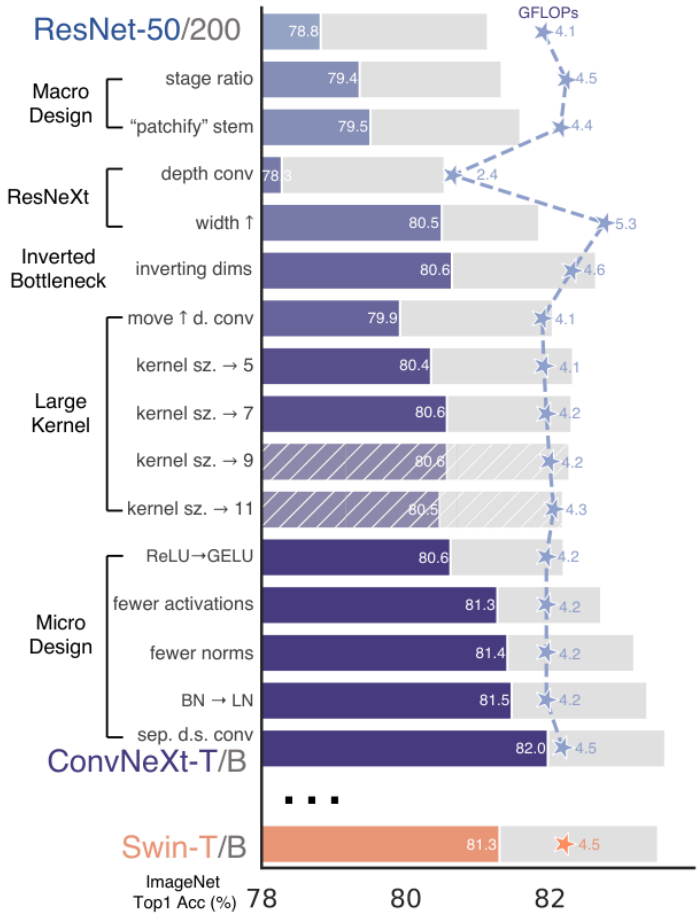
背景

目前很多视觉任务都使用基于 Transformer 的模型，而且一些榜上排名靠前的很多都是使用了 Transformer。那么纯卷积网络就一定比使用了 Transformer 的网络性能差么？使用 Vision Transformers 相比 CNN 要更难训练。比如 Transformer 需要更多的训练数据，需要迭代更多的轮数，需要更多的数据增强，且对数据增强很敏感。而且算力要求太高。

这篇文章就提出了纯卷积神经网络 ConvNeXt，模型是基于 ResNet50/200 模型，一开始使用的训练策略是基于 ViT 的，训练轮数从原来的 90 轮改为 300 轮，使用的优化器是 AdamW，使用了 Mixup，Cutmix，随机增强，随机擦除等数据增强手段，以及一些正则化方法，以这个作为实验基准，进行后续的改进。



上图展示了在相同的 FLOPs 下，ConvNeXt 与 Swin Transformer 在 ImageNet 数据集上的对比，可以看出纯卷积网络的准确率已经超过了 Swin Transformer。

方法

①macro design

(1)改变模块堆叠的比例。

从图 1 中红色框的 ResNet50 框架可以看出，原 bottle neck-residual block 的堆叠比例是[3, 4, 6, 3]。

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

图 1 ResNet 结构框架示意图

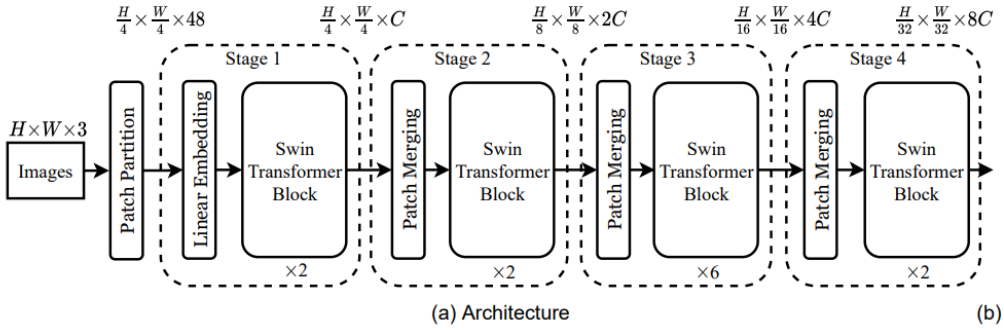


图 2 Swin Transformer

文章按照 Swin Transformer 中的 block 设计比例[1, 1, 3, 1]，将 ResNet50 的 block 堆叠比例进行了调整，调整为[3, 3, 9, 3]，准确率从 78.8%提升到 79.4%。

(2)改变下采样的方式

如图 1 蓝色方框中的所示，原 ResNet50 网络采用 7×7，步长为 2 的卷积操作对图像进行下采样，但是在 Swin Transformer 中采用的是 Patch Merging 操作。

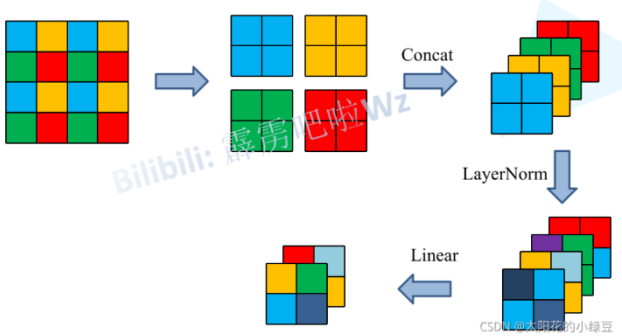


图 3 Patch Merging

假设输入特征图大小为 4×4×1，Patch Merging 会将每个 2×2 的相邻像素

划分为一个 patch，然后将每个 patch 中相同位置（同一颜色）像素给拼在一起就得到了 4 个特征图。接着将这四个特征图在深度方向进行 concat 拼接，然后在通过一个 LayerNorm 层。最后通过一个全连接层在特征图的深度方向做线性变化。

（原文链接：https://blog.csdn.net/qq_37541097/article/details/121119988）

作者将原来的下采样层换为和 Swin Transformer 中一样的 4×4，步长为 4 的 patchify 层。准确率有所提升。

②ResNeXt 风格

ResNeXt 网络每个模块使用组卷积，使用类似于 Inception 模块的设计，但不同的是每个路径使用相同的拓扑结构，作者采用的是深度可分离卷积，即卷积的组数和特征通道数相同。第一阶段最开始的通道数由原来的 64 改为和 Swin-T 一样的通道数 96。准确率达到了 80.5%

③反向瓶颈结构

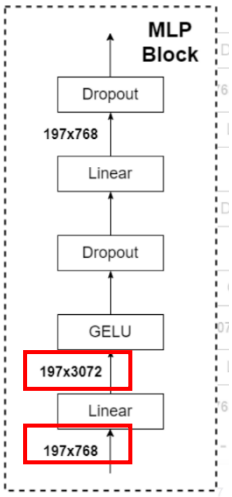


图 4 Vision Transformers 中的 MLP 结构示意图

从图 4 中可以看出 Vision Transformers 的 MLP 结构中间隐藏层部分的特征维度是输入图像的 4 倍，这样的结构和 MobileNetV2 里提出的反向残差结构类似。故作者也采用了反向瓶颈结构。

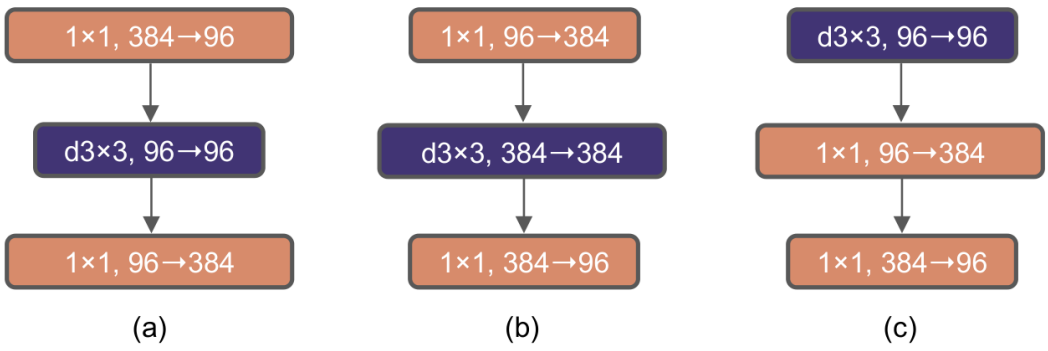


图 5 反向瓶颈结构

图 5 中(a)是 ResNeXt 使用的瓶颈结构，(b)是反向瓶颈结构，(c)是本文提出的反向瓶颈结构。模型准确率有所提升。

④大卷积核

在 Transformer 中一般都是对全局做 self-attention，但现在主流的卷积神经网络都是采用 3×3 大小的卷积核，之前 VGG 中提出了“堆叠多个 3×3 卷积可以替代一个更大的卷积”，而且现在的 GPU 设备针对 3×3 大小的卷积核做了优化，所以运行更高效。

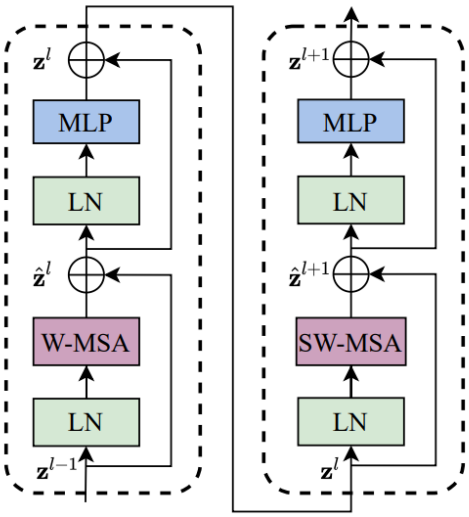


图 6 Swin Transformer Blocks

(1)调整深度可分离卷积的位置

参照 Swin Transformer 的结构，MSA 这种复杂且低效的模块其输出通道数少，将其放在 MLP 模块之前。所以设计的反向瓶颈结构中，将深度可分离卷积的位置进行了迁移，从图 5 的(b)改为(c)，这样改动后，准确率和 FLOPs 都有所下降。

(2)增加卷积核尺寸

作者对深度可分离卷积的卷积核大小寸尺做了一系列选择，通过对比发现卷积核尺寸取到 7 时，准确率饱和。因此将原来 3×3 卷积改为 7×7 卷积。

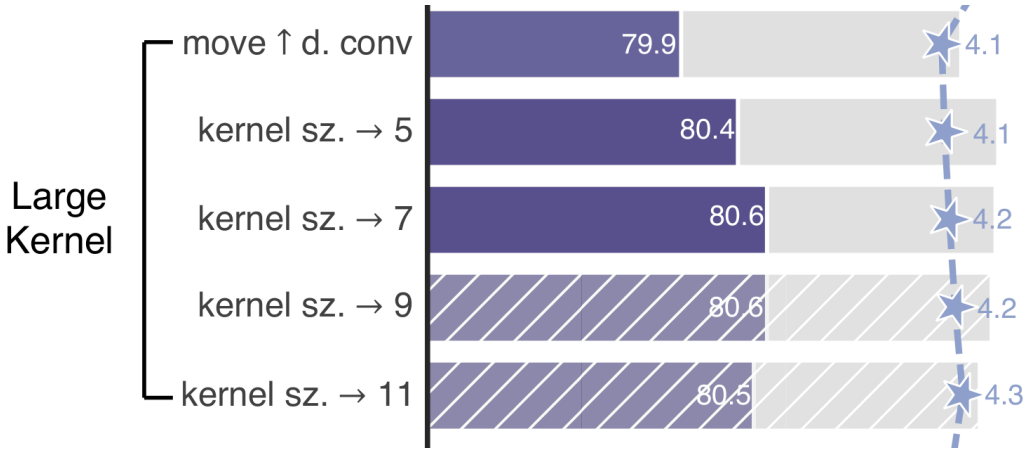


图 7 卷积核大小选择准确率对比

⑤细节处的设计

(1)激活函数的选择

通过和 Transformer 对比发现在 Transformer 中使用的激活函数都是 GELU，因此将卷积神经网络中常用的 ReLU 激活函数替换为 GELU 激活函数，发现准确率并没有变化。

(2)激活函数个数的选择

在卷积神经网络中可以发现在卷积完后都会接一个激活函数，但是从图 4 的 MLP 结构中发现，只有在第一个全连接层后面使用了激活函数。因此作者也减少了激活函数的个数。发现采用这样的变动后，准确率有所提高。

(3)归一化层的个数选择

从 Swin Transformer 的结构中可以看出其归一化层的使用也很少，因此，作者也减少了归一化层的使用。此时准确率已经超过了 Swin-T。

(4)将 BN 替换为 Layer Normal

在 Transformer 中基本都用的 Layer Normalization，因此作者也采用了同样的操作，准确率有提升。

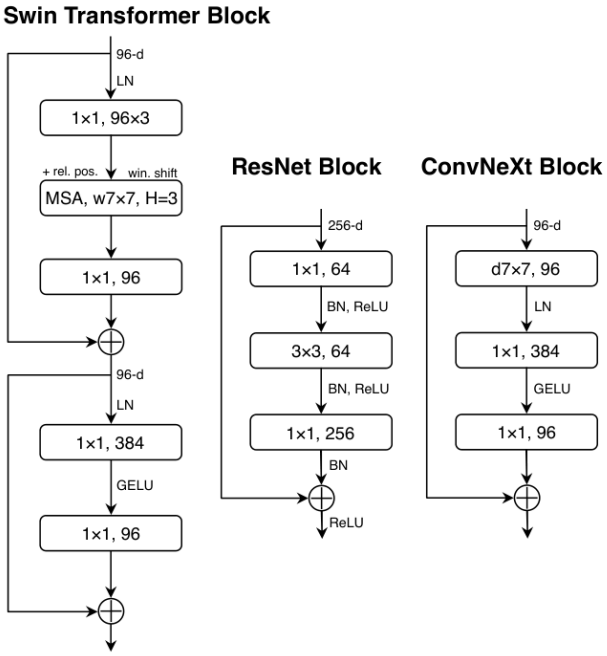


图 8 细节改变

(5)将下采样层分离出来

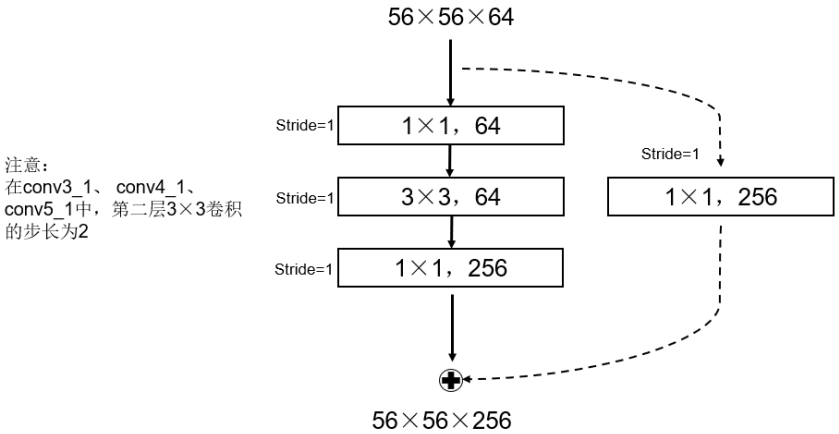


图 9 ResNet 中的虚线残差连接

如图 9 所示，在 ResNet50 网络中 conv3_1、conv4_1、conv5_1 的 3×3 卷积以及跳跃连接的步长都为 2 实现下采样。但是在 Swin Transformer 中下采样操作是通过单独的 Patch Merging 实现的，因此作者也为实现下采样单独设计了一个网络层结构，即一个 Laryer Normalization 加上一个卷积核大小为 2 步长为 2 的卷积。准确率就提升到了 82.0%

最后，效仿 Swin Transformer，也设计了四个版本，即 T/S/B/L.

- ConvNeXt-T: $C = (96, 192, 384, 768)$, $B = (3, 3, 9, 3)$
- ConvNeXt-S: $C = (96, 192, 384, 768)$, $B = (3, 3, 27, 3)$
- ConvNeXt-B: $C = (128, 256, 512, 1024)$, $B = (3, 3, 27, 3)$
- ConvNeXt-L: $C = (192, 384, 768, 1536)$, $B = (3, 3, 27, 3)$
- ConvNeXt-XL: $C = (256, 512, 1024, 2048)$, $B = (3, 3, 27, 3)$

图 10 四种版本的参数配置