Vision Transformer

用 *Transformer* 架构来解决视觉问题

结构

Vision Transformer 整体架构可分为三个部分。

- Linear Projection of Flattened Patches(Embedding Layer)
  - 16×16卷积核，stride=16， 卷积核个数为768
  - 拼接 `[class]token`
  - 叠加 Position Embedding
- Transformer Encoder
  - Encoder Block
  - MLP Block
- MLP Head
  - Linear

Patch Embedding

- 将图像划分为固定大小的 patches（如 16×16），展平后通过线性投影转换为 token*(image patches equal to tokens)* 嵌入。
- 添加可学习的 [class] token 用于最终分类。将其输入作为图像整体特征，作为整个模型的输出。
- 添加位置编码(*Position Embedding*)（在论文中选用1D），使模型感知空间位置信息
  - 根据嵌入的位置信息，对图像内的距离进行编码。更近的patches具有更类似的嵌入 *(Figure 7)*

Transformer Encoder

- 使用标准 Transformer 编码器结构

MLP Layers

包含两层具有 GELU 非线性变换。

实验与发现

1. 预训练与评测
   - **数据集**：ImageNet-1k（1.3M）、ImageNet-21k（14M）、JFT-300M（303M）。
   - **迁移任务**：ImageNet（ReaL）、CIFAR-10/100、Oxford Pets、Flowers、VTAB等。
   - **核心结论**：
     - ViT 在**大规模预训练**后，迁移到中小型数据集上表现优异，甚至超过卷积网络（如BiT、Noisy Student）。
     - 在**计算效率**上，ViT 训练所需计算资源显著少于同等性能的 ResNet。
2. 自注意力机制
   - 一些注意力头在浅层已经关注到了图像的全局整体信息（理解成"注意力距离"很远）

- 其他注意力头在浅层的注意力距离始终很小，这种高度的局部化可能类似于CNN中早期卷积层
- 注意力距离随着网络深度的增加而增加

3. 数据规模对模型性能的影响
   - 小数据（如 ImageNet）
     - ViT 表现不如 ResNet，因缺乏卷积的归纳偏置
   - 中大数据（如 ImageNet-21k、JFT-300M）
     - ViT 性能反超 ResNet，说明大数据可弥补结构先验的不足
   - 缩放规律
     - 模型性能随数据和模型规模单调提升，未出现饱和
4. 自监督预训练探索
   - 引入masked patch prediction来模拟Bert中掩码语言建模任务
   - 初步结果显示有提升（+2%），但仍落后于有监督预训练（-4%）

结论

1. Transformer模型可直接应用于图像分类任务，无需引入图像特定的归纳偏置。
2. 小规模数据上略输卷积神经网络；中等或者大规模数据集上，表现相当于或者优于卷积神经网络。随模型与数据规模上升，性能单调提升。
3. 在计算效率上，训练同等精度的模型，ViT模型比CNN更有优势。

未来方向

1. 将 ViT 应用于其它计算机视觉任务，例如检测、分割等
2. 探索更有效的**自监督训练**策略
3. 研究小样本学习下的 ViT 泛化能力

问题及解答

1. 什么是"**归纳偏差**"?
   CNN 的局部性、平移等变等结构先验。
   ViT 需通过更大数据和位置编码弥补先验不足。
2. 代码实现过程中，Dropout/DropPath 的区别是什么?
   两者都属于常用的正则化技术。

   Dropout 作用于神经元。
   DropPath 作用于**整条残差分支**，以概率 p 随机丢弃整个残差分支（只保留恒等映射部分），并做 $1/(1-p)$ 缩放以保持期望，推理期关闭。在ViT中使用DropPath，相当于训练多个不同深度的子网络，可以缓解过拟合，随机跳过某些Transformer块，并为梯度提供更直接的传播路径，从而提高模型的泛化能力。
3. 为什么几乎所有阶段都使用了Pre-Norm?
   LN 放在子层前能稳定深层训练与梯度传递，以保证深堆叠可训练。
4. 分辨率变化如何处理位置编码?
   论文实践中常对可学习位置编码按新长度**插值**，以保持迁移到更高/更低分辨率时的性能稳定。具体见补充部分。
5. drop_path_rate 进行了一个线性递增，其作用是?
   深层更强正则，浅层少正则，从 0 到设定上限线性分配到各层。

补充

插值

原论文中提出，若输入更高分辨率的图像，仍保持块的大小不变，会导致更长的序列长度（patches 的数量），此时预训练时的位置嵌入不一定有意义。因此我们考虑在原始图像中的位置对预训练的位置嵌入进行二维插值，这也是唯一将关于图像二维结构的归纳偏置手动注入vision transformer的地方。

如何理解插值？

在微调时，将预训练好的位置编码从旧的、较短的序列长度，智能地“扩展”到新的、更长的序列长度（如576），从而使模型能够处理高分辨率图像。在论文中，具体使用的是2D插值。具体步骤如下：

- **Reshape**：将预训练好的**一维位置编码序列**（形状为`[196, D]`，其中D是嵌入维度）重新排列成一个**二维网格。**这个网格的尺寸对应于原始图像被分割后的块网格尺寸（例如将196个位置编码重塑为`[14, 14, D]`的张量）。
- **Interpolate**：对这个`[14, 14, D]`的二维网格进行**上采样**，将其扩大到所需的网络尺寸。
    - 例如，目标分辨率是384x384，块大小16x16，那么新网格是24x24。因此，使用插值算法将`[14, 14, D]`的网格插值到`[24, 24, D]`。
- **Flatten**：将插值后得到的新二维位置编码网格（形状为`[24, 24, D]`）重新展平成一维序列（形状为`[576, D]`）。

这很好地解决了预训练与微调时分辨率不匹配的问题。