手术 SAM：高效、易于分类的手术器械分割

摘要

本文复现中聚焦强大的基础模型 Segment Anything Model（SAM），其革新了图像分割领域。在将 SAM 应用于手术器械分割时，常用方法是定位精确点或器械盒，以零射击方式作为 SAM 提示。但复现发现原始管道存在两大问题：一是自然物体与手术器械的域间隙，致使 SAM 泛化性差；二是 SAM 依赖精确的点或框位置实现精准分割，需大量人工指导或性能良好的专业检测器预先准备，导致复杂的多阶段管道。为解决上述问题，本文引入 SurgicalSAM，这是一种新颖的端到端高效适配 SAM 的方法，能有效融合手术特定信息与 SAM 预训练知识，提升泛化能力。具体提出基于轻量级原型的类提示调优编码器，直接从类原型生成提示嵌入，摒弃显式提示，增强鲁棒性并简化流程。针对手术器械类别间类间差异小的问题，提出对比原型学习，强化类原型辨别力，获取更精准类提示。本文在此基础上进一步改进，引入实例平衡重采样（IBR）方法，通过增加尾类样本的采样频率校准分类器权重中的共享特征。尽管采取了 IBR 方法，对 ndoVis2018 数据集的实验结果表明，SurgicalSAM 在实现先进性能的同时，加入 IBR 方法后整体效果并未产生明显变化。

关键词：SAM；弱监督；语义分割；重采样

1 引言

手术器械分割（Surgical Instrument Segmentation，SIS）是手术视觉领域中的一项核心任务，其目标是精确识别和勾画手术场景中的手术器械轮廓。这项技术在辅助外科医生开展手术操作以及推动先进计算机辅助手术系统的发展方面具有重要意义。目前，基于深度学习的方法在 SIS 任务中表现出了卓越的性能，这些方法通过设计和训练包含特定任务组件的专家模型，能够实现高精度分割。然而，这类方法通常需要依赖 SIS 数据集训练出完整的模型参数集，效率较低。此外，由于现有 SIS 数据集规模有限，训练得到的模型在面对未知场景时常表现出较差的泛化能力。

在计算机视觉领域，语义分割与 SIS 在技术需求上存在一定相似性，但传统的基于像素级标注的语义分割方法往往需要大量的人工成本，特别是在大规模数据集上，精确标注的获取更加困难。为了降低标注成本，弱监督语义分割（Weakly Supervised Semantic Segmentation，WSSS）逐渐成为研究热点。WSSS 通过利用较弱的监督信息（如图像级标签、点标注或边界框标注）进行训练，在大幅降低数据标注成本的同时，也实现了令人可喜的分割效果 [1]。

其中，基于图像级标签的 WSSS 方法备受关注，其关键在于从分类模型中提取类别激活图（Class Activation Map，CAM）作为伪标签，用于监督训练分割模型。然而，这一过程中仍

然存在许多挑战，例如，CAM 往往会出现噪声、边界模糊，以及过度集中于最具判别力区域等问题，从而导致分割结果不够完整和精准 [3]。此外，长尾分布问题在训练数据中广泛存在，即头类（样本丰富）容易过度激活，而尾类（样本稀少）激活不足，这进一步限制了 WSSS 的性能。

2 相关工作

为了应对这些挑战，研究者们开始尝试引入利用图像级标签训练分类器，通过分析分类器的决策过程获取像素级信息，如类别激活图 Class Activation Map(CAM) [6]。首先，使用图像级标签训练深度神经网络，使其学习到不同类别的整体图像表示。然后，分析分类器的中间层特征或权重生成 CAM，CAM 中激活度较高的区域表示对任务决策贡献较大的像素，进而将图像级标签传播到像素级，得到伪标签。最后，利用这些伪标签训练语义分割模型。这种方法适用于语义分割任务，尤其在处理大规模图像数据时，可以在仅使用图像级标签的情况下获得像素级语义信息，减少对像素级标注的依赖，降低标注成本。例如，Zhao 等人 [5] 提出了一种结合共享特征校准（SFC）与多尺度对比学习的 CAM 优化方法，通过调整共享特征比例，平衡头尾类激活效果，显著改善了长尾分布对分割性能的影响。

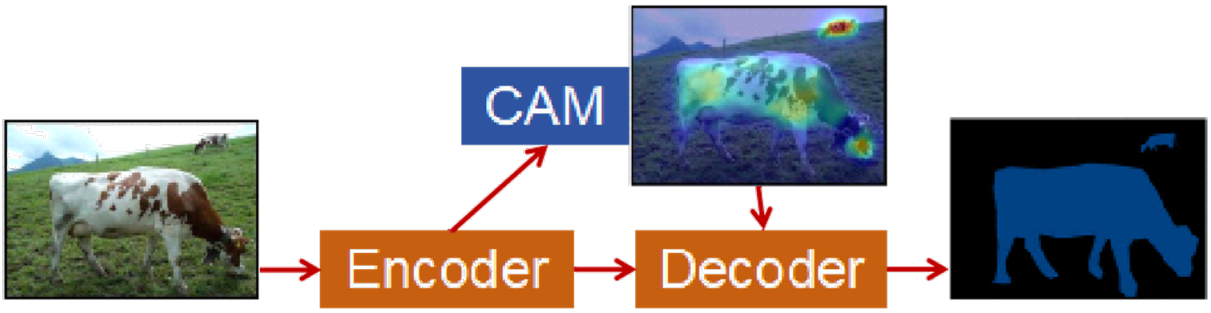

图 1. CAM 方法示意图

此外，近期研究尝试引入 Segment Anything Model（SAM）这一大规模预训练基础模型，以增强 WSSS 的伪标签质量。SAM 具备强大的泛化能力，通过 prompt（提示）驱动的方式在零样本场景下实现分割。通过引入自监督机制与对比学习策略，将 SAM 的无监督分割结果与 CAM 有效结合，生成边界更清晰、完整度更高的伪标签。最终，形成一个高效的弱监督语义分割框架，既能降低数据标注成本，又能提升分割性能与泛化能力。而在医疗图像处理领域中，基础模型 SAM 主要在自然图像上训练，在弱监督任务中直接应用会出现 SAM 与 CAM 生成的分割掩码之间存在显著域间差异，导致泛化性能不佳等问题。

Segment Anything Model（SAM）作为一种具有开创性的快速分割基础模型，近年来得到了广泛关注。将 SAM 用于下游的医疗任务，对于提高训练效率和利用强大的预训练知识具有很大的前景。目前的研究主要采用零样本方式的 SAM 进行医学图像分割。然而，SAM 预训练中缺乏足够的医学数据，且自然物与医学目标之间存在较大的领域鸿沟，阻碍了 SAM 对医学任务的直接泛化。此外，SAM 的性能依赖于显式 prompt 的精确位置。为了解决上述挑战，论文提出了 SurgerySAM，一种端到端的方法，通过对 SAM 的有效调整，有效地缓解了手术-自然域之间的鸿沟。SurgerySAM 与现有 pipeline 的比如 2所示。论文提出了一种基于轻量级原型的类提示编码器，它将一个仪器类作为提示，通过与图像嵌入交互来学习类原

型，直接为掩码解码器生成提示嵌入。通过调整基于原型的类提示编码器和掩码解码器，将手术知识与 SAM 的预训练知识相融合，有效地缓解了领域鸿沟。

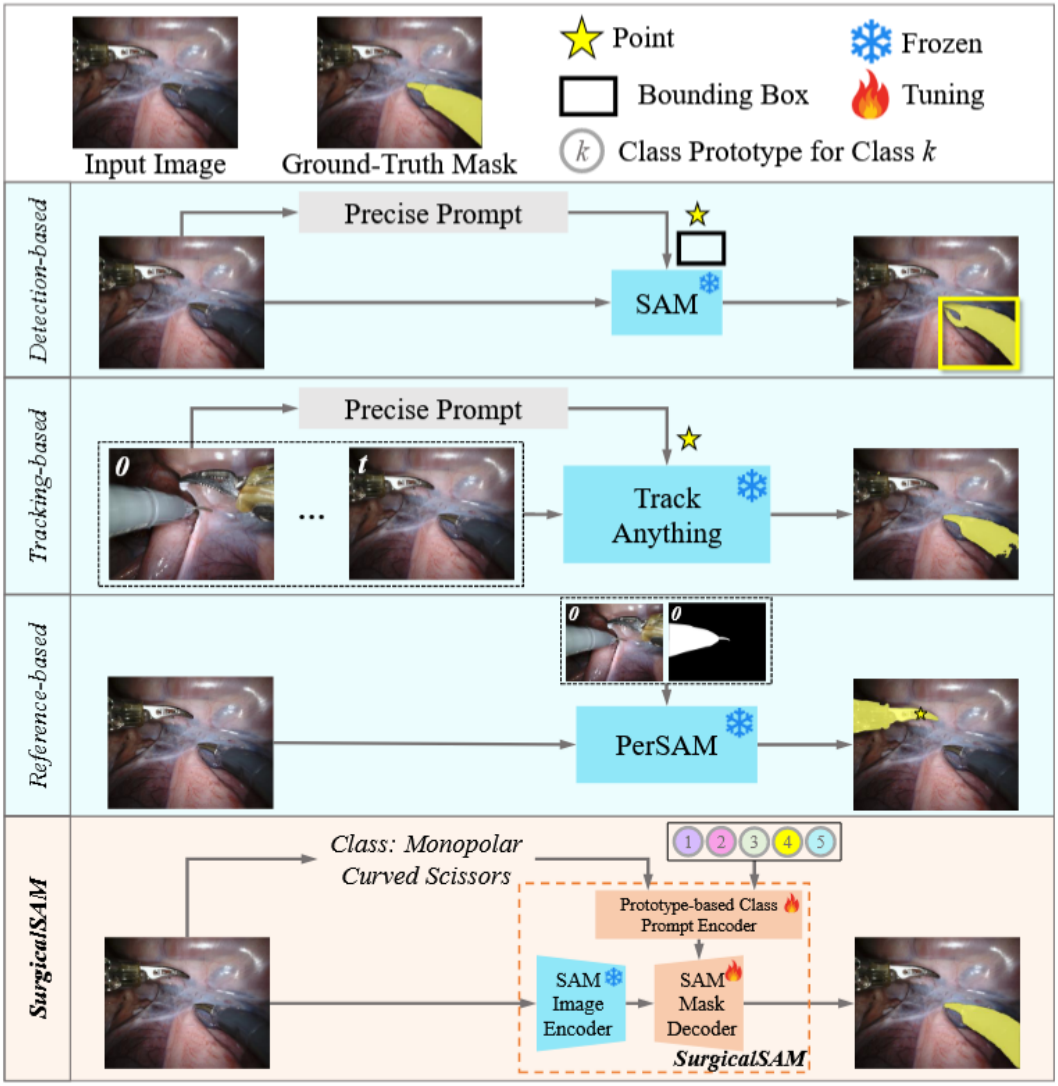

图 2. SAMs' pipeline

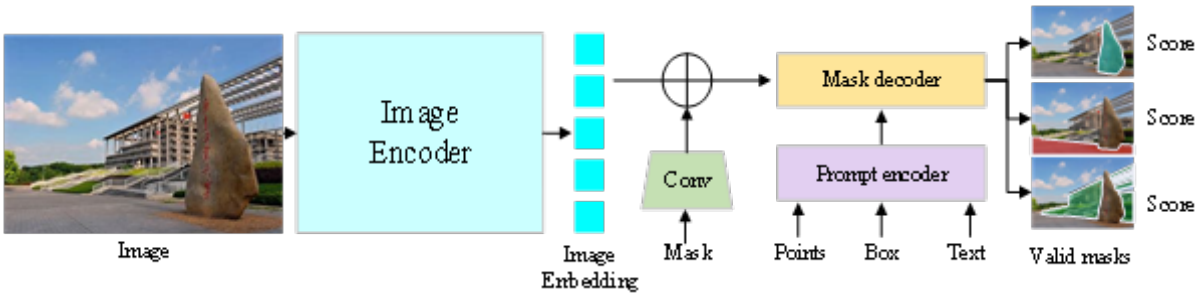

图 3. SAM 方法示意图

2.1 一般 SAM 方法

为了充分发挥预训练 SAM 的独特优势，本文在设计上力求保持 SAM 原始模型结构的完整性，如图 3所示。SAM 模型作为一个整体，其架构基于强大的 Transformer 模型构建，由

三个关键部分组成：图像编码器、提示编码器和掩码解码器。这些组件协同工作，使 SAM 能够实现对图像中任意目标的高质量分割。

（1）图像编码器（Image Encoder）：SAM 使用经过 MAE 预训练的 Vision Transformer（ViT）[2] 作为图像编码器，采用 H/16 模型的窗口注意力和四个等距的全局注意力块。图像编码器的输出为输入图像的 16 整除倍数的降采样嵌入。在输入维度上，SAM 的输入分辨率为 1024×1024，最终图像嵌入大小为 64×64。此外，为了降低通道维度，使用大小为 1×1 的卷积层将通道数减少到 256。随后，使用 256 通道的 3×3 卷积层，然后在每个卷积层之后进行层归一化。经过多层卷积和池化操作后，图像编码器将输出一个固定大小并包含了图像全局特征的特征向量。这个特征向量可以作为后续模型（如解码器）的输入，用于图像分割或其他相关任务。

（2）提示编码器（Prompt Encoder）：SAM 的提示编码器接受稀疏提示（目标点、矩形框、文本）和密集提示（掩码）两种类型的提示。对于稀疏提示，例如目标点和矩形框，以位置编码的形式表示，并添加到每个提示类型的学习嵌入中。Prompt Encoder 能够将输入的提示映射为特定维度的向量化嵌入，例如 Box Prompt 由一对表示左上角和右下角位置编码的坐标表示，框内区域表示所需分割的大致区域，在编码后映射嵌入到网络中。对于密集提示，掩码与图像具有空间对应关系，SAM 以低于输入图像 4 倍的分辨率摄取掩模，然后应用两个步长为 2 的 2×2 卷积，分别产生 4 个和 16 个输出通道，导致进一步的降采样 4。最后，一个 1×1 的卷积层将信道维度映射为 256，经处理后逐元素添加掩膜嵌入和图像嵌入。

（3）掩码解码器（Mask Decoder）：掩码解码器有效地将图像嵌入和一组提示嵌入映射为输出掩码。为了结合这些输入，SAM 从 Transformer 分割模型中得到启发，并修改了标准的 Transformer 解码器。在应用解码器之前，将学习到的输出令牌嵌入插入到提示嵌入的集合中，用于解码。SAM 中的每个解码器层执行 4 个步骤：

i. Self-Attention: 解码器中的每个注意力头会计算输入嵌入的自注意力权重，以捕捉图像和提示嵌入之间的上下文关系。这有助于模型理解输入图像中不同区域之间的关联性；

ii. Cross-Attention: 在自注意力计算之后，解码器会执行交叉注意力操作。交叉注意力将图像嵌入和提示嵌入作为查询和键值对，并处理为 64×64 个 256 维向量的集合，以帮助模型将图像中的相关特征与提示中的目标信息对齐；

iii. Feed-Forward Network: 在注意力计算之后，解码器会通过一个前馈网络来处理每个位置的嵌入。这个前馈网络通常由多个全连接层组成，用于进一步提取和转换特征；

iv. Layer Normalization and Residual Connection: 在每个解码器层中，都会应用层归一化来稳定训练过程，并使用残差连接来将输入和输出特征进行组合。最后一步使用提示信息来更新图像嵌入。每个 Self-attention、Cross-attention 和 MLP 在训练时都有残差连接、层归一化和 0.1 的 dropout 率，这有助于模型在深度网络中有效地传递信息。

下一个解码器层利用上一层更新的令牌和图像嵌入，且 SAM 总共使用了两个解码器层。

2.2 SurgicalSAM

如图 2展示了 SurgicalSAM 与其他 SAM 相关方法的不同之处，通过对比 detection-based、tracking-based 和 reference-based 的 SAM 变体，突出了 SurgicalSAM 的独特设计和优势。在

detection-based 方法中，SAM 通过精准提示（如点或边界框）生成分割掩码，但其依赖精确提示的能力对复杂场景的泛化较弱；在 tracking-based 方法中，利用时间序列中的帧间连续性，虽然可以增强目标的分割效果，但需要额外的时间信息并不适用于单帧图像分割；而 reference-based 方法通过引用参考图像与目标图像的匹配关系生成分割结果，但在手术场景中，因器械形态变化和遮挡问题，参考匹配的效果往往受限。相比之下，SurgicalSAM 在架构上融合了基于类的提示编码器，通过从输入图像中提取类别原型并与 SAM 图像编码器和掩码解码器联动，直接生成与目标器械类别相关的提示嵌入。这种方法不仅减轻了对显式提示（如点或边界框）的依赖，同时有效缓解了自然图像和手术图像之间的领域差异，显著提升了分割精度和泛化能力，特别适用于复杂的医疗场景。

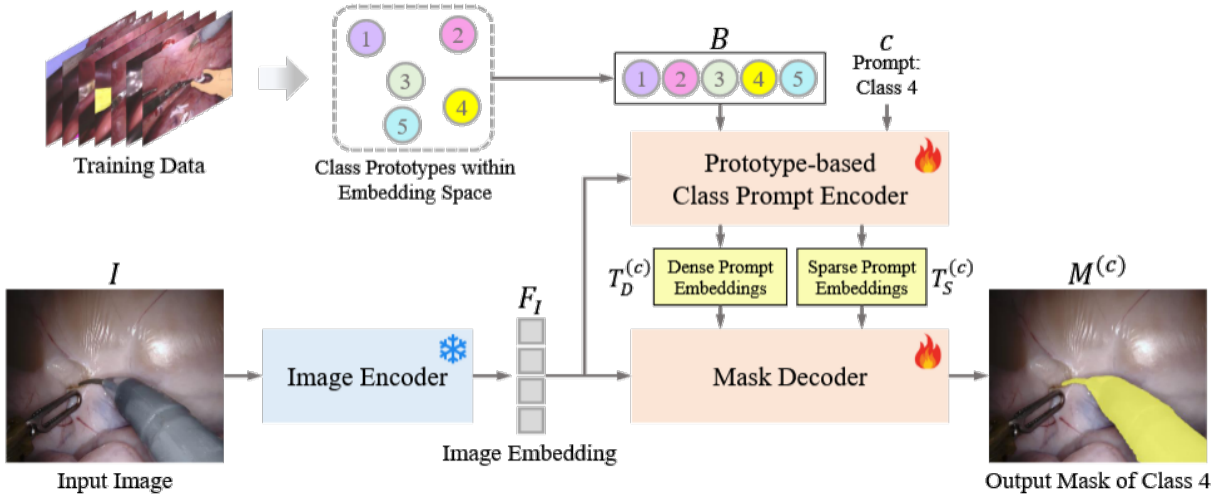

图 4. SurgicalSAM 方法示意图

SurgicalSAM [4] 由图像编码器、原型类提示编码器 Prototype-based Class Prompt Encoder（PCPE）和掩码解码器三个核心组件组成，旨在通过高效调优 SAM，实现端到端的手术器械分割。给定一张手术图像，图像编码器首先提取图像的特征嵌入；原型类提示编码器基于类别原型与图像嵌入进行交互，生成密集和稀疏的提示嵌入，捕捉类别相关的信息；掩码解码器在此基础上将图像嵌入与提示嵌入融合，预测出图像中特定类别的分割掩码。在训练过程中，通过对比原型学习 Contrastive Prototype Learning（CPL）在框架调整中获优化原型。基于 infoNCE 损失的原型对比损失，以类别原型为锚点、SAM 基类嵌入为样本，加强同类相似性、抑制异类相似性，得更具区分性原型表示，借 SAM 调整增强手术领域知识注入，有效提升分割结果的准确性和鲁棒性。

为解决 SAM 对手术器械分割中精确提示的依赖问题，ECP 通过利用类原型与图像嵌入的相似性生成提示嵌入，从而消除对显式提示（如点或框）的需求，实现更鲁棒的分割。如图 9 所示网络首先构建一个包含每个类代表原型的原型库 B，计算图像嵌入与所有类原型的空间相似度矩阵 S，通过相似度矩阵激活类特定区域，得到类激活特征。然后利用类激活特征分别生成密集和稀疏的提示嵌入，其中密集提示嵌入通过对正类（提示类别）的类激活特征进行两层多层感知机（MLP）处理得到，稀疏提示嵌入则基于所有类的类激活特征，通过区分正负类信息（利用一对正负嵌入）得到。

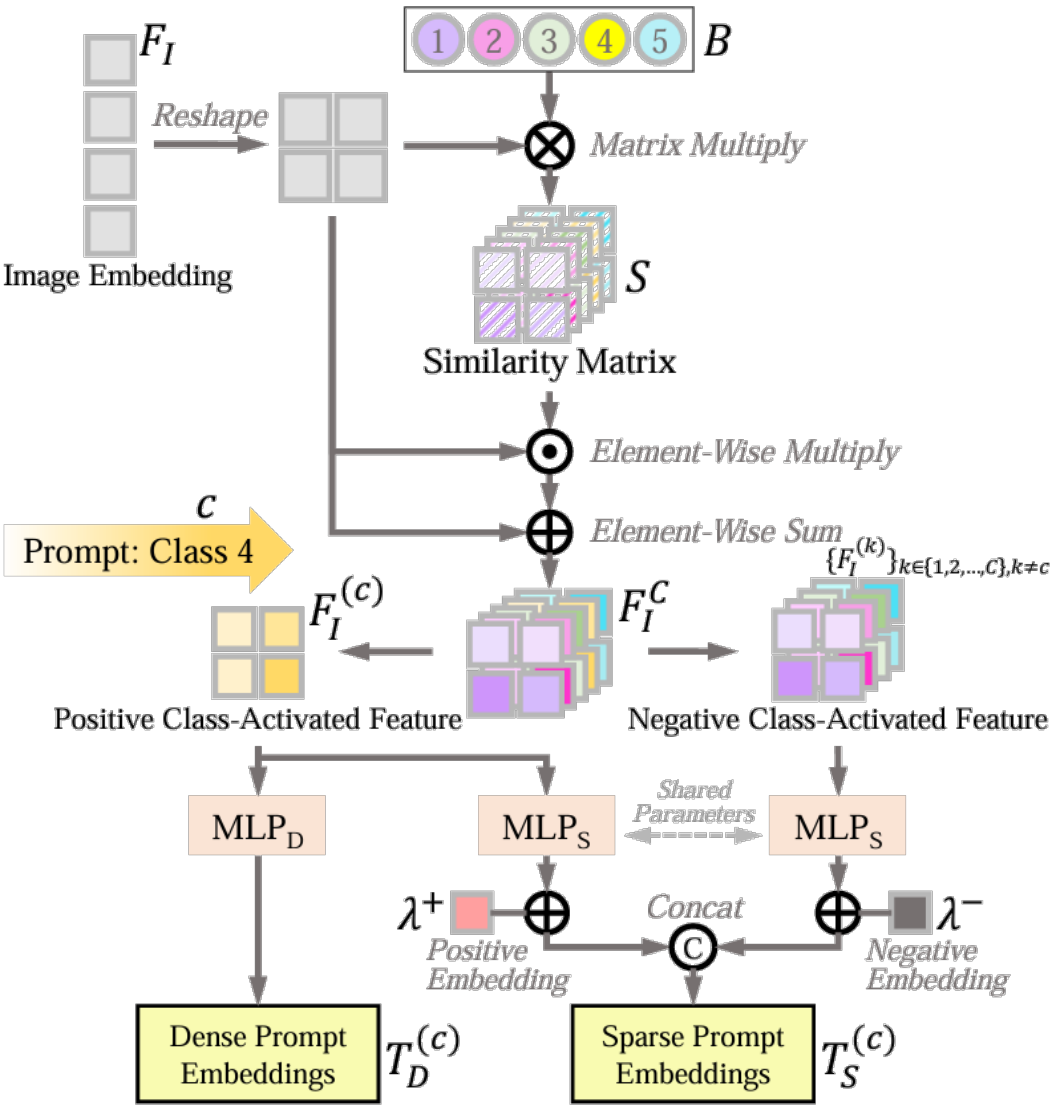

图 5. PCPE 模块结构

由于手术器械类别之间的相似度较高且差异较小，如何有效区分不同类别的器械，成为实现精确分割的关键。为了提升类原型的判别能力，使模型能够更加准确地识别和区分这些相似的类别，论文提出了一种创新的原型对比损失方法。该方法将类原型视为锚点，将图像中的基于 SAM 生成的类嵌入视为样本。如图 6所示，通过计算类原型与样本之间的相似度，原型对比损失强化了正样本（同类器械）与类原型之间的相似度，使得模型能够更加聚焦于同一类器械的特征表达；同时，抑制负样本（不同类器械）与类原型之间的相似度，减少不同类别之间的混淆。这种对比学习策略使得类原型能够更加准确地代表每个类别的特征，优化了类原型的表示，从而提升了模型在特征层面上对类别差异的理解能力。最终，原型对比损失使得模型在面对高度相似的器械类别时，能够更加精确地进行区分，显著提高了手术器械分割的精度和鲁棒性。

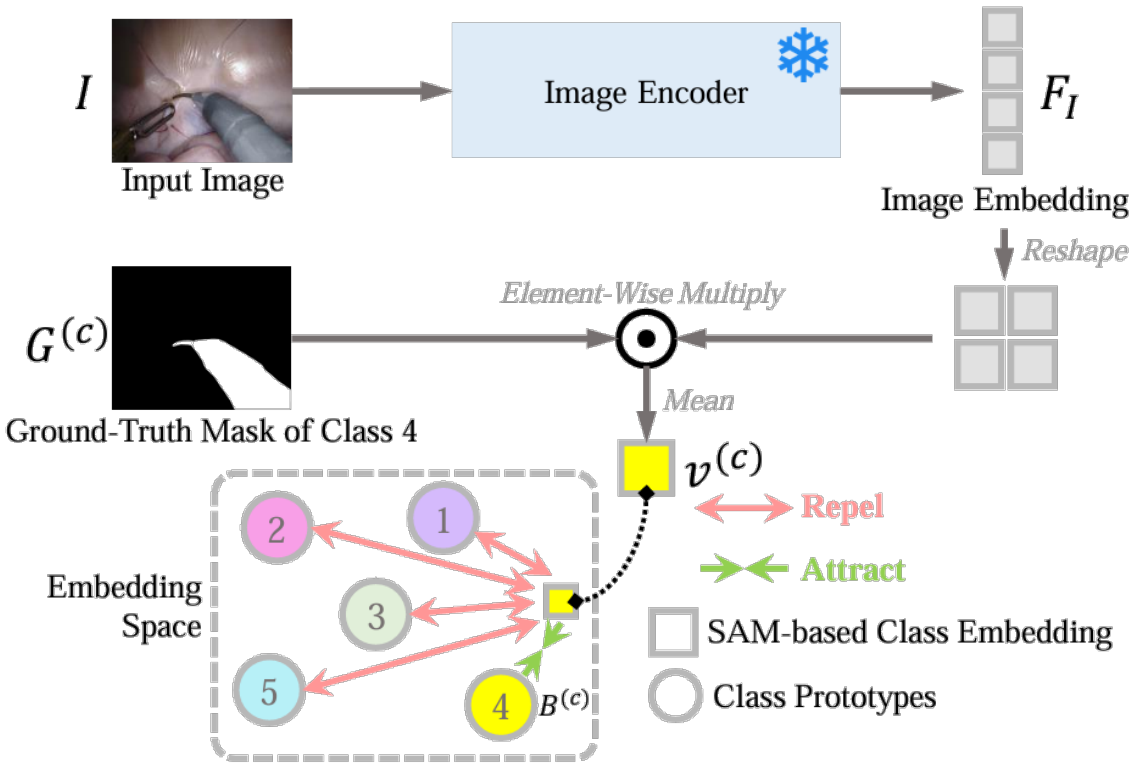

图 6. CPL 方法流程图

3 本文方法

在 SurgicalSAM 的基础上，我们进一步借鉴了 Shared Feature Calibration（SFC）[5] 方法（图 7）中的 Image Bank Re-sampling（IBR）机制，针对手术器械分割任务中特殊的长尾分布问题进行优化。在手术场景中，不同类型的手术器械在数据集中往往分布不均，部分器械类别（如常见器械）样本丰富，而稀有器械类别则数据匮乏。这种数据不平衡会导致 SurgicalSAM 中的类提示编码器对尾类（少数手术的专用器械）的特征捕捉不足，从而影响分割精度。为解决这一问题，我们通过在 SurgicalSAM 框架中引入 IBR 机制，构建一个图像库并均匀采样各类别的样本，特别是增加尾类的采样频率，从而使模型在训练时更关注这些样本较少的类别。这种调整能够显著优化分类器中与尾类相关的共享特征分布，避免模型因数据不平衡而对尾类产生忽视或偏差。此外，IBR 机制进一步增强了类提示编码器的表现，使 SurgicalSAM 能够在复杂的长尾分布场景下生成更加准确和全面的分割掩码，提升了模型对尾类器械的泛化能力。

IBR 通过增加尾类样本的采样频率，来校准分类器权重中的共享特征。具体做法是，论文中维护了一个图像库，专门存储每个前景类的图像。在训练过程中，当某个类别的图像出现时，就更新图像库中该类别的图像；如果该类别的图像没出现，则图像库中的数据保持不变。接下来，系统会从图像库中均匀采样一定数量的图像，并将这些图像与原始训练批次的图像合并，作为最终的训练输入。通过增加尾类样本的采样频率，IBR 使得 MSDW 损失能够更频繁地作用于尾类，从而调整尾类在分类器权重中共享特征的比例。从某个角度来说，IBR 就像是在“强化”尾类的训练，让模型在学习时更重视这些样本较少的类别，从而避免模型对这些类别的忽视或偏见，让模型能够更平衡地处理头类和尾类的特征，进而优化 CAM 的

生成。

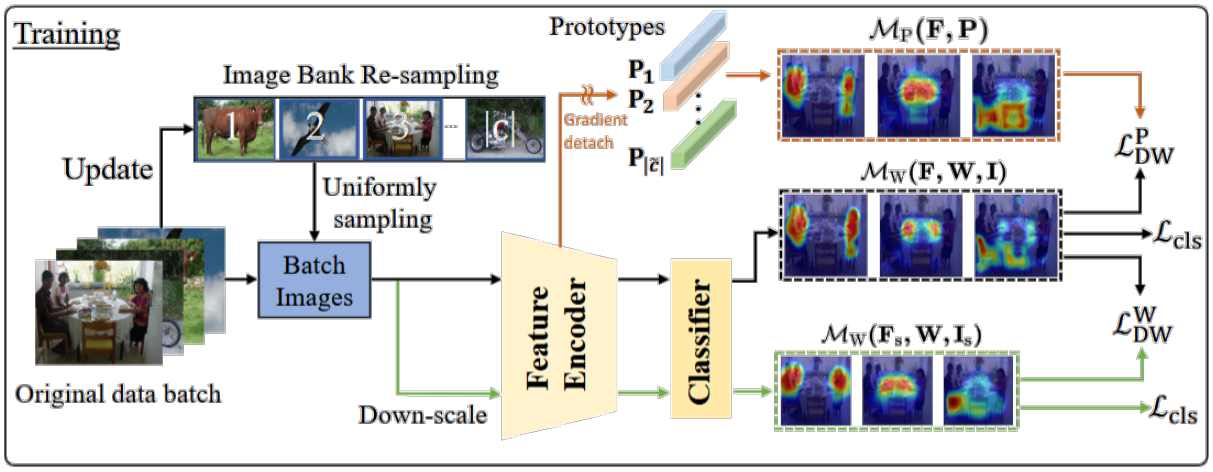

图 7. SFC 方法示意图

4 复现细节

首先构建了一个基于训练数据的图像库，其中包含了不同类别的器械样本，并通过 IBR 机制对图像库进行均匀采样。特别地，IBR 显式提高了尾类样本的采样频率，从而在训练过程中平衡了模型对头类和尾类样本的关注度。对于均匀采样后的图像，我们使用类提示编码器提取类别原型，将其与 SAM 图像编码器生成的特征嵌入进行交互学习。这一过程显著调整了模型中与尾类相关的共享特征分布，确保尾类特征在分类器权重中被充分学习，避免了由于数据不平衡导致的特征忽视或偏差。此外，通过 IBR 增强的类提示编码器进一步优化了 SurgicalSAM 的掩码解码器，使其能够生成更加完整且精准的分割掩码。

在数据预处理阶段，本文与复现论文设置的一样，对输入图像进行了统一的裁剪和归一化处理，将所有图像调整为尺寸 1024×1280，并归一化到 [0, 1] 范围。此外，为了应对类别不平衡问题，我们结合 IBR 机制对数据进行了重采样，并通过 Albumentations 库实现数据增强，包括随机旋转、缩放、翻转和颜色扰动等操作，以提升模型的鲁棒性。

实验的超参数设置包括：学习率为 1e-3，优化器采用 Adam，权重衰减系数为 1e-4，原型对比损失的温度为 0.07。训练过程使用动态批量大小（根据 GPU 显存动态调整，最大批量大小为 32），总训练轮数为 50，并采用早停策略防止过拟合。在所有实验中，同样固定随机种子为 666 以确保结果的可重复性。

4.1 与已有开源代码对比

在原有的 Surgical 代码中加入 IBR 策略 [5]，其代码及注释如下，

```
# 初始化存储图像的slot张量，形状为[7, 3, 512, 512]，这里的7代表7个前景类
    slot = torch.zeros([7, 3, 512, 512])
    # 初始化存储有效掩码的vm张量，形状为[7, 21, 512, 512]
    vm = torch.zeros([7, 21, 512, 512])
    # 初始化存储标签的sltarget张量
```

```
    sltarget = torch.zeros([7, 7])
    # 创建一个包含0到6的张量，代表类别索引，且放置在GPU上
    target2 = torch.linspace(0, 6, 7, dtype = torch.long).cuda()
    ems = torch.linspace(0, 6, 7, dtype = torch.long).cuda()


    def shuffle_batch(x, y, z, c):
        index = torch.randperm(x.size(0))
        x = x[index]
        y = y[index]
        z = z[index]
        c = c[index]

        return x, y, z, c

    for sam_feats, _, cls_ids, masks, class_embeddings in
       train_dataloader:

        # 获取当前批次的图像数据并放置在GPU上
        sam_feats = sam_feats.cuda()
        cls_ids = cls_ids.cuda(non_blocking = True)
        masks = masks.cuda()
        class_embeddings = class_embeddings.cuda()
        # 对图像、标签和有效掩码进行随机打乱
        sam_feats, cls_ids, masks, class_embeddings = shuffle_batch
           (sam_feats, cls_ids, masks, class_embeddings)
        # 获取当前批次的图像数量
        ba = len(cls_ids)
        for i in range(ba):
            # 找到标签中值为1的索引，即对应前景类别的索引
            argl = (cls_ids[i, :].squeeze() == 1).nonzero(as_tuple
               = False).squeeze()
            slot[argl] = sam_feats[i].cpu()
            vm[argl] = masks[i].cpu()
            sltarget[argl] = cls_ids[i].cpu()
            ems[argl] = class_embeddings[i].cpu()

        # 将标签重新放回GPU
        cls_ids = cls_ids.cuda()
        sam_feats = sam_feats.cuda()
```

```
43          masks = masks.cuda()
44          # 对存储的图像、类别索引和有效掩码进行随机打乱
45          slot_c, target2_c, vm_c = shuffle_batch(slot, target2, vm)
46          # 随机选择四个索引
47          first = random.randint(0, 19)
48          second = random.randint(0, 19)
49          third = random.randint(0, 19)
50          four = random.randint(0, 19)
51
52          # 将存储的图像、类别索引和有效掩码放回GPU
53          slot_c = slot.cuda()
54          target2_c = sltarget.cuda()
55          vm_c = vm.cuda()
56
57          # 将原始图像与从存储图像库中随机采样的图像在维度0上进行拼接
58          sam_feats = torch.cat([sam_feats, slot_c[first].unsqueeze
                (0), slot_c[second].unsqueeze(0), slot_c[third].
                unsqueeze(0), slot_c[four].unsqueeze(0)], dim = 0)
59          cls_ids = torch.cat([cls_ids, target2_c[first].unsqueeze(0)
                , target2_c[second].unsqueeze(0), target2_c[third].
                unsqueeze(0), target2_c[four].unsqueeze(0)], dim = 0)
60          class_embeddings = torch.cat([class_embeddings, vm_c[first
                ].unsqueeze(0), vm_c[second].unsqueeze(0), vm_c[third].
                unsqueeze(0), vm_c[four].unsqueeze(0)], dim = 0)
61
62          sam_feats = sam_feats.cuda()
63          cls_ids = cls_ids.cuda()
64          masks = masks.cuda()
65          class_embeddings = class_embeddings.cuda()
```

4.2 实验环境搭建

为了验证本文的有效性，我们在高性能计算平台上搭建了实验环境。实验的硬件配置包括：一台具有 NVIDIA P100 16GB GPU 的服务器，搭配 AMD EPYC 7742 64-Core Processor 和 512GB 内存。操作系统为 Ubuntu 20.04 LTS，提供了稳定的运行环境。

在软件配置方面，我们采用 Python 3.8 作为开发语言，核心深度学习框架选用 PyTorch 2.0，并结合 CUDA 11.7 和 cuDNN 8.5 以充分利用 GPU 的计算能力。同时，依赖的第三方库包括 Torchvision、NumPy、OpenCV 等，用于图像处理和数据加载。为了支持 SAM 的预训练模型加载，我们集成了官方提供的 SAM Python API，并在其基础上实现了 SurgicalSAM 框架。

4.3 创新点

与传统的 SAM 方法不同，SurgicalSAM 引入了一种基于轻量级类原型的类提示编码器，能够高效生成类别特定的提示嵌入，并与 SAM 预训练的特征知识进行深度融合，从而缓解手术领域与自然图像领域间的显著域间差异。本文为应对长尾分布问题对分类器和分割性能的影响，引入了 IBR 机制，通过建立图像库并均匀采样尾类数据，有效增加了模型对稀有类别的关注。

5 实验结果分析

虽然对训练策略进行了调整，但在本实验中，所选的改进方法未能显著提升模型性能。出现这个问题可能是由于当前的训练策略不适用于该任务的特点或数据集的性质。不同的任务和数据集往往有不同的挑战和需求，因此，训练策略的改进需要根据特定情况来量身定制。或者说，SAM 经过大量数据的预训练后，可能已经具备了较强的通用能力，对于尾部数据的处理已经达到了一定的稳定性或瓶颈。这表明其在广泛场景下的适应性可能较强，但在特定的尾部数据场景中，进一步提升性能的空间可能较为有限。

表 1. 结果对比

Method	IoU	mc IoU	Instrument Categories						
			BF	PF	LND	SI	CA	MCS	UP
SurgicalSAM	80.33	58.87	83.66	65.63	58.75	54.48	39.78	88.56	21.23
proposed	80.30	58.82	83.69	65.64	58.72	54.46	39.78	88.51	21.33

6 总结与展望

本文聚焦于手术器械分割任务，针对将 Segment Anything Model（SAM）应用于该任务时面临的域间隙和对精确提示的依赖问题，提出了 SurgicalSAM 方法。通过引入基于轻量级原型的类提示调优编码器与对比原型学习，SurgicalSAM 有效融合手术特定信息与 SAM 预训练知识，提升了模型泛化能力与类原型辨别力。在此基础上，本文进一步引入实例平衡重采样（IBR）方法，旨在通过增加尾类样本采样频率，校准分类器权重中的共享特征，以应对手术器械类别间数据分布不均衡问题。

然而，对 ndoVis2018 数据集的实验结果表明，尽管 SurgicalSAM 本身实现了先进性能，但加入 IBR 方法后整体效果并未产生明显变化。这可能源于当前训练策略与任务特点或数据集性质不匹配，不同任务和数据集对训练策略有特定要求，而现有策略未充分契合手术器械分割任务。此外，SAM 经大量数据预训练后，在处理尾部数据上或许已达到稳定性或瓶颈，虽具备广泛场景适应性，但在特定尾部数据场景下提升空间有限。

未来研究可考虑深入分析任务与数据集特性，定制更为适配的训练策略，以充分发挥 IBR 及相关方法在手术器械分割任务中的潜力。同时，探索如何突破 SAM 在处理尾部数据时可能存在的瓶颈，进一步挖掘其在特定领域的性能提升空间，有望推动手术器械分割技术的进一步发展。

参考文献

[1] Jiwoon Ahn, Sunghyun Cho, and Suha Kwak. Weakly supervised learning of instance segmentation with inter-pixel relations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2209–2218, 2019.

[2] Dosovitskiy Alexey. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv: 2010.11929*, 2020.

[3] Yude Wang, Jie Zhang, Meina Kan, Shiguang Shan, and Xilin Chen. Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12275–12284, 2020.

[4] Wenxi Yue, Jing Zhang, Kun Hu, Yong Xia, Jiebo Luo, and Zhiyong Wang. Surgicalsam: Efficient class promptable surgical instrument segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 6890–6898, 2024.

[5] Xinqiao Zhao, Feilong Tang, Xiaoyang Wang, and Jimin Xiao. Sfc: Shared feature calibration in weakly supervised semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7525–7533, 2024.

[6] Yanning Zhou, Hao Chen, Jiaqi Xu, Qi Dou, and Pheng-Ann Heng. Irnet: Instance relation network for overlapping cervical cell segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part I 22*, pages 640–648. Springer, 2019.