Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

摘要

本文提出一种创新的视觉 Transformer，即 Swin Transformer，该模型可作为计算机视觉的通用骨干网络。将 Transformer 从语言领域引入视觉领域面临着两个主要挑战：一是视觉实体的规模变化较大，二是与文本中的单词相比，图像中的像素具有更高的分辨率。为了解决这些差异，本文提出了一种分层 Transformer，通过移位窗口进行计算。移位窗口方案通过将自注意力计算限制在非重叠的局部窗口，同时允许跨窗口交换信息，从而带来了更高的效率。这种层次结构具有在各种规模上建模的灵活性，并且相对于图像大小具有线性计算复杂性。Swin Transformer 的这些特性使其与广泛的视觉任务兼容，包括图像分类和密集预测任务，如目标检测和语义分割。其性能大大超过了之前的最先进水平，展示了基于 Transformer 的模型作为视觉骨干的潜力。分层设计和移位窗口方法也被证明对全 MLP 架构有益。

关键词： Swin Transformer；移动窗口；SW-MSA

1 引言

长期以来，计算机视觉中的建模一直由卷积神经网络 (CNN) 主导。从 AlexNet [1] 及其在 ImageNet 图像分类挑战中的革命性表现开始，CNN 架构已经通过更大的规模 [2,3]，更广泛的连接 [4] 和更复杂的卷积形式 [5] 进化得越来越强大。随着 CNN 成为各种视觉任务的骨干网络，这些架构上的进步促进了性能的提高，广泛提升了整个领域的性能。

另一方面，与自然语言处理 (NLP) 中的网络架构不同，当今的流行架构是 Transformer [5]。Transformer 专为序列建模和转导任务设计，以其对数据中的长期依赖关系的关注而闻名。它在语言领域的巨大成功使得研究人员对它在计算机视觉的应用进行了研究，近来在某些任务上展示了良好的结果，特别是图像分类 [6] 和视觉-语言联合建模 [7]。

本文试图扩大 Transformer 的适用性，使其可以作为计算机视觉的通用骨干网络，就像它在 NLP 和 CNN 在视觉中的作用一样。将其在语言领域的高性能迁移到视觉域的重大挑战可以用两种模态之间的差异来解释。其中之一就是规模，与语言 Transformer 中作为处理基本元素的单词标记不同，视觉元素的规模可以有很大的变化，这是在目标检测等任务中受到关注的问题 [8]。在现有的基于 Transformer 的模型 [6] 中，token 都是固定规模的，这一特性不适合这些视觉应用。另一个区别是图像中的像素分辨率比文本中的单词高得多。存在许多视觉任务，如语义分割，需要在像素级别进行密集预测，而这对 Transformer 来说在高

分辨率图像上是难以处理的，因为其自注意力的计算复杂度是图像大小的二次方。为克服这些问题，本文提出一种通用的 Transformer 骨干 Swin Transformer 构建分层特征映射，对图像大小具有线性计算复杂度。如图 1(a) 所示，Swin Transformer 通过从小尺寸的块 (灰色轮廓) 开始，并逐渐在更深的 Transformer 层中合并邻近的块来构建分层表示。通过这些分层特征映射，Swin Transformer 模型可以容易地利用高级技术进行密集预测，如 feature pyramid networks(FPN) [8] 或 U-Net [9]。线性计算复杂度是通过在划分图像 (红色轮廓) 的非重叠窗口内局部计算自注意力来实现的。每个窗口中的图像块数量是固定的，因此复杂度与图像大小成线性关系。这些优点使 Swin Transformer 适合作为各种视觉任务的通用骨干，与之前基于 Transformer 的架构 [6] 相比，后者产生单分辨率的特征图，具有二次复杂度。

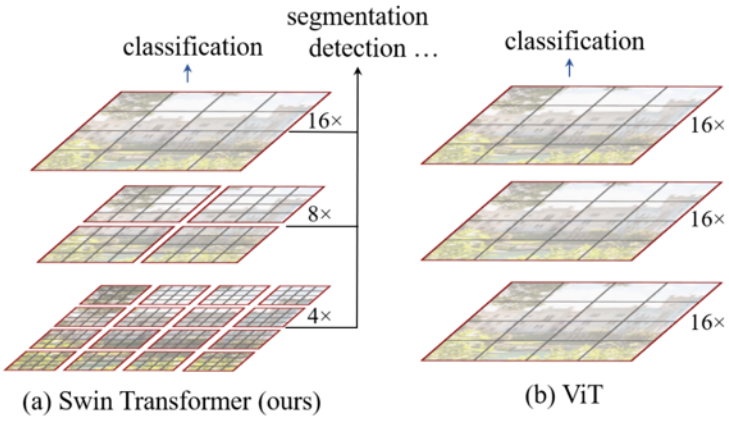

图 1. Swin Transformer vs ViT

Swin Transformer 设计的一个关键元素是它在连续自注意力层之间窗口划分的移动，如图 2所示。移位的窗口连接了前一层的窗口，提供了它们之间的连接，显著地增强了建模能力。这种策略对于实际的延迟也是有效的：窗口内的所有 query 块共享相同的 key，这有利于硬件中的内存访问。相比之下，早期的基于滑动窗口的自注意力方法在通用硬件上具有较低的延迟，这归因于不同 query 像素的关键集合不同。本文的实验表明，提出的移动窗口方法比滑动窗口方法具有更低的延迟，但在建模功率方面是相似的。移位窗口方法对于所有的 MLP 结构也是有益的。

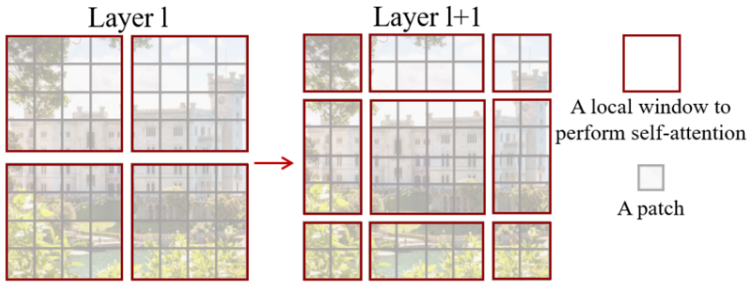

图 2. 移位窗口

2 相关工作

2.1 CNN 及其变体

当谈到计算机视觉领域时，CNN 无疑是不可忽视的关键技术。尽管 CNN 的概念已经存在了数十年，但其真正的崛起可以追溯到 AlexNet 的推出。AlexNet 在 ImageNet 大规模视觉识别竞赛中取得了引人注目的成功，这一突破标志着 CNN 的飞速发展并奠定了其在计算机视觉中的主导地位。

随着时间的推移，研究人员们提出了一系列更深、更高效的卷积神经网络架构，这进一步推动了深度学习在计算机视觉领域的浪潮。其中一些重要的架构包括 VGG [10], GoogleNet [11], ResNet [2], DenseNet [4], HRNet [12] 和 EfficientNet [13]。这些架构在不同方面进行了创新，如网络深度、模块设计和参数效率，为各种计算机视觉任务提供了强大的工具。

除了整体架构的进步之外，对单个卷积层的改进也成为研究的焦点。深度卷积和可变形卷积等技术在提高特征提取和模型表达能力方面发挥了关键作用，使得模型更具适应性和泛化能力。

尽管 CNN 及其各种变体在计算机视觉应用中扮演着主要角色，本文强调了类 Transformer 架构在视觉和语言之间建模的潜在威力。Transformer 架构因其在自然语言处理任务中的卓越表现而闻名，近年来，研究人员开始将其成功的思想引入计算机视觉领域，为实现视觉与语言的统一建模提供了新的可能性。

2.2 基于自注意力的骨干架构

受到自注意力层和 Transformer 架构在自然语言处理（NLP）领域取得成功的启发，一些研究开始尝试在计算机视觉中引入自注意力层，以替代流行的 ResNet [2] 等架构中的部分或全部空间卷积层。这一趋势的动机在于自注意力层的能力，它能够在每个像素的局部窗口内进行计算，从而加速优化过程。这些工作通过权衡精度和计算复杂度（FLOPs），相较于传统的 ResNet 等结构，实现了稍好的性能。

然而，它们昂贵的内存访问导致它们的实际延迟明显大于卷积网络。为了克服这一问题，本文提出在连续层之间采用移动窗口的方式，而非使用传统的滑动窗口。这一创新的设计允许在一般硬件中更有效地实现模型，从而降低了内存访问的开销。通过引入连续层之间的移动窗口，本文的方法在减小模型计算复杂度的同时保持了相对较高的精度。这种设计在概念上类似于传统卷积的感受野，但通过巧妙的窗口移动机制，有效地减少了计算和内存访问的负担。这不仅提高了模型的实用性，还使其更容易在各种硬件平台上部署和优化。

2.3 自注意力/transformer 补充 CNN

在对计算机视觉领域的不断深入研究中，另一项重要的工作集中在通过引入自注意力层或 Transformer 来增强标准 CNN 架构。这种方法的核心思想是利用自注意力的能力，提供对远程依赖关系或异构交互的编码，以有效地补充 CNN 主干网络 [14] 或特定任务的头网络 [15]。自注意力层的引入为模型提供了更广阔的上下文信息捕捉能力，有助于改进对复杂关系的建模。

不少研究将 Transformer 中的编码器-解码器设计成功应用于目标检测和实例分割任务[16]，这表明了 Transformer 架构在计算机视觉任务中的潜在应用价值。与这些工作不同，本文的重点在于深入探讨 Transformer 如何在基本视觉特征提取方面具有自适应性。本文的探索涉及到如何使 Transformer 在基础特征提取层面更灵活自适应。这一自适应性的设计使得模型能够更好地捕捉输入图像中的结构和关系，而不仅仅是针对特定任务的高级特征，同时也解决了多尺度的问题。这种方法的优势在于它提供了一种通用性的增强方式，为各种计算机视觉任务提供了更为灵活和强大的特征表示。

2.4 基于 Transformer 的视觉骨干

与本文的研究密切相关的是 Vision Transformer（ViT）及其后续工作。ViT 是一项开创性的工作，直接将 Transformer 架构应用于非重叠的中型图像块，以进行图像分类 [6]。相对于传统的卷积网络，ViT 在图像分类方面实现了引人注目的速度和精度的平衡。尽管 ViT 在表现上需要大规模的训练数据集，但 DeiT 引入了多种训练策略，使 ViT 能够有效地利用较小的 ImageNet-1K 数据集 [17]。虽然 ViT 在图像分类任务上表现出色，但由于其低分辨率特征图和复杂度随图像大小的二次增长，不太适合用作密集视觉任务或输入图像分辨率高的通用骨干网络。

在应用 ViT 于密集视觉任务的尝试中，有一些研究通过直接上采样或反卷积的方式将 ViT 模型应用于目标检测和语义分割，但其性能相对较低 [18]。同时，一些研究对 ViT 架构进行了修改以改善图像分类性能 [19]。在这其中，Swin Transformer 架构突出了通用性能，在图像分类任务中取得了最佳的速度和精度平衡。此外，还有并行研究探索了在 Transformer 上构建多分辨率特征图的类似思路 [20]，但其复杂度仍然是图像大小的二次型。相比之下，本文的算法是线性的，并且采用了局部操作，这在建模视觉信号的高度相关性方面被证明是有益的 [21]。本文的方法不仅在效率和性能上超越了传统 ViT 的应用范围，而且在密集视觉任务上实现了优越的性能，如在 COCO 目标检测和 ADE20K 语义分割任务中取得了最先进的精度。

3 本文方法

3.1 整体架构

Swin Transformer 架构的概述见图3，这是一个微型版本 (Swin-T)。它首先通过块分割模块将输入的 RGB 图像分割为不重叠的块。每个 patch 被视为一个'token'，其特征被设置为原始像素 RGB 值的拼接。在本文的实验中，使用 4×4 的 patch 大小，因此每个 patch 的特征维度是 $4\times4\times3 = 48$。在这个原始值特征上应用线性嵌入层，将其投影到任意维度 (记为 C)。

在这些 patchs 上应用了几个具有改进的自注意力计算的块 (Swin Transformer 块)。Transformer 块维护 token 的数量 ($\frac{H}{4} \times \frac{W}{4}$)，并与线性嵌入一起称为 “第一阶段”。

为了产生分层表示，随着网络的加深，通过 Patch Merging 层来减少 token 的数量。第一个块 Merging 层连接每组 2×2 相邻块的特征，并在连接后的 $4C$ 维特征上应用线性层。这将标记的数量减少了 $2 \times 2 = 4$ 的倍数 (2× 分辨率下采样)，并将输出维度设置为 $2C$ 。之后应用 SwinTransformer 块进行特征变换，分辨率保持在 $\frac{H}{8} \times \frac{W}{8}$。第一个 Patch Merging 和特

征变换的块被称为'Stage 2'。该过程重复两次，分别为“阶段 3”和“阶段 4”，输出分辨率分别为 $\frac{H}{16} \times \frac{W}{16}$ 和 $\frac{H}{32} \times \frac{W}{32}$。这些阶段共同产生一个分层表示，具有与典型卷积网络相同的特征图分辨率，例如 VGG [10] 和 ResNet [2]。所提出的架构可以方便地替换现有方法中的骨干网络，用于各种视觉任务。

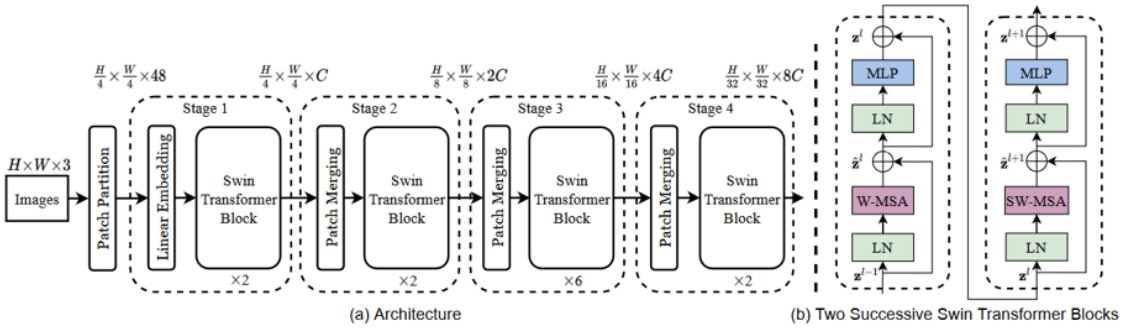

图 3. Swin Transformer 架构

Swin Transformer Block 是通过将 Transformer 块中的标准多头自注意力 (MSA) 模块替换为基于移位窗口的模块来构建的，其他层保持不变。如图 3(b) 所示，SwinTransformer 块由基于移位窗口的 MSA 模块组成，其次是中间具有 GELU 非线性的 2 层 MLP。在每个 MSA 模块和每个 MLP 模块之前应用 LayerNorm (LN) 层，并在每个模块之后应用残差连接。

3.2 基于移位窗口的自注意力机制

标准 Transformer 架构 [5] 及其在图像分类任务中的自适应应用 [6] 通常采用全局自注意力机制，这意味着它在计算标记（tokens）与所有其他标记之间的关系时涵盖整个输入序列。尽管全局自注意力在自然语言处理任务中表现出色，但对于许多视觉问题，特别是需要处理大量标记或高分辨率图像的密集预测任务，它的二次复杂性可能导致计算资源的巨大开销。全局自注意力的二次复杂性是因为在处理每个标记时，都需要与序列中的所有其他标记进行交互。这对于大型数据集或高分辨率图像来说可能是不切实际的，因为计算的时间和内存需求会迅速增加。

本文的方法通过引入基于移位窗口的注意力机制，部分弥补了全局自注意力的复杂性，使得处理大规模标记集或高分辨率图像的问题变得更为可行。这一创新不仅提高了效率，而且在密集预测和处理高分辨率图像方面取得了显著的性能提升。

3.2.1 非重叠窗口中的自注意力

为了更高效地进行建模，本文提出在局部窗口内进行自注意力计算。通过以不重叠的方式排列这些窗口并均匀地分割图像，可以实现对图像局部区域的更有针对性的关注。假设每个窗口包含 $M \times M$ 个 patch，全局多头自注意力模块（MSA）和基于 $h \times w$ 大小的 patch 图像窗口的计算复杂性分别为：

$$\Omega(MSA) = 4hwC^2 + 2(hw)^2C \tag{1}$$

$$\Omega(W\text{-}MSA) = 4hwC^2 + 2M^2hwC \tag{2}$$

其中 $\Omega(MSA)$ 表示全局 MSA 模块的计算复杂性，其结果是 hw（图像宽度和高度的乘积）的二次函数。相反，$\Omega(W\text{–}MSA)$ 表示基于窗口的自注意力模块的计算复杂性，其结果是当窗口大小 M（默认设置为 7）时的线性函数。

上述公式揭示了全局自注意力计算对于较大的 hw 通常是难以承受的，因为其复杂性呈现二次增长。相比之下，基于窗口的自注意力是可扩展的，其复杂性随着窗口大小的增加而线性增长。这一建议使得 Transformer 能够更灵活地适应不同规模和分辨率的图像，提高模型的可扩展性，特别是在处理大型图像或大规模标记集的情况下。因此，本文的方法通过引入基于窗口的自注意力机制，有效地降低了计算复杂性，为处理复杂的视觉任务提供了更可行的解决方案。

3.2.2 连续分块的移位窗口

虽然基于窗口的自注意力模块在每个窗口内部有效地捕捉局部信息，但其缺乏跨窗口的信息交互，这在一定程度上限制了其建模能力。为了引入跨窗口的连接并保持在非重叠窗口的情况下实现高效计算，本文引入了一种创新的移位窗口划分方法，并将其应用于连续的 Swin Transformer 块中。

如图 2所示，第一个模块采用从左上角像素开始的传统窗口划分策略，将 8×8 的特征图均匀划分为大小为 $4\times 4(M=4)$ 的 2×2 窗口。接下来的模块则采用了相对于前一层窗口配置的偏移，通过将窗口从规则划分的窗口移动 $(\lfloor \frac{M}{2} \rfloor, \lfloor \frac{M}{2} \rfloor)$ 像素。

使用移位窗口划分方法，连续的 Swin Transformer 块的计算过程为：

$$\begin{aligned} \hat{z}^{l} &= W\text{–}MSA(LN(z^{l-1})) + z^{l-1}, \\ z^{l} &= W\text{–}MLP(LN(z^{l})) + z^{l}, \\ \hat{z}^{l+1} &= SW\text{–}MSA(LN(z^{l})) + z^{l}, \\ z^{l+1} &= W\text{–}MLP(LN(z^{l+1})) + z^{l+1}, \end{aligned} \tag{3}$$

其中 $\hat{z}^{l}$ 和 z^{l} 分别表示块 1 的 (S)W-MSA 模块和 MLP 模块的输出特征；W-MSA 和 SW-MSA 则分别使用常规和移位的窗口分区配置表示基于窗口的多头自注意。

移位窗口划分方法的引入有效地增加了前一层相邻非重叠窗口之间的联系，这对于图像分类、目标检测和语义分割等任务带来了显著的性能提升。

3.2.3 移位配置的高效批处理计算

移动窗口分区的一个问题是，它将导致更多的窗口，在移动的配置中，从 $\lceil \frac{h}{M} \rceil \times \lceil \frac{w}{M} \rceil$ 增加到 $(\lceil \frac{h}{M} \rceil + 1) \times (\lceil \frac{w}{M} \rceil + 1)$，并且其中一些窗口可能小于 $M \times M$。为了解决这个问题，一个朴素的方法是将较小的窗口填充到 $M \times M$ 的大小，然后在计算注意力时屏蔽填充值。然而，这种朴素的解决方案可能会在常规分区的窗口数量相对较小时（例如 2×2），引入相当大的额外计算量（从 2×2 增加到 3×3 相当于增加了 2.25 倍）。在这里，本文引入了一种更为高效的批处理计算方法，即向左上方向循环移位，如图 4所示。通过这种循环移位的转换，批处理窗口可能由特征映射中不相邻的几个子窗口组成。为了确保计算注意力时限制在每个子窗口内，采用了屏蔽机制。值得注意的是，通过循环移位，批处理窗口的数量与常规窗口分区的数量保持一致，因此仍然保持了高效计算的优势。这种批处理计算方法在有效处理移动窗

口分区时，既减少了冗余计算，又保持了计算的高效性。这一改进不仅使得模型更加灵活适应各种窗口配置，而且在保持计算效率的同时提升了整体性能。

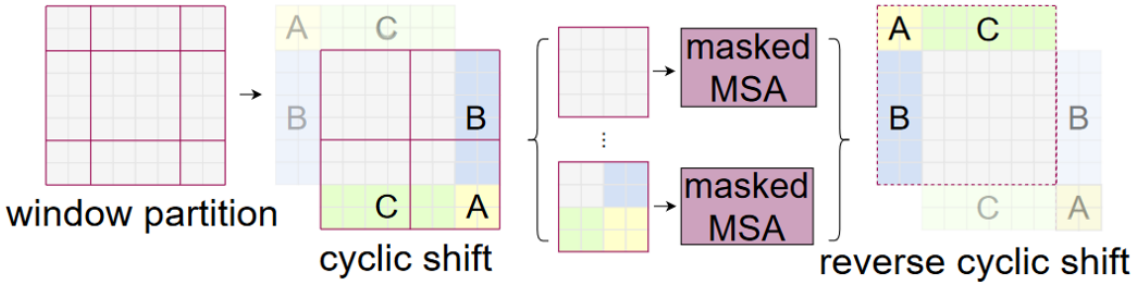

图 4. 循环移位

3.2.4 相对位置偏差

在计算自注意力时，本文遵循 [2]，在计算相似度时为每个头部加入一个相对位置偏差 $B \in \mathbb{R}^{M^2 \times M^2}$：

$$Attention(Q, K, V) = SoftMax(QK^T/\sqrt{d} + B)V \tag{4}$$

其中 $Q, K, V \in \mathbb{R}^{M^2 \times d}$ 为 $query$，key 和 $value$；d 为 $query/key$ 维数，M^2 为窗口 patch 数。由于沿每个轴的相对位置在 $[-M+1, M-1]$ 范围内，参数化了一个较小尺寸的偏置矩阵 $\widehat{B} \in \mathbb{R}^{(2M-1)\times(2M-1)}$，并且 B 中的值取自 $\widehat{B}$。实验观察到，与没有此偏差项或使用绝对位置嵌入的对应项相比，有显著的改进。进一步向输入中添加绝对位置嵌入 (如 [6]) 会略微降低性能，因此在本文没有采用它。预训练中学习到的相对位置偏差也可以用来初始化一个模型，通过双三次插值进行不同窗口大小的微调 [6]。

3.3 体系结构变体

本文建立了基础模型，称为 Swin-B，具有类似于 ViT-B/DeiT-B 的模型大小和计算复杂度。本文还介绍了 Swin-T、Swin-S 和 Swin-L，它们分别是模型大小和计算复杂度分别为 $0.25\times, 0.5\times$ 和 $2\times$ 的版本。需要注意的是，Swin-T 和 Swin-S 的复杂度分别与 ResNet-50 (DeiT-S) 和 ResNet-101 相似。窗口大小默认设置为 $M = 7$。对于所有实验，每个头部的 query 维数为 $d = 32$，每个 MLP 的扩展层为 $\alpha = 4$。这些模型变体的架构超参数如下:

- Swin-T: $C = 96$，层数 $= \{2,2,6,2\}$
- Swin-S: $C = 96$，层数 $= \{2,2,18,2\}$
- Swin-B: $C = 128$，层数 $= \{2,2,18,2\}$
- Swin-L: $C = 192$，层数 $= \{2,2,18,2\}$

其中 C 为第一阶段隐藏层的通道数。

4 复现细节

4.1 与已有开源代码对比

在复现过程中，我认真研究了原论文提供的开源代码，并将其作为参考。主要参考了原论文源码中的 models/swin_transformer.py 文件，该文件实现了 Swin Transformer 的整体框架。在使用这一部分代码时，我进行了适当的修改和优化，以适应多模态输入的需求。具体而言，我的工作主要包括以下几个方面：我对代码进行了修改，以确保其能够处理多模态输入。这包括调整输入处理部分，以兼容不同类型的数据。为了增强代码的可读性，我添加了详细的注释。这有助于理解代码的功能和实现细节，也方便其他人更容易使用和理解这一部分的实现。鉴于我的任务需求，我对一些超参数进行了调整，以适应我的实验设置。这确保了代码在新场景下的适用性。

1. **SwinTransformerBlock 模块：**

```
# 就是论文中（b）图的顺序：LN——W-MSA/SW-MSA——Dropout——(+)——LN——MLP——Dropout———(+)

        self.norm1 = norm_layer(dim) # 第一个 Layer Norm
        # WindowAttention就是W-MSA或SW-MSA，若传入的 shift_size = 0则使用W-MSA，否则使用SW-MSA
        self.attn = WindowAttention(
            dim, window_size=(self.window_size, self.window_size),
            num_heads=num_heads, qkv_bias=qkv_bias,
            attn_drop=attn_drop, proj_drop=drop)

        # Dropout/DropPath
        self.drop_path = DropPath(drop_path) if drop_path > 0. else nn.Identity()
        # 第二个LN层
        self.norm2 = norm_layer(dim)
        # 计算MLP隐层的 dimension维度，输入层的 MLPratio*输入的 dim
        mlp_hidden_dim = int(dim * mlp_ratio)
        # 就像是NN里面，输入层神经元个数、隐藏层神经元个数、输出层神经元个数，
        # act_layer为激活函数，drop决定 dropout率
        self.mlp = Mlp(in_features=dim, hidden_features=mlp_hidden_dim,
                       act_layer=act_layer, drop=drop)
```

2. **PatchMerging 模块：**

```
        """
        x: B, H*W, C
        """
        B, L, C = x.shape
        assert L == H * W, "input feature has wrong size"

        x = x.view(B, H, W, C)

        # padding
        # 如果输入feature map的H, W不是2的整数倍，需要进行padding
        pad_input = (H % 2 == 1) or (W % 2 == 1)
        if pad_input:
            # to pad the last 3 dimensions, starting from the last dimension
            # and moving forward.
            # (C_front, C_back, W_left, W_right, H_top, H_bottom)
            # 这里的Tensor通道是[B, H, W, C]，和官方文档有些不同
            x = F.pad(x, (0, 0, 0, W % 2, 0, H % 2))

        x0 = x[:, 0::2, 0::2, :]  # [B, H/2, W/2, C]
```

```
        x1 = x[:, 1::2, 0::2, :]  # [B, H/2, W/2, C]
        x2 = x[:, 0::2, 1::2, :]  # [B, H/2, W/2, C]
        x3 = x[:, 1::2, 1::2, :]  # [B, H/2, W/2, C]
        # [B, H/2, W/2, 4*C], -1表示在最后一个维度上进行拼接
        x = torch.cat([x0, x1, x2, x3], -1)
        x = x.view(B, -1, 4 * C)  # [B, H/2*W/2, 4*C]

        x = self.norm(x)
        x = self.reduction(x)  # [B, H/2*W/2, 2*C]
```

4.2 实验环境搭建

- **硬件配置**
  - 计算机: wilkes-PT6620P
  - CPU: Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz
  - 内存: 251 GB
  - GPU: Quadro RTX 8000 * 2
- **软件配置**
  - 操作系统: Ubuntu 20.04
  - Python: Python 3.7.16
  - 深度学习框架: torch 1.8.0, tensorflow 2.14.0
- **数据集和样本**
  - 数据集来源: CIFAR-100, flowers 数据集
  - 数据预处理: 图像加载、图像大小调整和数据标准化等
- **硬件和软件设置**
  - 学习率: 0.0001
  - 模型架构: Swin-Transformer
- **实验执行**
  - 数据输入: 训练数据集包含 50000 个样本，测试数据集包含 10000 个样本
  - 算法执行: 使用 Python 脚本执行训练和评估

4.3 创新点

集成 Grad-CAM：为了提高 Swin Transformer 的可解释性，引入了 Grad-CAM（梯度类激活映射）。这一技术使得能够可视化模型在图像中关注的区域，从而更清晰地理解模型的决策过程。Grad-CAM 的整合使得模型的输出更具可解释性，为深度学习模型的应用提供更大的信任度和可靠性。

4.4 实验设计

4.4.1 数据集选择：

CIFAR-100：

CIFAR-100 是一个包含 60000 张 32x32 像素的彩色图像的数据集（如图5所示），其中包含 100 个不同的类别，每个类别有 600 张图像。这 100 个类别被分为 20 个大类，每个大类包含 5 个小类。每张图像有三个通道（红、绿、蓝），图像尺寸相对较小，为 32x32 像素。数据集涵盖了广泛的主题，包括动物、植物、交通工具、日常物品等，使其适用于多样性和复杂性的图像分类任务。数据集划分为训练集（50000 张图像）和测试集（10000 张图像）。

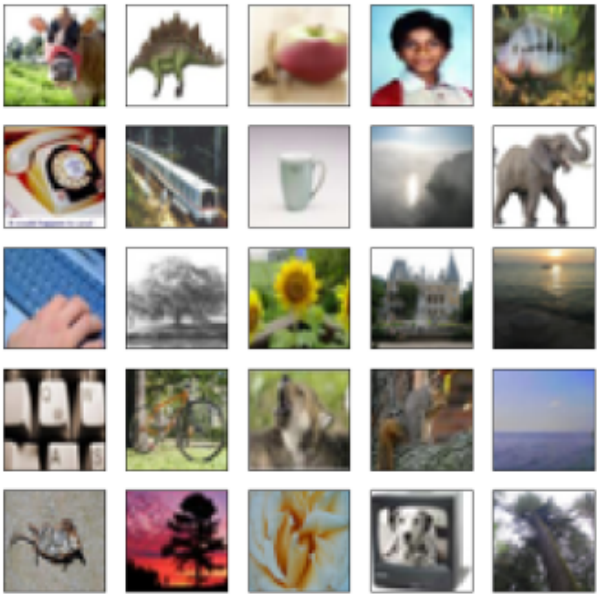

图 5. CIFAR-100 数据集部分图片

因为其包含了 100 个类别，覆盖了丰富的物体和场景。这使得训练模型需要具备对多样性和复杂性的理解和泛化能力。32x32 像素的小尺寸图像使得模型需要关注图像的局部细节，从而增加了任务的难度。这也使得在计算资源有限的情况下更容易进行实验和快速迭代。训练模型在 CIFAR-100 上可以帮助验证其在更大和更复杂数据集上的性能。由于图像相对较小，模型的训练和评估速度较快，这有助于在短时间内进行快速原型开发和实验。

flowers 数据集：

这是一个花分类数据集，其中包含 3670 个样本，包含 5 个不同的类别（如图6所示）：分别为 daisy（雏菊）、dandelion（蒲公英）、roses（玫瑰）、sunflowers（向日葵）和 tulips（郁金香）。数据集划分为训练集（3306 张图像）和测试集（364 张图像）。

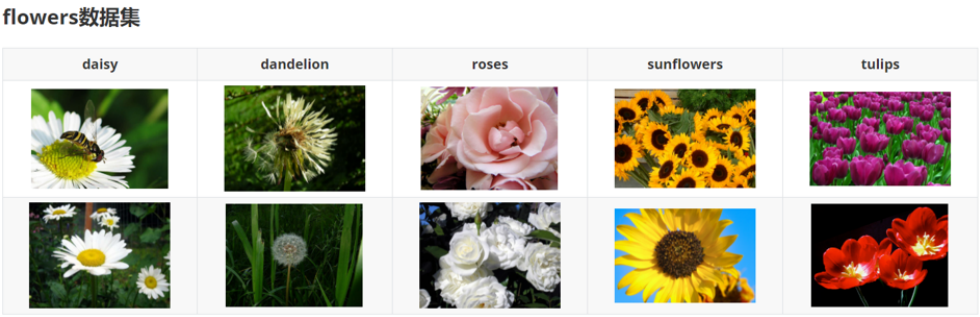

图 6. flower 数据集部分图片

4.4.2 复现过程:

模型训练: 首先搭建适宜的开发环境，包含相关深度学习库。其次进行数据集的选择和预处理，确保数据与模型需求相匹配。随后初始化模型，设置合理的初始参数和加载预训练权重。在训练阶段，配置训练参数并开始训练模型，同时监控模型性能指标以优化训练过程。具体的流程如图 7所示。

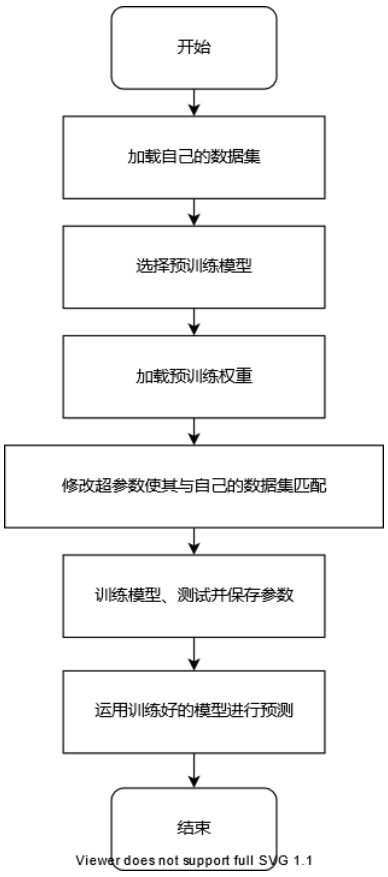

图 7. 训练过程

解释过程: 首先，选择合适的数据集并进行必要的预处理。搭建模型架构，并对其进行必要的配置，这包括设置 Layer Norm 层和其他关键结构。然后进行模型的训练和验证，期间调整超参数以优化性能。训练完成后，使用 Grad-CAM 可视化技术来理解和解释模型是如何对输入数据做出决策。具体的流程如图 8所示。

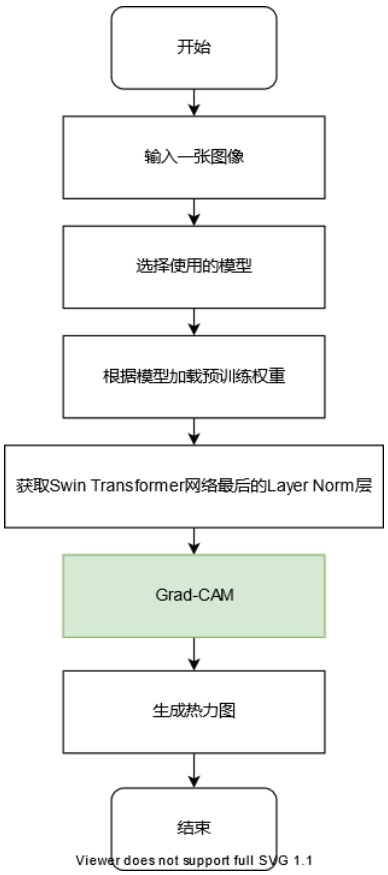

图 8. 解释过程

5 实验结果分析

1. **在 CIFAR-100 上训练 Swin Transformer 模型**

   Swin Transformer 模型在 CIFAR-100 数据集上经历了 150 个训练周期，模型在验证集上的准确率和前五准确率有稳定的提升，最终准确率约为 45% 左右，前五准确率约为 75% 左右。

   训练损失和验证损失随训练周期变化的情况。从图中可以看出，随着周期数的增加，训练损失和验证损失都在逐渐下降，显示出模型在学习过程中的收敛性。最后，模型在测试集上的性能被评估，测试损失（Test loss）为 2.5975，测试准确率（Test accuracy）为 46.04%，测试前五准确率为 76.03%。这表明模型在未见数据上有着相对稳定的预测能力。

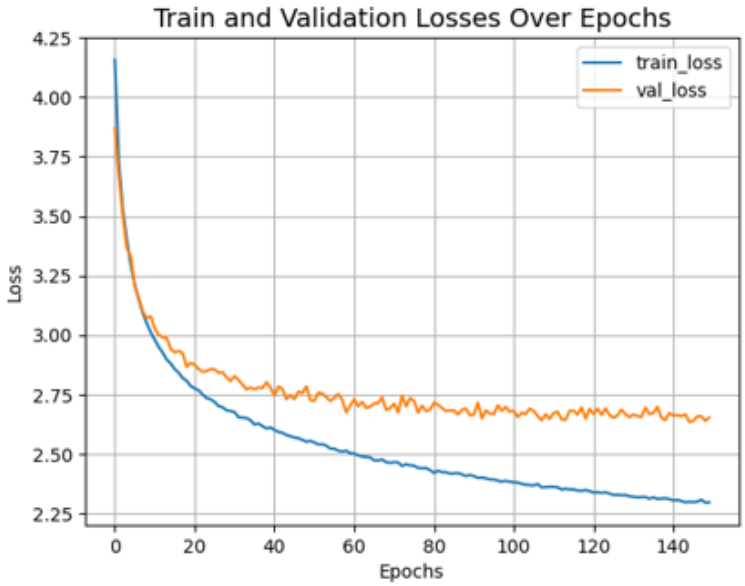

图 9. 训练 150 个周期

2. **使用预训练模型在 flower 数据集进行微调**

将模型最后一层（分类头部）的输出通道数从预训练权重对应的类别数修改为 5，以匹配 flower 数据集的类别数。使用 flower 数据集进行微调，调整学习率较小，因为只需要微调模型以适应新的数据集，而不是从头开始学习。微调完成后，保存模型的权重。

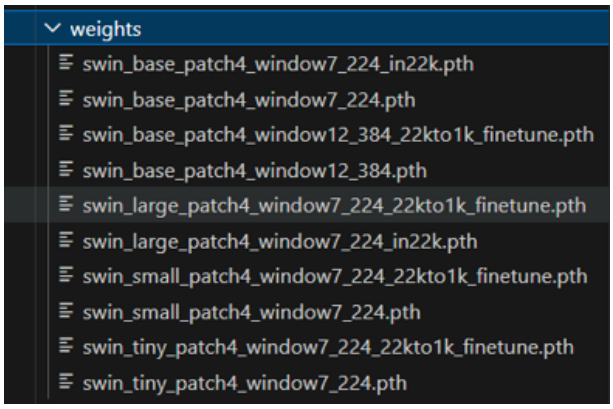

图 10. 微调后保存的模型

3. **在 Swin Transformer 中添加 Grad-CAM 使其结果可解释**

图11展示了在不同的 Swin Transformer 模型变种上应用 Grad-CAM 可视化技术的结果。Grad-CAM 用于突出影响神经网络决策的图像区域，即模型预测某一类别时最为关注的部分。图中每一行代表一个类别（例如猫、狗、玫瑰、郁金香），每一列代表一个不同的模型变种或训练设置。热图显示了模型对每个类别的注意焦点。例如，对于猫和狗的类别，热点集中在动物的面部区域；而对于玫瑰和郁金香，则集中在花的形状和花瓣上。这些热图能够帮助我们理解模型的预测行为，并验证模型是否在正确的区域内学习特征。

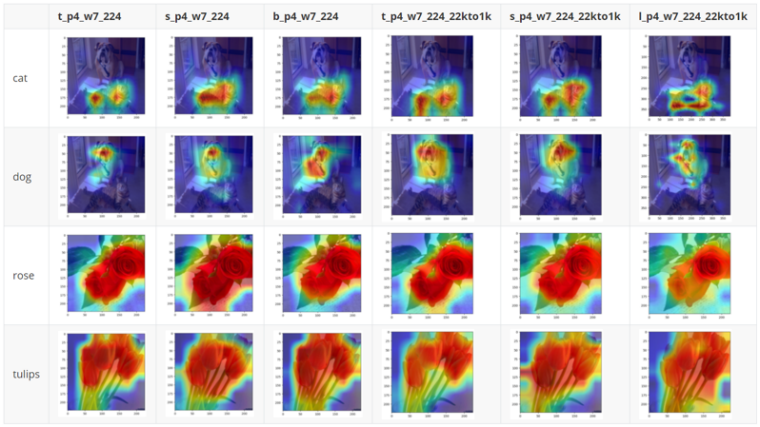

图 11. 不同预训练模型比较

6 总结与展望

本文详尽地阐述了 Swin Transformer 这一创新的计算机视觉变换器方法，充当多任务视觉处理的通用网络骨架。本文探讨了处理图像尺度多样性和高分辨率问题的新策略，并提出了一种基于移动窗口的层次化变换器架构。该方法与现有的视觉任务兼容，并在性能上优于前沿模型。本文详细描述了模型架构、实验设置及优化流程，我在模型中集成 Grad-CAM 以提升模型可解释性和针对多模态输入的调整。结果表明，该模型在图像分类、目标检测和语义分割等方面表现出色。对于未来的研究方向，可以提升变换器的适应性、架构效率的优化，以及模型可解释性的深入研究。

参考文献

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[2] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3464–3473, 2019.

[3] Xiyang Dai, Yinpeng Chen, Jianwei Yang, Pengchuan Zhang, Lu Yuan, and Lei Zhang. Dynamic detr: End-to-end object detection with dynamic attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2988–2997, 2021.

[4] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.

[5] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. In

Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 2736–2746, 2022.

[6] Hao-Shu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 682–691, 2019.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[8] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10213–10224, 2021.

[9] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16519–16529, 2021.

[10] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14454–14463, 2021.

[11] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[12] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pages 191–207. Springer, 2020.

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[14] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 173–190. Springer, 2020.

[15] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

[16] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[17] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European conference on computer vision (ECCV)*, pages 418–434, 2018.

[18] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection. *arXiv preprint arXiv:2012.09958*, 2020.

[19] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10076–10085, 2020.

[20] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zi-Hang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 558–567, 2021.

[21] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.