ViT 模型

1. 图像预处理与 Patch 划分

假设输入图像尺寸为 (H \times W)（例如 (224 \times 224)），通道数为 (C)（通常为 3）。ViT 首先将图像分割为固定大小的 patch，通常设定 patch 尺寸为 (P \times P)（例如 (16 \times 16)）。

- **Patch 数量**
  每个方向上的 patch 数量为 $N_h = \frac{H}{P}, \quad N_w = \frac{W}{P}$因此总$patch$数量为$N = N_h \times N_w = \frac{H \times W}{P^2}$.
- **Flatten 处理**
  每个 patch 的大小为 $P \times P \times C$，经过 flatten 后成为一个长度为 $P^2 \cdot C$ 的向量。
- **线性投影（Patch Embedding）**
  使用一个线性层将每个 patch 向量映射到一个固定的嵌入维度 (D)。令投影矩阵为 $E \in \mathbb{R}^{(P^2 \cdot C) \times D}$，则第$i$个$patch$的嵌入为$x_i = \text{flatten}(\text{patch}_i)\, E, \quad i = 1, 2, \ldots, N$.将所有 patch 嵌入堆叠，得到$X \in \mathbb{R}^{N \times D}$.

2. 类别标记与位置编码

为了进行分类任务，ViT 引入一个专用的类别标记（class token），记为 x_{class}，并将其与 patch 嵌入一起构成输入序列：

$$X_0 = \begin{bmatrix} x_{\text{class}} \\ x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \in \mathbb{R}^{(N+1) \times D}.$$

由于 Transformer 对序列顺序没有天然敏感性，需要加入位置编码
$E_{\text{pos}} \in \mathbb{R}^{(N+1) \times D}$，最终输入为：$Z_0 = X_0 + E_{\text{pos}}$.

3. Transformer 编码器结构

ViT 使用标准的 Transformer 编码器层，对输入序列进行处理。每层包含两大模块，并在每个模块前后加入残差连接和层归一化（LayerNorm）。

3.1 多头自注意力机制（Multi-Head Self-Attention）

假设输入为$Z \in \mathbb{R}^{(N+1) \times D}$，我们将其映射为查询（$Q$）、键（$K$）、值（$V$）向量。对于第$h$个头（共$H$个头）：

$$Q^{(h)} = ZW_Q^{(h)}, \quad K^{(h)} = ZW_K^{(h)}, \quad V^{(h)} = ZW_V^{(h)},$$

其中
$W_Q^{(h)}, W_K^{(h)}, W_V^{(h)} \in \mathbb{R}^{D \times d}, \quad d = \frac{D}{H}.$

单头注意力计算：$[\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right) V]$

将所有头的输出拼接后，通过线性映射 $(W_O \in \mathbb{R}^{D \times D})$：$\text{MSA}(Z) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)\, W_O.$

加残差与层归一化：$Z' = \text{LayerNorm}\big(Z + \text{MSA}(Z)\big).$

3.2 前馈全连接网络（Feed-Forward Network）

FFN 包含两层全连接与非线性激活（GELU）：
$\mathrm{FFN}(z) = \mathrm{GELU}(zW_1 + b_1)\,W_2 + b_2,$
其中
$W_1 \in \mathbb{R}^{D\times D_{\mathrm{ff}}}, \quad W_2 \in \mathbb{R}^{D_{\mathrm{ff}}\times D}, \quad D_{\mathrm{ff}} = 4D.$

加残差与层归一化：
$Z_{\mathrm{out}} = \mathrm{LayerNorm}\big(Z' + \mathrm{FFN}(Z')\big).$

3.3 多层堆叠

将上述两步作为一层 Transformer 编码器，堆叠 L 层：
$Z_\ell = \mathrm{TransformerLayer}(Z_{\ell-1}), \quad \ell = 1, 2, \ldots, L,$ 最终得到Z_L。

4. 分类与输出

取最终输出序列中类别标记对应的向量 (z_0^{(L)})，通过全连接分类头预测：
$\hat{y} = \mathrm{softmax}\big(z_0^{(L)} W_{\mathrm{cls}} + b_{\mathrm{cls}}\big),$
其中
$W_{\mathrm{cls}} \in \mathbb{R}^{D\times C'}, \quad C' = \text{类别数}.$

5. 总结模型的数学流程

1. **Patch 划分与嵌入**
   $x_i = \mathrm{flatten}(\mathrm{patch}_i)\,E, \quad X = [x_1; \ldots; x_N] \in \mathbb{R}^{N\times D}.$
2. **类别标记与位置编码**
   $Z_0 = [x_{\mathrm{class}}; X] + E_{\mathrm{pos}}.$
3. **Transformer 编码器（(L) 层）**
   $Z' = \mathrm{LayerNorm}(Z + \mathrm{MSA}(Z)), \quad Z_{\mathrm{out}} = \mathrm{LayerNorm}(Z' + \mathrm{FFN}(Z')).$
4. **分类头**
   $\hat{y} = \mathrm{softmax}(z_0^{(L)} W_{\mathrm{cls}} + b_{\mathrm{cls}}).$

6. 细节讨论

- **位置编码**
  - 可学习或固定（正弦）编码。
  - 可学习编码允许在训练中自适应捕捉空间信息。
- **残差连接与层归一化**
  - 保证深层网络梯度稳定，提升收敛速度。
- **大规模预训练**
  - ViT 通常需在 ImageNet-21k、JFT-300M 等大规模数据上预训练，再在小数据集上微调。
- **与 CNN 比较**
  - CNN 强调局部特征；ViT 强调全局依赖。
  - 数据量不足时，ViT 容易过拟合；数据量大时性能优异。

7. 优势

ViT 将图像看作一系列 patch，通过线性投影和位置编码将其映射为序列，并利用 Transformer 的多头自注意力机制捕捉全局特征，最后通过类别标记进行分类。该方法将 NLP 中 Transformer 的成功经验引入视觉任务，为计算机视觉提供了全新的建模思路。

1. 全局信息捕捉

- **自注意力机制**
  ViT 利用 Transformer 中的自注意力机制，可以直接捕捉图像中各个 patch 之间的全局依赖关系。相比于 CNN 通过局部卷积核逐层堆叠获得全局感受野的方式，ViT 在每一层都能直接建立远距离的联系，对全局上下文信息的建模更加直接和高效。

2. 灵活的模型设计

- **更少的归纳偏差**
  CNN 内置的局部感受野和权值共享机制在一定程度上限制了模型的表达能力，而 ViT 减少了这些归纳偏差，使得模型更加灵活。这种灵活性使 ViT 能够更容易适应不同类型的视觉任务，尤其在大规模数据集上，模型能够学习到更丰富和多样的特征表示。
- **模块化设计**
  Transformer 的模块化结构使得模型可以轻松扩展和堆叠更多层，且各层之间具有统一的设计，这有助于在更深的网络中保持信息传递和梯度流动，从而支持更大规模的模型设计。

3. 预训练和迁移学习优势

- **大规模数据预训练的优势**
  ViT 在大规模数据集（如 ImageNet-21k、JFT-300M）上预训练时，可以充分发挥其建模全局信息的优势。预训练得到的模型在迁移到下游任务时，往往能达到比传统 CNN 更高的性能，尤其是在数据充足的情况下。
- **迁移学习的灵活性**
  由于 ViT 采用了类似 NLP 中 Transformer 的架构，研究者可以借鉴大量在 NLP 中验证有效的预训练与微调方法，使得模型在不同视觉任务上的适应性更强。

4. 可解释性与可视化

- **注意力机制的可解释性**

  自注意力机制使得每个输出 token 的计算可以追溯到输入序列中各个 patch 的贡献。这种机制为模型的决策过程提供了一定的可解释性，研究者可以通过可视化注意力权重来观察模型关注的区域，从而更好地理解模型的行为。

5. 并行计算和扩展性

- **高效的并行计算**
  Transformer 架构中的自注意力操作天生适合并行计算，在现代硬件（如 GPU、TPU）上能够高效实现。相比于 CNN 的逐层卷积操作，ViT 更容易利用大规模并行计算资源，从而加速训练和推理过程。
- **模型扩展性**
  由于 ViT 采用的是标准的 Transformer 架构，可以直接借助 Transformer 领域的发展（如改进的注意力机制、优化器策略等）来进一步提升模型性能。此外，模型的扩展（如加深、加宽）也更为直接，这为在不同硬件和任务场景下调整模型提供了便利。

从我个人观点，4与5对后续论文的提升很有帮助，多个模型的扩展与新一代显卡对transformer的更好的支持会在工程中很有意义