PWLT：基于金字塔窗口的图像分类轻量级变换器

摘要：近年来，视觉变换器（ViTs，vision Transformers）在图像分类领域上已经取得了显著的成就，由于采用自注意力机制的 ViTs 计算成本与输入标签的数量呈二次方关系，一些学者提出基于窗口的 ViTs 方法来解决这个问题。然而，这些方法将自注意力机制的计算量限制到一个有限空间的局部窗口内，因此丧失了编码图像全局交互的能力。此外，使用固定大小的窗口总是会受到单尺度表示的限制，这不适用于可变尺度的目标识别。为了解决上述问题，本文提出了一种基于金字塔窗口的轻量级变换器 PWLT，用于图像分类。具体来说，为了解决对多尺度信息的需求，我们采用不同大小的窗口来对不同尺度的对象进行编码。为了恢复在不同窗口之间的关系并研究全局上下文，我们引入了一个双重自注意力机制，用局部到全局的注意力来重建这些关系。我们在 ImageNet-1K 和 CIFAR100 数据集上进行了大量实验来证明 PWLT 在图像分类上的有效性。

关键词：图像分类，轻量级视觉变换器，金字塔窗口，自注意力。

1. 引言

在过去十年间，卷积神经网络（CNNs，Convolutional Neural Networks）在计算机视觉领域十分重要。一系列杰出的网络架构，比如 VGG[1], ResNet[2]and Efficient Net[3],已经有力地证明了 CNNs 的优越性。然而，传统的 CNNs 主要侧重于用卷积核编码局部特征，忽视了捕捉整个图像的全局信息。相比之下，最近 Transformers[4][5][6]在用多头自注意力机制(MHSA,multi-head self-attention mechanism)建立特征之间的长程依赖方面取得了显著的成功。这些方法已经在图像分类[7]、目标检测[8]和语义分割[9]等任务方面展现出卓越的效果。如图 1(a)所示，ViT[5]首先为图像分类设计了基于 Transformer 的主干网络，并且已经取得了显著成效。然而，由于 MHSA 的计算成本与输入标签数量的平方成正比，这限制了基于 Transformer 的主干网络在图像分类[7]和下游任务[8][9]中的广泛应用。目前已有学者提出几种基于窗口的 ViTs 来解决这个问题，如图 1(b)中所示的 Swin[10]。然而，这种方法[6][10][11]在两个方面具有局限性。首先，它们在捕捉整个场景的全局上下文信息方面存在不足，这是由于它们主要侧重于用预定义的窗口来捕捉局部信息。其次，考虑到图像中通常存在目标的尺度变换，仅依赖于单尺度窗口不足以提取多尺度特征。

为了解决上述这些问题，本文设计了一个新颖的 Transformer 主干网络，即基于金字塔窗口的轻量变换器（PWLT），用于图像分类。如图 1(c)所示，PWLT 在窗口之间建立互连，并且用一个金字塔窗口来有效地在多尺度上编码信息。具体而言，针对第一个问题，我们提出了一个双重自注意力机制(DSA,Dual Self-Attention)，其主要包含两个步骤：窗口自注意力（WSA，Window-based Self-Attention）和池化自注意力（PSA,Pooling-based Self-Attention)。WSA 通过将注意力限制在窗口内，来有效地在局部区域建立关系，然而，正如前文所言，这种方法不能捕捉全局上下文。PSA 则是用窗口的标签池化来建立这些区域之间的关系，由于窗口的数量较少，因此仅需要相对较少的计算量。针对第二个问题，我们提出在不同的注意力头组使用不同大小的窗口来捕捉特定区域的特征。与引入并行路径[12][13]不同，我们的方法简化了网络架构。简而言之，本文的主要贡献主要体现在以下三个方面：

- 设计了一个新型 DSA 模块,包含两个部分：窗口自注意力 WSA 和池化自注意力 PSA,用于捕捉全局上下文，且计算成本较低。WSA 和 PSA 都模仿了自注意力机制,其中 WSA 将 MHSA 的计算量限制到划分的窗口内,而 PSA 则重建了窗口连接以捕捉基于图像的全局交互。
- PWLT 使用了一个金字塔窗口从多尺度而不是单尺度特征[10][11]来编码特征表示。此外，与采用具有复杂网络架构的多路径 ViT[12][13]不同,PWLT 在每个头组构建金

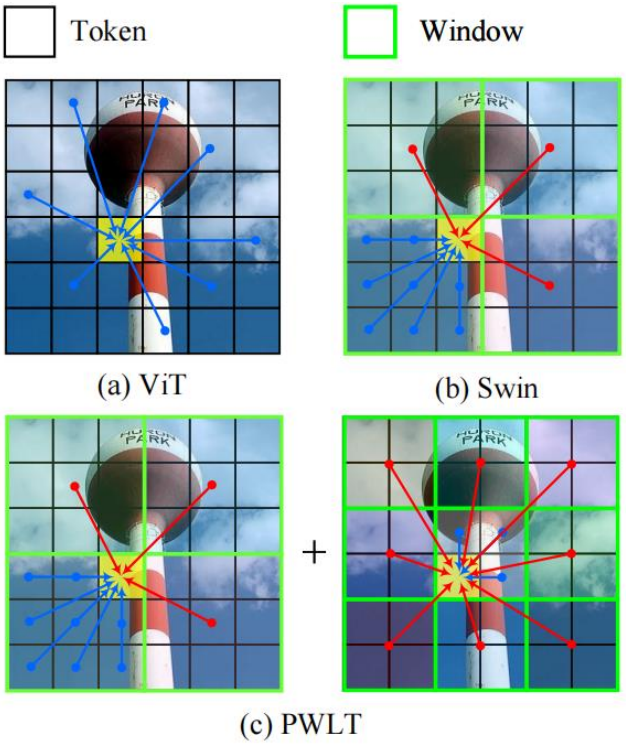

图 1 ViT[5]、Swin[10]和 PWLT 的对比。蓝色和红色箭头分别表示源自标签和窗口的信息。(a)ViT 计算所有标签之间的相似度，以编码全局交互。(为了视觉效果，图中只画了几条指向黄色标签的箭头。然而，实际上，每个标签都与黄色标签相关联。) (b) Swin 将注意力计算限制在局部窗口内，以降低计算复杂度，并采用移位窗口策略来建立全局上下文。(c) PWLT 不仅建立所有标签之间的关系，还利用窗口金字塔来编码多尺度依赖关系。(彩色显示效果最佳)

字塔窗口，从而显著简化了网络架构。

- 我们在两个广泛应用的图像分类数据库：ImageNet-1K[7]和 CIFAR100[14]上评估了 PWLT。实验表明，我们的方法在检测精度与实现效率间达到了最优的平衡。具体而言，PWLT 在 ImageNet-1K[7]上取得了 79.2%/82.1%/83.2% 的 Top-1 准确率，其模型大小仅为 12.2/29.8/52.2M，计算量分别为 1.8/4.9/9.3 GFLOPs。在 CIFAR100[14]上取得了 78.4%的 Top-1 准确率，模型大小仅为 11.8M，计算量为 0.5GFLOPs。

2. 相关工作

2.1. 基于窗口的 Transformers

为了减少 Transformers 的计算成本，研究人员已经提出很多基于窗口的 Transformer 主干网络。这些方法一般将 MHSA 的计算量限制在预定义的窗口内，避免对所有特征标签进行全局计算，从而有效减轻了计算负担。Swin[10]先均匀地将图像划分为不重叠的窗口，并在每个独立的窗口内限制 MHSA 的计算，使 MHSA 的计算复杂度变为线性的。然而，这依赖于移动窗口来恢复全局交互，导致了复杂的窗口掩码操作。SepViT[6]也将注意力计算限制在窗口内，但是它用窗口标签来重建窗口连接，在某种程度上克服了采用移动窗口的不足。VSA[11]采用了一个窗口回归模型来预测目标窗口的大小和位置，该模型可以适应不同大小的目标。Shuffle Transformer[15]是一个卷积神经网络-Transformer 混合模型，采用深度卷积[16]来补充空间混洗，以增强相邻窗口之间的连接。NAT[17]将每个查询的 MHSA 计算限制在最近的相邻标签上，这本质上等同于基于窗口的 Transformer。DaViT [18]利用 WSA 和通道自注意力来捕捉全局上下文，同时保持较快的运行速度。Couplformer[19]提出将注意力图解耦为子矩阵，并根据空间信息生成对其分数，以提高时间和内存效率。

与仅研究局部关系或采用复杂操作（比如移动窗口）来编码全局上下文的方法不同，我们采用基于池化的标签来研究窗口交互。SepViT[6]中使用的标签相似与我们的 PSA，但是它先被随机初始化，然后通过训练过程进行校正。此外，PWLT 并非仅依赖于单尺度的窗口表示，而是生成一个金字塔窗口，从而能够自适应的提取多尺度信息。

2.2. Transformer 中的多尺度信息

多尺度信息已经被成功地应用于 Transformers [13][20][21]。例如，Shunted Transformer[21]通过合并标签来表示更大的目标特征，同时保留某些标签来保持细粒度特征，这一新颖方案使自注意力机制能够学习不同尺度目标之间的关系。P2T[20]使用融合自适

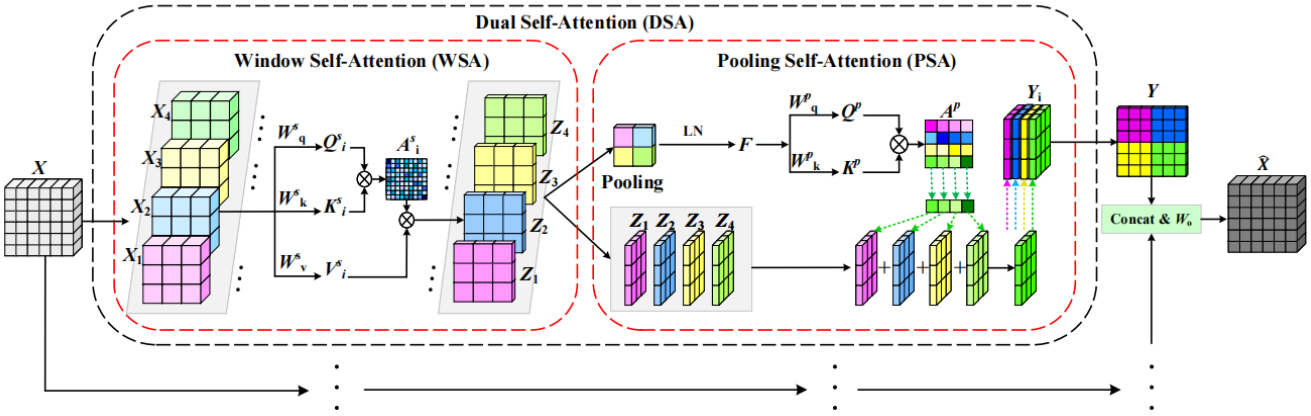

图 2 DSA 的示意图。DSA 由两个组件组成： WSA 和 PSA。注意，DSA 是在头组中执行的，其中输入特征可以被划分为不同数量的窗口，从而形成 PWLT 的金字塔窗口架构。LN 表示层归一化。（彩色显示效果最佳）

应金字塔池化的 MHSA，以捕捉强大的上下文特征。MPViT[12]通过重叠卷积同时嵌入不同大小的补丁特征，从而能在同一层次上实现细粒度特征和粗粒度特征的表示。融合多尺度信息的另一种方法是使用图像金字塔[13][22]。例如，MMViT[13]从多个视角并行地对不同分辨率的输入进行编码。CrossViT [22]设计了一个双分支 Transformer 主干网络，通过组合不同大小的图像块来生成更强大的特征。

尽管这些方法已经取得了较好的分类效果，但由于先进的 Transformers 采用了多路径架构来生成多尺度信息，因此产生较高的计算成本。相比之下，我们的 PWLT 将不同大小的窗口分配给不同注意力的头组，以提取多尺度特征，在节省计算上有显著成效。

3. 我们的方法

在本节中，首先介绍了用于捕捉全局上下文的 DSA，然后详细介绍了 PWLT 的架构。

3.1. 双重自注意力机制

为了有效且高效地捕捉多尺度信息，我们构建 DSAs 层以建立窗口金字塔。具体来说，在这一层，每个 DSA 在不同的头组工作，并用不同的窗口划分标准（如 2×2，3×3）来捕捉特征。尽管所有的头组都有不同的窗口大小，但 DSA 的工作方式很相似。不失一般性，我们仅介绍用 2×2 窗口划分的 DSA。如图 2 所示，DSA 有两部分组成：WSA 和 PSA。首先，将输入特征输入 WSA 中以计算在窗口中的局部注意力，这有助于捕捉局部依赖。然后 WSA 的输出通过 PSA，此时会充分重建窗口交互以捕获全局上下文。下面，我们将分别具体介绍 WSA 和 PSA 的细节。

3.1.1. WSA

WSA 的计算遵循传统的基于窗口的 Transformers[10][12]。具体点来说，给出输入特征$X \in \mathbb{R}^{H\times W\times C}$，$H$表示高度，$W$表示宽度，$C$表示$X$的通道数量。如图 2 所示，输入特征$X$被平均分成 2×2 的窗口，每个窗口对应一个特征块$X_i \in \mathbb{R}^{\frac{H}{2}\times\frac{W}{2}\times\frac{C}{2}}$，$i \in \{1,2,3,4,\}$。根据[4]，每个窗口独立计算自注意力。具体来说，三个线性投影$\{W_q^s \in \mathbb{R}^{C\times d}, W_k^s \in \mathbb{R}^{C\times d}, W_v^s \in \mathbb{R}^{C\times d}\}$首先被用来将$X_i$映射到三个低维嵌入空间中，包括查询$Q_i^s \in \mathbb{R}^{m\times d}$，键$K_i^s \in \mathbb{R}^{m\times d}$，值$V_i^s \in \mathbb{R}^{m\times d}$：

$$Q_i^s = Re(X_i)W_q^s, K_i^s = Re(X_i)W_k^s, V_i^s = Re(X_i)W_v^s, \forall i \tag{1}$$

其中$Re(\cdot)$表示重组操作，m是X_i中的标签数量，$d << C$表示每个头组的通道数。注意投影$\{W_q^s, W_k^s, W_v^s\}$是所有X_i的共享参数，以节省模型大小。因此用SoftMax(·)函数生成一个注意

力图$A_i^s \in \mathbb{R}^{m\times m}$，代表$X_i$中标签的依赖关系：

$$A_i^s = \text{SoftMax}\left(\frac{Q_i^s K_i^{s\top}}{\sqrt{d}}\right) \quad (2)$$

最后，通过注意力图A_i^s对V_i^s加权，得到 WSA 的输出中间特征$Z_i \in \mathbb{R}^{\frac{H}{2}\times\frac{W}{2}\times\frac{C}{2}}$：

$$Z_i = Re(A_i^s V_i^s) \quad (3)$$

到目前为止，我们已经成功使用 WSA 提取局部特征。然而，不同窗口之间缺乏信息交互。因此，我们引入了 PSA 模块来重建这些窗口连接。

3.1.2. PSA

给定从 WSA 中提取的中间特征Z_i，$i \in \{1,2,3,4,\}$，首先用全局池化来生成四个基于窗口的标签特征，这些标签特征分别代表每个窗口的全局信息。PSA 先将所有标签特征收集在一起，再依次将标签特征输入到层归一化、重塑、线性投影、注意力计算和特征重加权等操作。设$F \in \mathbb{R}^{4\times d}$为归一化标签特征，其中 4 代表划分好的窗口数量。然后F经过两次线性投影$\{W_q^p \in \mathbb{R}^{d\times d}, W_k^p \in \mathbb{R}^{d\times d}\}$，分别得到查询$Q^p \in \mathbb{R}^{4\times d}$和键$K^p \in \mathbb{R}^{4\times d}$：

$$Q^p = FW_q^p \quad K^p = FW_k^p \quad (4)$$

接下来使用SoftMax函数来生成一个注意力图$A^p \in \mathbb{R}^{4\times 4}$，它编码了窗口的依赖关系：

$$A^p = \text{SoftMax}\left(\frac{Q^p K^{p\top}}{\sqrt{d}}\right) \quad (5)$$

由于每行的A^p编码了一个特定窗口和其他所有窗口之间的关系，PSA 的最后一步是更新中间特征Z_i，通过A^p从所有窗口中收集信息，进而生成重新加权特征$Y_i \in \mathbb{R}^{\frac{H}{2}\times\frac{W}{2}\times d}$：

$$Y_i = \sum_j A_{ij}^p Z_j \quad i,j \in \{1,2,3,4\} \quad (6)$$

其中A_{ij}^p表示A^p中i^{th}行j^{th}列的元素。将所有Y_i组合得到DSA的输出为特征$Y \in \mathbb{R}^{H\times W\times d}$。回顾DSA实际上是在不同的头组内的金字塔窗口上进行操作的，因此我们必须将所有组连接在一起，并将他们输入线性投影$W_o \in \mathbb{R}^{C\times C}$，得到最终输出$\hat{X} \in \mathbb{R}^{H\times W\times C}$，其维度和输入特征$X$相同。

3.2. 复杂性分析

在上一节中，我们已经详细阐述了 DSA 是如何实现的。为了更好的展示 DSA 的优点，我们将在这一节中分析 DSA 的参数数量和计算成本。注意，我们假设输入特征图的大小为$N \times C$，其中N表示标签数量，C表示通道数量。

3.2.1. DSA 的参数

如图 2 所示，DSA 的参数起初来自三个部分：WSA、PSA 和的W_o线性投影层。在 WSA 中，来自W_q^s，W_k^s，W_v^s的参数总数量为$3C^2$。PSA 的参数数量取决于头组的数量。假设有D组，每组有C/D个通道，那么 PSA 的参数数量为$2C^2/D$（每组有$2C^2/D^2$个参数）。线性映射投影W_o有C^2个参数。因此，DSA 的总参数数量为$(4+2/D)C^2$。ViT[5]和 Swin[10]的参数数量为$4C^2$，与其相比我们的 DSA 虽然增加了一个参数，但是这一微小的增加可以忽略不计，并且这使得我们的模型在不用移位操作的情况下建立了不同窗口之间的关系。

3.2.2. DSA 的计算成本

计算成本在很大程度上决定了网络的工作效率。在ViT[5]中MHSA的计算复杂度为$4NC^2 + 2N^2C$。具体而言，$4NC^2$是进行查询、键、值的操作和来自W_o的计算成本。$2N^2C$是 MHSA 的主要计算成本，包含生成注意力图和与值矩阵进行乘法运算。

和 MHSA 相比，我们的 DSA 的主要计算量从$2N^2C$降低到$2N^2C/m$。因此，我们的方法有效地降低了 ViT 的工作量。具体而言，DSA 的工作量计算主要源于 WSA，PSA 和线性投影层W_o。为了表述的简洁性，我们假设所有的头组都被划分为m个窗口。WSA的计算量为$3NC^2 + 2N^2C/m$，其中$3NC^2$是查询Q_i^s、键K_i^s、值V_i^s的计算量，而$2N^2C/m$是生成注意力图和在m个窗口中与值矩阵的乘法运算的计算量（每个窗口的计算量为$2N^2C/m^2$）。PSA 的计算量为$2mC^2 + m^2C +$

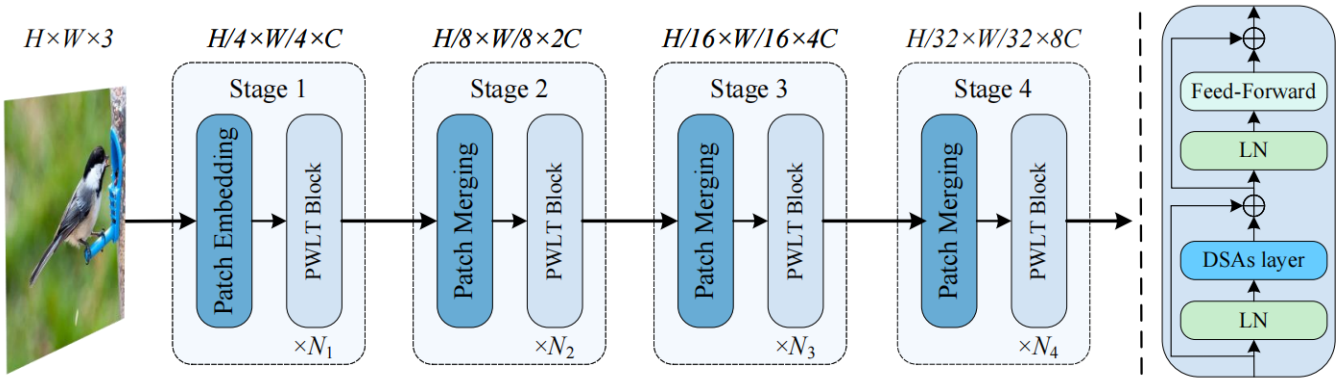

图 3 PWLT 网络架构总览。该架构分为四个阶段，每个阶段由一个补丁嵌入模块或补丁合并模块以及N_i个 PWLT 块组成。右列是 PWLT 块的详细实现。（彩色显示效果最佳）

表 1 PWLT 主干网络的详细架构。Conv(k×k，s)表示使用 k×k 滤波器核大小和步长 s 的卷积。n×n 表示特征图网格上的窗口大小。h 是头的索引，D 表示每个 PWLT 块中头组的总数。

阶　段	输入大小	层	PWLT-Tiny	PWLT-Base	PWLT-Medium
1	224×224	Patch Embedding	$Conv(7\times7, s=4)$		
	56×56	PWLT Block	$\begin{bmatrix} D=2 \\ 2\times2, h\in\{1\} \\ 3\times3, h\in\{2\} \end{bmatrix}\times1$	$\begin{bmatrix} D=2 \\ 2\times2, h\in\{1\} \\ 3\times3, h\in\{2\} \end{bmatrix}\times2$	$\begin{bmatrix} D=2 \\ 2\times2, h\in\{1\} \\ 3\times3, h\in\{2\} \end{bmatrix}\times3$
2	56×56	Patch Merging	$Conv(3\times3, s=2)$		
	28×28	PWLT Block	$\begin{bmatrix} D=3 \\ 2\times2, h\in\{1,2\} \\ 3\times3, h\in\{2\} \\ 5\times5, h\in\{4\} \end{bmatrix}\times1$	$\begin{bmatrix} D=3 \\ 2\times2, h\in\{1,2\} \\ 3\times3, h\in\{2\} \\ 5\times5, h\in\{4\} \end{bmatrix}\times4$	$\begin{bmatrix} D=3 \\ 2\times2, h\in\{1,2\} \\ 3\times3, h\in\{2\} \\ 5\times5, h\in\{4\} \end{bmatrix}\times6$
3	28×28	Patch Merging	$Conv(3\times3, s=2)$		
	14×14	PWLT Block	$\begin{bmatrix} D=3, C=64 \\ 2\times2, h\in\{1\sim4\} \\ 3\times3, h\in\{5,6\} \\ 5\times5, h\in\{7,8\} \end{bmatrix}\times4$	$\begin{bmatrix} D=3 \\ 2\times2, h\in\{1\sim4\} \\ 3\times3, h\in\{5,6\} \\ 5\times5, h\in\{7,8\} \end{bmatrix}\times16$	$\begin{bmatrix} D=3 \\ 2\times2, h\in\{1\sim4\} \\ 3\times3, h\in\{5,6\} \\ 5\times5, h\in\{7,8\} \end{bmatrix}\times30$
4	14×14	补丁合并	$Conv(3\times3, s=2)$		
	7×7	PWLT Block	$[D=1]\times1$	$[D=1]\times2$	$[D=1]\times3$

mNC。$2mC^2$是查询Q^p和键K^p的计算量，m^2C是生成注意力图的计算量，mNC是注意力图与特征图之间的矩阵乘法。最后，线性投影层W_o的计算量为NC^2。因此，DSA 的计算量为$(4N+2m)C^2+(2N^2/m+m^2+mN)C$。

然而，假设每个窗口有个M标签，那么在 Swin[10]中 WSA 的计算复杂度为$4NC^2+2MNC$。其中$4NC^2$表示查询、键、值的操作和来自Wo的计算量，$2MNC$是生成注意力图和所有窗口中与值矩阵相乘的计算量（每个窗口的计算量为$2M^2C$，一共有N/M个窗口）。尽管我们的方法和 Swin[10]相比有额外的计算量，但是它避免了使用移动窗口的方法。相反，它利用注意力机制建立了全面的连接，提供了一种更简单的解决的方法，并且具有更好的性能。

3.3. PWLT 的网络架构

如图 3 所示，PWLT 继承了具有四层的分层架构，其中特征分辨率是逐渐减小的，通道数量增加至原来的两倍。因此，它生成了一系列特征图，其大小有$\frac{H}{4}\times\frac{W}{4}\times C$，$\frac{H}{8}\times\frac{W}{8}\times2C$，$\frac{H}{16}\times\frac{W}{16}\times4C$，$\frac{H}{32}\times\frac{W}{32}\times8C$，其中$H$和$W$分别代表输

入图像的高度和宽度，C表示通道数。除了第一层包含补丁嵌入模块，其他层都由补丁合并模块和重复 PWLT 块组成。如图 3 的右图所示，DSA 层依照传统的基于 ViT 的设计[5][10]，含有前馈神经网络（FFN，Feed Forward Networks），残差连接和层归一化（LN，Layer Normalization）。

基于前述的 Transformer 架构[6][10][21]，我们通过在每个阶段堆叠不同数量的 PWLT 块，研发出了不同深度的 PWLT。因此，我们提出三种变形：PWLT-Tiny，PWLT-Base 和 PWLT-Medium，其差异仅在于不同阶段的层数。PWLT 骨架的详细结构如表 1 所示。具体而言，补丁嵌入模块采用步长为 4 的 7×7 卷积操作，通过一步提高四倍有效降低了输入分辨率，同时补丁合并模块采用步长为 2 的 3×3 卷积操作，将输入尺寸的下采样降至原来的二分之一。每个 PWLT 块由多组注意力头部组成，用来生成一个金字塔窗口以及进行 DSA 操作。以第二阶段为例，除了前两个头被视为一个组并使用 2×2 的窗口外，其余两个头分别单独成组，分别对应 3×3 和 5×5 的窗口分区大小。在最后阶段，由于特征的输入分辨率太小而无法分割（例如，7×7×512），因此在此阶段我们直接采用 MHSA 而非 DSA。

4. 实验

为了评估 PWLT，我们在 ImageNet-1K[7]和 CIFAR100[14]数据集上进行了大量实验。

4.1. 数据集与评估指标

4.1.1. 数据集

ImageNet-1K[7]是图像分类领域最流行的数据集。它包含 1000 种类别，每个类别包含大约 1000 张图像。具体而言，此数据集包含 128 万张训练图像和 5 万张测试图像。该数据集中的每幅图像在物体类型、比例和背景方面各不相同，对评估图像分类模型的性能来说是一个具有挑战性和多样性的数据集。而 CIFAR100[14]是一个规模较小的数据集，仅包含 100 个类别和 60,000 张分辨率为 32×32 的彩色图像，其中 50000 张用于训练，10000 张用于测试。相较于 ImageNet-1K 数据集，CIFAR100 的类别略少，但其类别区分难度更高且采用较低的图像分辨率，为图像分类模型带来了一系列独特的挑战。

4.1.2. 评估指标

为了与其他最先进的轻量级方法进行公平比较，我们采用了标准评估指标：Top-1 识别准确率，该指标衡量的是模型预测总数中正确预测的比例。它直观地评估了模型将实例正确分类到各自类别的能力。此外，采用每秒浮点运算次数（FLOPs）衡量实现效率。

4.2. 实验细节

4.2.1. 训练设置

PWLT 是在配备 8 张 RTX 3090 GPU 卡的硬件服务器平台上实现的。软件代码基于 MMPretrain 工具箱，一个用于图像分类的开源库。我们采用广泛使用的 AdamW[6]来优化 PWLT，其中权重衰减和初始学习率分别设置为 0.05 和 1×10^{-3}。此外，采用余弦学习策略，最小学习率为 1×10^{-5}。我们在 mageNet-1K[7]和 CIFAR100[14]数据集上对模型进行 300 个周期的训练，前五个周期用于预热。为了增强数据多样性并提高泛化能力，我们采用了各种数据增强技术，包括随机裁剪、随机翻转、Mixup[23]、CutMix[24]以及随机擦除[10]。

4.2.2. 损失函数设置

我们采用标签平滑损失（LSL，Label Smoothing Loss）[10]来监督 PWLT 的整个过程。该损失函数将交叉熵[2]与标签平滑项相结合，通过将真实标签的概率分布与其他类别的概率分布进行平均来生成目标标签。在这种情况下，模型通常能够实现更好的泛化性能，尤其是在数据嘈杂或不确定的情况下。以下是实现细节：

$$\mathcal{L} = (1-\epsilon)\mathcal{L}_{CE} - \epsilon/N \sum \log y_{pred} \qquad (7)$$

其中$\mathcal{L}_{CE}$是交叉熵损失，ϵ表示非负平滑参数，该参数平衡了交叉熵损失和平滑项之间的关系，N表示种类数，y_{pred}表示预测概率。

4.3. ImageNet-1K 数据集上的实验结果

表 2 展示了 PWLT 与最先进网络的比较结果。为了增加轻量级主干网络的多样性，我们同

表 2 在 ImageNet-1K 数据集[7]上，从分类准确率和实现效率方面与最先进的方法进行了比较。所有方法的输入分辨率均固定为 224×224 以保证公平。

方法	年	参数（M）	FLOPs (G)	Top-1 (%)
ResNet18[2]	CVPR2016	11.7	1.8	69.8
DeiT-T[24]	ICML2021	5.7	1.3	72.2
PVT-T[23]	ICCV2021	13.2	1.9	75.1
MoCoVIT-D[25]	ARXIV2022	12.1	0.2	75.5
ConViT-T[26]	ICML2021	10.0	1.0	76.7
ViL-T[27]	ICCV2021	6.7	1.3	76.7
Swin[10]	ICCV2021	11.0	1.5	76.9
XCiT-T12[28]	NIPS2021	7.0	1.2	77.1
PoolFormer-S12[29]	CVPR2022	11.9	2.0	77.2
Flatten-PVT-T[30]	ICCV2023	12.2	2.0	77.8
RegNetY-1.6G[31]	CVPR2020	11.2	1.6	78.0
LocalViT-PVT[32]	ARXIV2021	13.5	4.8	78.2
PVT-V2-B1[33]	CVM2022	13.1	2.1	78.7
SepViT[6]	ARXIV2022	12.2	1.8	78.8
FasterNet-T[34]	CVPR2023	15.0	1.9	78.9
PWLT-T	-	12.2	1.8	79.2
ResNet50[2]	CVPR2016	25.0	4.1	76.2
RegNetY-4G[31]	CVPR2020	20.6	4.0	79.4
PVT-S[23]	ICCV2021	24.5	3.8	79.8
DeiT-S[24]	ICML2021	22.1	4.6	79.9
Couplformer-T[19]	WACV2023	28.0	6.4	80.5
Swin-T[10]	ICCV2021	28.3	4.5	81.2
ConViT-S[26]	ICML2021	27.0	5.4	81.3
FasterNet-S[34]	CVPR2023	31.1	4.6	81.3
PoolFormer-S36[29]	CVPR2022	31.0	5.0	81.4
Flatten-PVT-S[30]	ICCV2023	21.7	4.0	81.7
XCiT-S12[28]	NIPS2021	26.0	4.8	82.0
PVT-V2-B2[33]	CVM2022	25.4	4.0	82.0
ViL-Small[27]	ICCV2021	24.6	4.9	82.0
PWLT-B	-	29.8	4.9	82.1
ViT-B[5]	ICLR2021	86.8	17.6	77.9
ResNet-101[2]	CVPR2016	45.0	7.9	79.8
RegNetY-8G[31]	CVPR2020	39.2	18.0	79.9
PVT-M[23]	ICCV2021	44.2	6.7	81.2
PVT-L[23]	ICCV2021	61.4	9.8	81.7
DeiT-B[24]	ICML2021	86.6	17.5	81.8
ConVIT-S+[26]	ICML2021	48.0	10.0	82.2
Couplformer-S[19]	WACV2023	49.0	20.4	82.3
PoolFormer-M48[29]	CVPR2022	73.0	11.6	82.5
XCiT-S24[28]	NIPS2021	48.0	9.1	82.6
FasterNet-M[34]	CVPR2023	53.5	8.7	83.0
Swin-S[10]	ICCV2021	49.6	8.7	83.1
PWLT-M	-	52.2	9.3	83.2

时采用了基于卷积神经网络（CNN）[2][31]和基于 Transformer[23][26][10][33]的方法。显然，PWLT 的表现优于基于卷积神经网络（CNN）的模型，比如 ResNet[2]和 RegNetY[31]。例如，在模型大小和 FLOPs 相似的情况下，PWLT-T/B/M 的 Top-1 准确率分别比 ResNet32/50/101 和 RegNetY-1.6G/4G/8G 高出 9.6%/5.9%/3.4%和 1.2%/2.7%/3.3%。此外，与最近的基于 Transformer 的模型相比，PWLT 也取得了更优的结果。例如，在模型大小和 FLOPs 相似的情况下，PWLT-T/B/M 比 Swin/T/S[10]分别高出 2.3%、0.9%和 0.1%的 top-1 准确率。与 SepViT[6]相比，我们的 PWLT-T 由于能够捕捉多尺度信息而具有更好性能。一些方法具有较小的模型大小和 GFLOPs（例如，DeiT T/S/B 和 XCiT-T/S12/S24），但与 PWLT-T/B/M 相比，它们的 top-1 准确率分别下降了 7.0%/2.2%/1.4%和 2.1%/0.1%/0.6%。

表 3 在 CI FAR100 数据集[14]上，与最先进方法在分类准确率和实现效率方面的比较。“-”表示未报告结果。

方法	年	参数（M）	FLOPs (G)	Top-1 (%)
ConvNet				
ResNet18[2]	CVPR2016	11.2	1.8	75.6
ResNet34[2]	CVPR2016	21.3	3.7	76.7
SENet34[35]	CVPR2018	21.6	-	77.9
Transformer				
DeiT-S[24]	ICML2021	21.4	5.5	63.7
PVT-T[23]	ICCV2021	15.8	0.6	69.6
Swin-T[10]	ICCV2021	27.5	1.4	78.0
SepViT[6]	ARXIV2022	11.8	0.5	78.1
PWLT	-	11.8	0.5	78.4

4.4. CIFAR100 数据集上的实验结果

在本节中，我们在 CIFAR100[14]数据集上进行了实验，以进一步评估我们方法的有效性。表 3 展示了实验结果。无论是基于 CNN 还是基于 Transformer 的模型，我们的 PWLT 都能实现最佳准确率，同时有甚至更低的模型大小和 FLOPs。特别注意，与 DeiT-S[24]、PVT-T[23]和 Swin-T[10]相比，PWLT 在模型大小更小（前两者为 11.8M，PWLT 为 21.4/15.8/27.5M）且 FLOPs 更少（前两者为 0.5，PWLT 为 5.5/0.6/1.4G）的情况下， Top-1 准确率显著提升（14.7%、8.8%和 0.4%）。在模型大小和 FLOPs 相同的情况下，PWLT 将 SepViT[6]的 Top-1 准确率从 78.1%提高到了 78.4%。

4.5. 消融实验

为了解析 PWLT 的内在机制，本节报告了一系列消融实验的结果。

4.5.1. WSA 和 PSA 的消融实验

表 4 展示了 ImageNet-1K [7]上的一些消融

表 4 DSA 中各部分有效性的消融实验。

WSA	PSA	参数（M）	FLOPs (G)	Top-1 (%)
		13.25	2.18	76.8
✓		11.79	1.46	75.1
✓	✓	12.36	1.73	76.9

表 5 PSA 中的池化方法的消融实验。AvgPooling 和 Conv 分别表示平均池化和步长卷积。

方法	参数（M）	FLOPs (G)	Top-1 (%)
Conv	12.4	1.8	78.7
AvgPooling	12.2	1.8	79.2

表 6 金字塔窗口的消融实验，其中特征图被划分为 n ×n 的窗口网格。

2×2	3×3	5×5	7×7	参数（M）	FLOPs (G)	Top-1 (%)
✓				12.1	0.5	77.12
✓	✓			11.9	0.5	78.07
✓	✓	✓		11.8	0.5	78.40
✓	✓	✓	✓	11.8	0.5	78.20

实验，这些研究量化了两个主要部分：WSA 和 PSA 的各自作用。我们在固定 WAS 和 PSA 的情况下，首先生成传统的基于 ViT 的主干网络的基线模型。然后逐步加入 WAS 和 PSA。当采用基于 ViT 的主干网络时，其分类准确率与 PWLT 相当，但参数数量和 FLOPs 最多。当采用 WSA 来减少模型大小（11.79Mvs13.25M）和 FLOPs（1.46Gvs2.18G）时，Top-1 准确率从 76.8% 下降到 75.1%，这可能是由于分割好的窗口之间的连接丢失。最后引入 PSA 来恢复窗口依赖关系，此时我们得到了与基线模型相同的结果，模型大小和 FLOPs 仅略有增加。

4.5.2. PSA 池化方法的消融实验

为了评估 PSA 中池化方法（如平均池化和步长卷积）对网络的影响，我们在 ImageNet-1K[7]上开展了一些消融实验。实验结果如表 5 所示。显然，和平均池化相比，步长卷积会引入额外的参数（12.2M vs 12.4M）。然而，相较于步长卷积，平均池化具有更好的 Top-1 准确率（79.2% vs 78.7%），这可能是因为卷积未能有效的从Z_i提取信息。

4.5.3. 金字塔窗口的消融实验

头组的数量决定了生成窗口的大小，这极大地影响了多尺度上下文表示能力和计算效率之间的平衡。因此我们在 CIFAR100 [14]数据集上随着窗口大小的改变来评估其性能变化。实验结果如表 6 所示。随着加入不同大小的窗口，模型的性能稳步提升，这表明融入多尺度信息能够有效提升模型的性能。我们可以观察到，当仅使用 3 个大小的窗口（2×2、3×3 和 5×5）时，可达到最佳效果，因此我们将其作为 PWLT 的默认设置。注意，采用更多尺寸的金字塔窗口会减小模型的大小，这可能是因为在W_q^p和W_k^p中使用的参数较少。

5. 结论与未来展望

在本文中，我们提出了一种新颖的网络，叫做基于金字塔窗口的图像分类轻量级变换器[7]。相较于以前的研究，我们的方法在两个关键的方面有所不同。首先，我们提出了双重自注意力机制 DSA 来重建不同窗口之间的连接，以捕捉全局特征。其次，我们的模型融合了金字塔窗口，提高了网络提取多尺度特征的能力，以识别不同尺寸的目标。在本次研究中，我们在 ImageNet-1K 和 CIFAR100 数据集中评估 PWLT 的性能。实验表明，我们的方法在检测精度与实现效率间达到了最优的平衡。未来，我们希望将 PWLT 引入到密集预测任务中，如目标检测[8]和语义分割[9]。此外，未来的研究将侧重于优化和增强 PWLT，在更广泛的计算机视觉应用中进一步提高其性能。

参考文献

[1] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, CoRR abs/1409.1556 (2014).

[2] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition,

2016, pp. 770–778.
[3] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International conference on machine learning, 2019, pp. 6105–6114.
[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).
[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, in: International conference on learning Representations, 2021.
[6] W. Li, X. Wang, X. Xia, J. Wu, X. Xiao, M. Zheng, S. Wen, Sepvit: Separable vision transformer, arXiv preprint arXiv:2203.15380 (2022).
[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2009, pp. 248–255.
[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: European conference on computer vision, 2014, pp. 740–755.
[9] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, A. Torralba, Scene parsing through ade20k dataset, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5122–5130.
[10] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 10012–10022.
[11] Q. Zhang, Y. Xu, J. Zhang, D. Tao, Vsa: Learning varied-size window attention in vision transformers, in: European conference on computer vision, 2022, pp. 466483.
[12] Y. Lee, J. Kim, J. Willette, S. J. Hwang, Mpvit: Multipath vision transformer for dense prediction, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2022, pp. 7287–7296.
[13] Y. Liu, N. Ong, K. Peng, B. Xiong, Q. Wang, R. Hou, M. Khabsa, K. Yang, D. Liu, D. S. Williamson, et al., Mmvit: Multiscale multiview vision transformers, arXiv preprint arXiv:2305.00104 (2023).
[14] A. Krizhevsky, Learning multiple layers of features from tiny images, 2009.
[15] Z. Huang, Y. Ben, G. Luo, P. Cheng, G. Yu, B. Fu, Shuffle transformer: Rethinking spatial shuffle for vision transformer, arXiv preprint arXiv:2106.03650 (2021).
[16] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).
[17] A. Hassani, S. Walton, J. Li, S. Li, H. Shi, Neighborhood attention transformer, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2023, pp. 6185–6194.
[18] M. Ding, B. Xiao, N. Codella, P. Luo, J. Wang, L. Yuan, Davit: Dual attention vision transformers, in: European conference on computer vision, 2022, pp. 74–92.
[19] H. Lan, X. Wang, X. Wei, Couplformer: Rethinking vision transformer with coupling attention map, ArXiv abs/2112.05425 (2021).
[20] Y.-H. Wu, Y. Liu, X. Zhan, M.-M. Cheng, P2t: Pyra mid pooling transformer for scene understanding, IEEE Transactions on pattern analysis and machine intelligence 45 (2021) 12760–12771.
[21] S. Ren, D. Zhou, S. He, J. Feng, X. Wang, Shunted selfattention via multi-scale token aggregation, in: Proceedings of the IEEE

conference on computer vision and pattern recognition, 2022, pp. 10853–10862.

[22] C.-F. R. Chen, Q. Fan, R. Panda, Crossvit: Crossattention multi-scale vision transformer for image classification, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2021, pp. 357366.

[23] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang,T. Lu, P. Luo, L. Shao, Pyramid vision transformer: A versatile backbone for dense prediction without convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2021, pp. 568578.

[24] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, in: International conference on machine learning, 2021, pp. 1034710357.

[25] H. Ma, X. Xia, X. Wang, X. Xiao, J. Li, M. Zheng, Mocovit: mobile convolutional vision transformer, arXiv preprint arXiv:2205.12635 (2022).

[26] S. d'Ascoli, H. Touvron, M. L. Leavitt, A. S. Morcos, G. Biroli, L. Sagun, Convit: Improving vision transformers with soft convolutional inductive biases, in: International conference on machine learning, 2021, pp. 2286–2296.

[27] P. Zhang, X. Dai, J. Yang, B. Xiao, L. Yuan, L. Zhang, J. Gao, Multi-scale vision longformer: A new vision transformer for high-resolution image encoding, in: Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 2998–3008.

[28] A. Ali, H. Touvron, M. Caron, P. Bojanowski, M. Douze, A. Joulin, I. Laptev, N. Neverova, G. Synnaeve, J. Verbeek, et al., Xcit: Cross-covariance image transformers, Advances in neural information processing systems 34 (2021) 20014–20027.

[29] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, S. Yan, Metaformer is actually what you need for vision, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2022, pp. 10819–10829.

[30] D. Han, X. Pan, Y. Han, S. Song, G. Huang, Flatten transformer: Vision transformer using focused linear attention, Proceedings of the IEEE/CVF international conference on computer vision (2023) 5938–5948.

[31] I. Radosavovic, R. P. Kosaraju, R. Girshick, K. He, P. Dollár, Designing network design spaces, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2020, pp. 10428–10436.

[32] Y. Li, K. Zhang, J. Cao, R. Timofte, L. Van Gool, Localvit: Bringing locality to vision transformers, arXiv preprint arXiv:2104.05707 (2021).

[33] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, Pvt v2: Improved baselines with pyramid vision transformer, Computational visual media 8 (3) (2022) 415–424.

[34] J. Chen, S. hong Kao, H. He, W. Zhuo, S. Wen, C.H. Lee, S.-H. G. Chan, Run, don't walk: Chasing higher flops for faster neural networks, Proceedings of the IEEE conference on computer vision and pattern recognition (2023) 12021–12031.

[35] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 71327141.