了一种简单而有效的新分类器预调方法，通过学习从旧分类器到新分类器的变换来增强类增量语义分割（CISS）的能力。我们还找到了一种通过利用新旧类别之间跨任务类别相似性的信息来初始化矩阵的方法，帮助模型实现稳定性与可塑性之间的平衡。在两个数据集上的实验表明，NeST 可以显著提高基线方法的性能，并且可以轻松应用于其他 CISS 方法。尽管我们的方法可以带来巨大的性能提升，但仍然存在引入额外的计算开销带来的局限性。在未来的工作中，我们会尝试将我们的方法应用于其他持续学习任务中。

## 致谢

本工作得到了国家自然科学基金（NO. 62206135）、中国科学技术部国际合作重点项目（NO. 2024YFE0100700）、中国科协青年人才托举工程项目（NO. 2023QNRC001）、天津市应用基础和前沿研究计划青年项目（NO. 23JCQNJC01470）以及南开大学中央高校基本科研基金的资助。本研究的计算资源由南开大学超级计算中心提供支持。

## 参考文献

1. Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., Tuytelaars, T.: Memory aware synapses: Learning what (not) to forget. In: Eur. Conf. Comput. Vis. pp. 139–154 (2018)
2. Baek, D., Oh, Y., Lee, S., Lee, J., Ham, B.: Decomposed knowledge distillation for class-incremental semantic segmentation. In: Adv. Neural Inform. Process. Syst. (2022)
3. Bang, J., Kim, H., Yoo, Y., Ha, J.W., Choi, J.: Rainbow memory: Continual learning with a memory of diverse samples. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8218–8227 (2021)

4. Cermelli, F., Cord, M., Douillard, A.: Comformer: Continual learning in semantic and panoptic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3010–3020 (2023)
5. Cermelli, F., Mancini, M., Bulo, S.R., Ricci, E., Caputo, B.: Modeling the background for incremental learning in semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 9233–9242 (2020)
6. Cha, S., Yoo, Y., Moon, T., et al.: Ssul: Semantic segmentation with unknown label for exemplar-based class-incremental learning. In: Adv. Neural Inform. Process. Syst. vol. 34, pp. 10919–10930 (2021)
7. Chaudhry, A., Dokania, P.K., Ajanthan, T., Torr, P.H.: Riemannian walk for incremental learning: Understanding forgetting and intransigence. In: Eur. Conf. Comput. Vis. pp. 532–547 (2018)
8. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. **40**(4), 834–848 (2017)
9. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Eur. Conf. Comput. Vis. pp. 801–818 (2018)
10. Chen, Y.H., Chen, W.Y., Chen, Y.T., Tsai, B.C., Frank Wang, Y.C., Sun, M.: No more discrimination: Cross city adaptation of road scene segmenters. In: Int. Conf. Comput. Vis. pp. 1992–2001 (2017)
11. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1290–1299 (2022)
12. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention mask transformer for universal image segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1290–1299 (2022)
13. Cheng, B., Schwing, A., Kirillov, A.: Per-pixel classification is not all you need for semantic segmentation. In: Adv. Neural Inform. Process. Syst. vol. 34, pp. 17864–17875 (2021)
14. Cong, W., Cong, Y., Dong, J., Sun, G., Ding, H.: Gradient-semantic compensation for incremental semantic segmentation. IEEE Trans. Multimedia (2023)
15. De Lange, M., van de Ven, G., Tuytelaars, T.: Continual evaluation for lifelong learning: Identifying the stability gap. arXiv preprint arXiv:2205.13452 (2022)

16. Douillard, A., Chen, Y., Dapogny, A., Cord, M.: Plop: Learning without forgetting for continual semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. (2021)
17. Douillard, A., Cord, M., Ollion, C., Robert, T., Valle, E.: Podnet: Pooled outputs distillation for small-tasks incremental learning. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 86–102. Springer (2020)
18. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html (2012)
19. Gao, S.H., Cheng, M.M., Zhao, K., Zhang, X.Y., Yang, M.H., Torr, P.: Res2net: A new multi-scale backbone architecture. IEEE Trans. Pattern Anal. Mach. Intell. **43**(2), 652–662 (2019)
20. Geng, C., Huang, S.j., Chen, S.: Recent advances in open set recognition: A survey. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3614–3631 (2020)
21. Goswami, D., Schuster, R., van de Weijer, J., Stricker, D.: Attribution-aware weight transfer: A warm-start initialization for class-incremental semantic segmentation. In: WACV. pp. 3195–3204 (2023)
22. Grossberg, S.T.: Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control, vol. 70. Springer Science & Business Media (2012)
23. Guo, M.H., Lu, C.Z., Hou, Q., Liu, Z., Cheng, M.M., Hu, S.M.: Segnext: Rethinking convolutional attention design for semantic segmentation. In: Adv. Neural Inform. Process. Syst. vol. 35, pp. 1140–1156 (2022)
24. Hadsell, R., Rao, D., Rusu, A.A., Pascanu, R.: Embracing change: Continual learning in deep neural networks. Trends in cognitive sciences **24**(12), 1028–1040 (2020)
25. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conf. Comput. Vis. Pattern Recog. (2016)
26. Hu, Z., Li, Y., Lyu, J., Gao, D., Vasconcelos, N.: Dense network expansion for class incremental learning. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 11858–11867 (2023)
27. Hung, C.Y., Tu, C.H., Wu, C.E., Chen, C.H., Chan, Y.M., Chen, C.S.: Compacting, picking and growing for unforgetting continual learning. In: Adv. Neural Inform. Process. Syst. vol. 32 (2019)
28. Kim, D., Han, B.: On the stability-plasticity dilemma of class-incremental learning. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 20196–20204 (2023)

29. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences **114**(13), 3521–3526 (2017)
30. Li, Z., Hoiem, D.: Learning without forgetting. IEEE Trans. Pattern Anal. Mach. Intell. **40**(12), 2935–2947 (2017)
31. Lin, Z., Wang, Z., Zhang, Y.: Continual semantic segmentation via structure preserving and projected feature alignment. In: Eur. Conf. Comput. Vis. pp. 345–361. Springer (2022)
32. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Int. Conf. Comput. Vis. pp. 10012–10022 (2021)
33. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3431–3440 (2015)
34. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3431–3440 (2015)
35. Maracani, A., Michieli, U., Toldo, M., Zanuttigh, P.: Recall: Replay-based continual learning in semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7026–7035 (2021)
36. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: The sequential learning problem. In: Psychology of learning and motivation, vol. 24, pp. 109–165. Elsevier (1989)
37. Michieli, U., Zanuttigh, P.: Incremental learning techniques for semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 0–0 (2019)
38. Michieli, U., Zanuttigh, P.: Continual semantic segmentation via repulsion-attraction of sparse and disentangled latent representations. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1114–1124 (2021)
39. Oh, Y., Baek, D., Ham, B.: Alife: Adaptive logit regularizer and feature replay for incremental semantic segmentation. In: Adv. Neural Inform. Process. Syst. vol. 35, pp. 14516–14528 (2022)
40. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Int. Conf. Comput. Vis. pp. 12179–12188 (2021)
41. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. pp. 234–241. Springer (2015)
42. Shang, C., Li, H., Meng, F., Wu, Q., Qiu, H., Wang, L.: Incrementer: Transformer for class-incremental semantic segmentation with knowledge distillation focusing on old class. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7214–7224 (2023)

43. Shin, H., Lee, J.K., Kim, J., Kim, J.: Continual learning with deep generative replay. In: Adv. Neural Inform. Process. Syst. vol. 30 (2017)

44. Strudel, R., Garcia, R., Laptev, I., Schmid, C.: Segmenter: Transformer for semantic segmentation. In: Int. Conf. Comput. Vis. pp. 7262–7272 (2021)

45. Verwimp, E., De Lange, M., Tuytelaars, T.: Rehearsal revealed: The limits and merits of revisiting samples in continual learning. In: Int. Conf. Comput. Vis. pp. 9385–9394 (2021)

46. Vinogradova, K., Dibrov, A., Myers, G.: Towards interpretable semantic segmentation via gradient-weighted class activation mapping (student abstract). In: AAAI. pp. 13943–13944 (2020)

47. Wang, E., Peng, Z., Xie, Z., Yang, F., Liu, X., Cheng, M.M.: Unlocking the multimodal potential of clip for generalized category discovery (2024), `https://arxiv.org/abs/2403.09974`

48. Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., et al.: Deep high-resolution representation learning for visual recognition. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3349–3364 (2020)

49. Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., Fu, Y.: Large scale incremental learning. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 374–382 (2019)

50. Xiao, J.W., Zhang, C.B., Feng, J., Liu, X., van de Weijer, J., Cheng, M.M.: Endpoints weight fusion for class incremental semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7204–7213 (2023)

51. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: Segformer: Simple and efficient design for semantic segmentation with transformers. Adv. Neural Inform. Process. Syst. **34**, 12077–12090 (2021)

52. Yan, S., Xie, J., He, X.: Der: Dynamically expandable representation for class incremental learning. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 3014–3023 (2021)

53. Yang, G., Fini, E., Xu, D., Rota, P., Ding, M., Nabi, M., Alameda-Pineda, X., Ricci, E.: Uncertainty-aware contrastive distillation for incremental semantic segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **45**(2), 2567–2581 (2022)

54. Yoon, J., Yang, E., Lee, J., Hwang, S.J.: Lifelong learning with dynamically expandable networks. arXiv preprint arXiv:1708.01547 (2017)

55. Zhang, C.B., Xiao, J.W., Liu, X., Chen, Y.C., Cheng, M.M.: Representation compensation networks for continual semantic segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 7053–7064 (2022)

56. Zhang, Z., Gao, G., Jiao, J., Liu, C.H., Wei, Y.: Coinseg: Contrast inter-and intra-class representations for incremental segmentation. In: Int. Conf. Comput. Vis. pp. 843–853 (2023)
57. Zhang, Z., Zhang, X., Peng, C., Xue, X., Sun, J.: Exfuse: Enhancing feature fusion for semantic segmentation. In: Eur. Conf. Comput. Vis. pp. 269–284 (2018)
58. Zhao, B., Xiao, X., Gan, G., Zhang, B., Xia, S.T.: Maintaining discrimination and fairness in class incremental learning. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 13208–13217 (2020)
59. Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P.H., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 6881–6890 (2021)
60. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 633–641 (2017)
61. Zhu, F., Zhang, X.Y., Wang, C., Yin, F., Liu, C.L.: Prototype augmentation and self-supervision for incremental learning. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5871–5880 (2021)

## A 基线方法介绍

在本节中，我们会介绍实验中使用的基线方法。

**MiB.** 在 MiB [5] 中，使用两种损失来建模背景，即 $\mathcal{L}_{unce}$ 和 $\mathcal{L}_{unkd}$：

$$\begin{aligned}\mathcal{L}_{unce} &= -\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\log\tilde{q}_x^t(i,y_i)\,,\\ \mathcal{L}_{unkd} &= -\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\sum_{c\in\bigcup_{j=1}^{t-1}\mathcal{C}_j} q_x^{t-1}(i,c)\log\hat{q}_x^t(i,c)\,,\end{aligned} \tag{9}$$

其中 $\mathcal{I}$ 表示图像中的像素集，$y_i \in c_{bg}\cup\mathcal{C}_t$ 表示像素 $i$ 的真实标签，$q_x^t$ 表示模型在步骤 $t$ 中的输出，$\tilde{q}_x^t$ 和 $\hat{q}_x^t$ 表示当前模型的更新输出，分别考虑旧类别的交叉熵损失和新类别的知识蒸馏损失。

**PLOP.** 不同于 MiB [5]，PLOP [16] 通过伪标签解决背景漂移问题，如下所示：

$$\mathcal{L}_{pseudo} = -\frac{\nu}{WH}\sum_{w,h}^{W,H}\sum_{c\in\mathcal{C}_t}\tilde{S}(w,h,c)\log\hat{S}^t(w,h,c)\,, \tag{10}$$

其中 $\hat{S}$ 表示模型的预测结果，$\tilde{S}$ 表示前一步旧模型生成的伪标签。

PLOP 还通过 Local POD 蒸馏中间特征，如下所示：

$$\mathcal{L}_{LocalPod} = \frac{1}{L}\sum_{l=1}^{L}\left|\left|\Phi(f_l^t(I)) - \Phi(f_l^{t-1}(I))\right|\right|\,, \tag{11}$$

其中 $L$ 表示层的数量，$\Phi$ 表示 Local POD 提取操作，$f_l^t(I)$ 表示层 $l$ 对输入 $I$ 的输出特征。

**RCIL.** RCIL [55] 通过增加一个卷积层和归一化层的并行模块来解耦旧知识的记忆和新知识的学习，适用于每个 $3\times 3$ 卷积模块。在步骤 0，所有参数都是可训练的。在每个增量步骤中，分支之一的输出与来自前一分支的输出融合，以记住旧知识，而另一个分支是可学习的。融合两条分支的输出时，还使用了 drop path 策略，如下所示：

$$x_{out} = \eta\cdot x_1 + (1-\eta)\cdot x_2\,, \tag{12}$$

其中 $x_{out}$ 表示融合的输出，$x_1$ 和 $x_2$ 表示两条分支的输出，$\eta$ 表示逐通道的权重向量。在训练过程中，$\eta$ 从集合 $\{0, 0.5, 1\}$ 中采样，而在评估中 $\eta$ 设置为 0.5。RCIL 还提出了一种 Pooled Cube 知识蒸馏方法，使用空间和通道维度上的平均池化操作。

## B 不相交设置下的结果与分析

在不相交设置下，在每个步骤中，背景分类器将不会看到未来的任何类别，这导致 MiB 的初始化在这些设置中变得困难。相比之下，我们的 NeST 利用旧分类器的语义知识来生成新分类器进行初始化，预调过程也有助于模型的稳定性。NeST 和基线方法在 *15-1* 不相交和 *10-1* 不相交设置下的结果如 表 8所示结果表明 NeST 能够显著提升现有方法在不相交设置下的性能。

**表 8:** Pascal VOC 2012 数据集上不相交设置下的结果

| 方法 | *15-1* 不相交 | *10-1* 不相交 |
|---|---|---|
| MiB | 38.6 | 2.0 |
| MiB+NeST | **41.0** | **20.5** |
| PLOP | 40.7 | 12.6 |
| PLOP+NeST | **52.7** | **22.4** |

## C COCO-Stuff 10K 实验

为了证明 NeST 在包含更多类别的场景中的应用能力，我们引入了另一个用于类增量语义分割的数据集 COCO-Stuff 10K，以评估我们方法的有效性。COCO-Stuff 10K 包含 80 个实例类别和 91 个语义类别，这是原始 COCO-Stuff 数据集的一个子集。我们在 *80-91* (2 步) 的相交设置下评估了我们的 NeST 该设置包含比 ADE20K 和 Pascal VOC 2012 更多的类别。如 表 9所示，我们的方法可以处理在一个步骤中包含更多类别的场景。

**表 9:** COCO-Stuff 10K 数据集上 *80-91* 相交设置下的结果

| 方法 | *0-80* | *81-171* | *all* |
|---|---|---|---|
| MiB | 40.2 | 18.6 | 28.9 |
| MiB+NeST | 41.9 | 20.7 | **30.7** |
| PLOP | 46.1 | 17.0 | 30.8 |
| PLOP+NeST | 46.0 | 18.8 | **31.6** |

## D 基于 Transformer 的 SOTA 方法比较

今年来，出现了许多基于 Transformer 的 CISS 方法，在这里我们简要讨论 NeST 与这些方法之间的差异。Comformer [4] 使用通用分割模型 Mask2Former [12] 进行掩码分类，用于统一的全景分割和持续语义分割。CoinSeg [56] 引入了预训练的 Mask2Former 模型作为类别无关的掩码生成器。Incrementer [42] 依次添加新类别的标记，并在特征与更新的类标记之间进行点积以生成分割预测结果。*基线方法*是基于简单的逐像素分类模型 SETR，使用 ViT-B 作为骨干网络，而将 NeST 加入其配置可以实现 SOTA 的性能效果，如 表 10所示。此外，NeST 具有应用到这些基于 Transformer 的方法中的潜力，我们未来将针对该部分进行更多的探索。

**表 10:** 与基于 Transformer 的 SOTA 方法的比较

| 方法 | 骨干网络 | 模型 | *15-1* | *15-5* | *10-1* |
|---|---|---|---|---|---|
| CoinSeg | Swin-B | Deeplab+Mask2Former | 75.5 | 77.6 | 70.5 |
| Incrementer | ViT-B | Segmenter | 75.5 | 79.9 | 70.2 |
| MiB | ViT-B | SETR | 53.3 | 80.2 | 25.5 |
| MiB+NeST (Ours) | ViT-B | SETR | **76.5** | **80.3** | **71.9** |

## E 更多实现细节

**权重对齐** 为了防止在预调过程中新分类器的权重过大，我们应用了权重对齐 (WA) [58]，具体公式如下：

$$\hat{w}_{new} = w_{new} \cdot \frac{Mean(Norm_{old})}{Mean(Norm_{new})} \tag{13}$$

其中，$Norm_{old}$ 和 $Norm_{new}$ 分别表示旧分类器和新分类器权重的范数，$Mean(\cdot)$ 表示计算均值的操作。 表 11的相关结果表明，WA 可以纠正偏置权重，从而提升 NeST 的性能。

**冻结旧分类器** 我们发现伪标签策略可能会改变旧分类器的几何结构，这对我们的方法产生了不利影响。这一点在 Pascal VOC 2012 数据集上尤其明

**表 11:** NeST 的权重对齐消融实验，所有的性能效果均在 *15-1* 设置下进行测试

| **方法** | *0-15* | *16-20* | *all* |
|---|---|---|---|
| MiB+NeST w/o WA | 58.4 | 10.9 | 47.1 |
| MiB+NeST w/ WA | 61.7 | 20.4 | **51.8** |
| PLOP+NeST w/o WA | 72.5 | 32.4 | 62.9 |
| PLOP+NeST w/ WA | 72.2 | 33.7 | **63.1** |

显。为了保留先前步骤中学习到的旧知识，我们按照 EWF [50] 的方法，在 Pascal VOC 2012 设置的正式训练步骤中冻结旧分类器。相关实验如 表 12所示。

**表 12:** 基于 PLOP [16] 的 NeST 针对冻结旧分类器的消融实验，所有性能效果均在 *15-1* 设置下进行测试

| **方法** | *0-15* | *16-20* | *all* |
|---|---|---|---|
| PLOP w/ fix | 56.9 | 11.3 | 46.0 |
| PLOP+NeST w/o fix | 66.8 | 20.2 | 55.7 |
| PLOP+NeST w/ fix | 72.2 | 33.7 | **63.1** |

## F 进一步分析

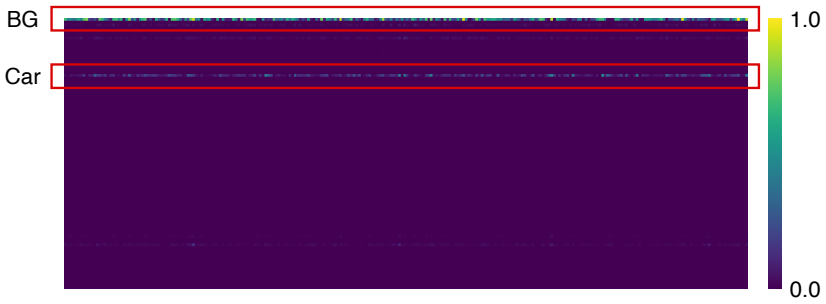



**图 6:** ADE20K *100-5* 设置下步骤 1 的重要性矩阵可视化。

**重要性矩阵的有效性** 为了增强可塑性，我们学习了如何与相关的旧分类器生成一个新分类器。特别地，重要性矩阵可以帮助我们更好地捕捉语义关系，从而帮助旧分类器在通道级别上对齐。为了验证这一点，我们在 ADE20K 的 *100-5* 设置的步骤 1 包含的新类 *van* 上可视化了重要性矩阵 $M \in \mathbb{R}^{100\times256}$。我们将所有绝对值归一化为 $[0, 1]$，并为背景类别设置了较高的对比度以提高可视化效果。如 图 6所示，背景类别 (第 0 行) 和汽车类别 (第 21 行) 做出了最大的贡献。这很明显，因为当类名在训练数据中标记为背景时，汽车和面包车旧类别在语义上是相似的。

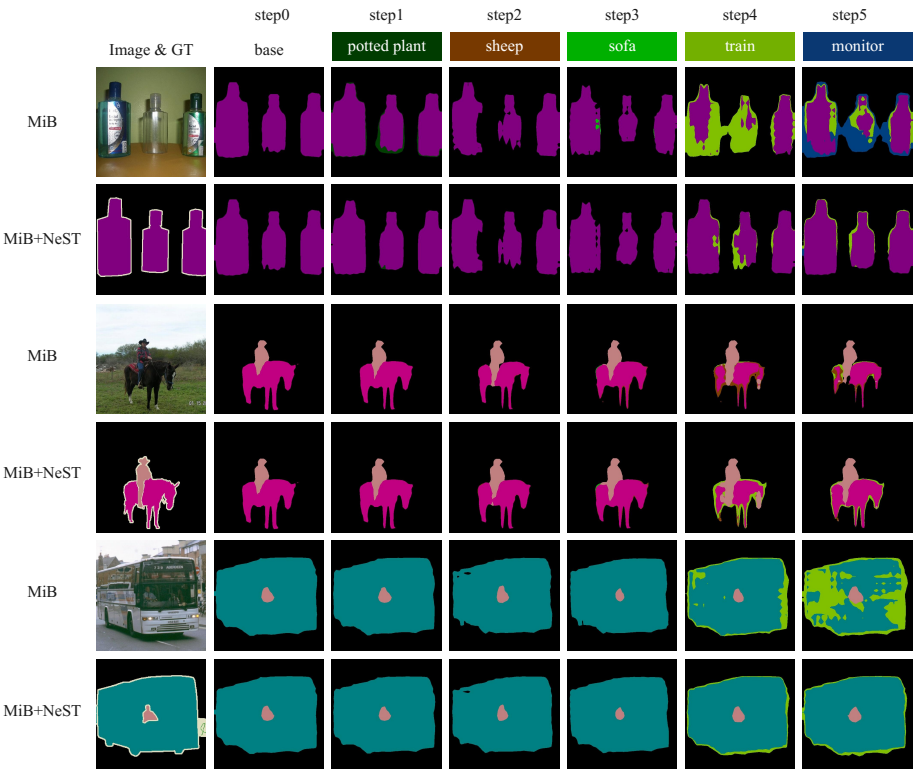



**图 7:** 更多定性结果。所有实验均在 *15-1* 设置下进行。

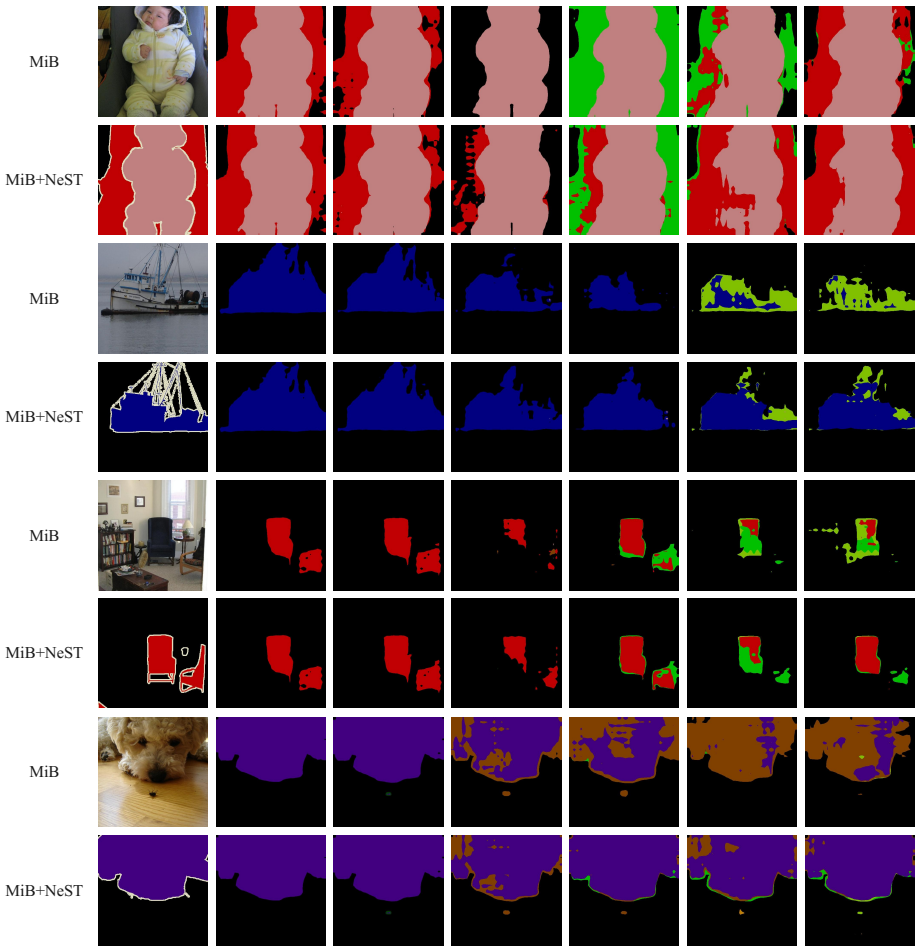



**图 8:** 更多定性结果。所有实验均在 *15-1* 设置下进行。

**不同的类别顺序** 为了评估我们方法的有效性，基于 PLOP [16] 我们在 Pascal VOC 2012 的 *15-1* 相交设置下进行五种不同类别顺序的实验。如下：

$$
\begin{aligned}
A &: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20] , \\
B &: [0, 12, 9, 20, 7, 15, 8, 14, 16, 5, 19, 4, 1, 13, 2, 11, 17, 3, 6, 18, 10] , \\
C &: [0, 13, 19, 15, 17, 9, 8, 5, 20, 4, 3, 10, 11, 18, 16, 7, 12, 14, 6, 1, 2] , \\
D &: [0, 15, 3, 2, 12, 14, 18, 20, 16, 11, 1, 19, 8, 10, 7, 17, 6, 5, 13, 9, 4] , \\
E &: [0, 7, 5, 3, 9, 13, 12, 14, 19, 10, 2, 1, 4, 16, 8, 17, 15, 18, 6, 11, 20] .
\end{aligned}
\tag{14}
$$

**更多定性结果** 在 图 7和 图 8中展示了更多定性结果。通过应用预调过程，我们的方法可以帮助模型保留旧知识。

此外，为了验证矩阵初始化的有效性，我们还可视化了最后一个类别 *tv/ monitor* 的类激活图。为了可视化分割 CAMs，我们采用了 [46] 中提出的方法。如 图 9所示，通过设计的矩阵初始化策略，模型可以更多地将注意力放在新类别的区域。

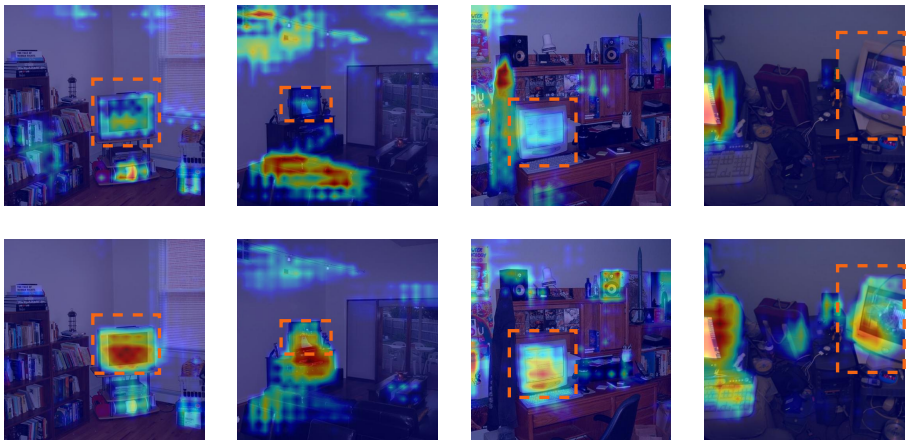


**图 9:** 最后一个类别 *tv/monitor* 在 *15-1* 设置下的类激活图，分别展示了没有 (图片顶部) 和有 (图片底部) 我们的矩阵初始化策略的情况。