6 月 23 日~6 月 29 日（第一周）

1. Vision Transformer

1.1 核心思想

一张图片等价于 16×16 大小的单词

——Transformer 应用于大规模的图像识别

目标：作者有鉴于 Transformer 在自然语言处理（NLP）任务中取得的巨大成功，研究探索如何直接将 Transformer 架构直接应用于图像分类任务。

难点：图像输入序列过大导致计算成本过大。

已有方法有将 Transformer 替换卷积中的某些模块，或者在卷积操作后加入合并一起使用。例如一些特殊的注意力机制，轴注意力，孤注意力。Vision Transformer 摒弃传统卷积操作，将图像分割为固定大小的图像块（patches）视为序列输入 Transformer，以此减少序列长度。

1.2 结构框架

参考示意图 1-2，主要包含以下部分。

线性映射层：将输入的图像块矩阵展平为向量，进而得到一组序列:

$$\left[X_p^1 E; X_p^2 E; \ldots; X_p^3 E\right] \tag{1-1}$$

a) 在序列开头添加一个额外的可学习 **class token**，其最终状态作为最终输出

b) 添加位置编码

- 1-dimensional positional embedding（向量）
- 2-dimensional positional embedding（矩阵）
- Relative positional embedding（相对距离）

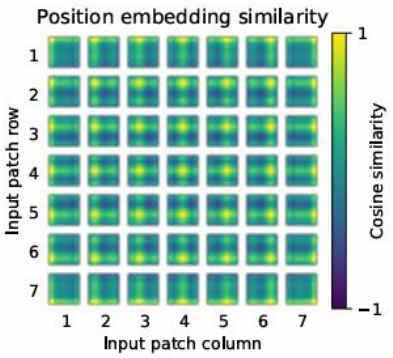


图 1-1 位置编码相似矩阵

从上图结果可知 1d 的位置编码已经可以很好的表示出位置信息。

经过以上两步操作得到最终的输入序列Z_0

$$Z_0 = \left[X_{class}; X_p^1 E; X_p^2 E; \ldots; X_p^3 E\right] + E_{pos} \tag{1-2}$$

Transformer Encoder：主要包含一个多头自注意力和 MLP

模型中使用多个 Encoder，假定使用L层，则有式 1-3 和 1-4

$$Z_l' = MSA\big(LN(Z_{l-1})\big) + Z_{l-1} \tag{1-3}$$

$$Z_l = MLP\big(LN(Z_l')\big) + Z_l' \tag{1-4}$$

其中均使用了残差连接，最后输出图像特征

$$y = LN(Z_L^0) \tag{1-5}$$

Question 1：为什么没有解码器？

Answer：不需要额外生成向量，仅需通过编码器获得输入特征进行判定即可

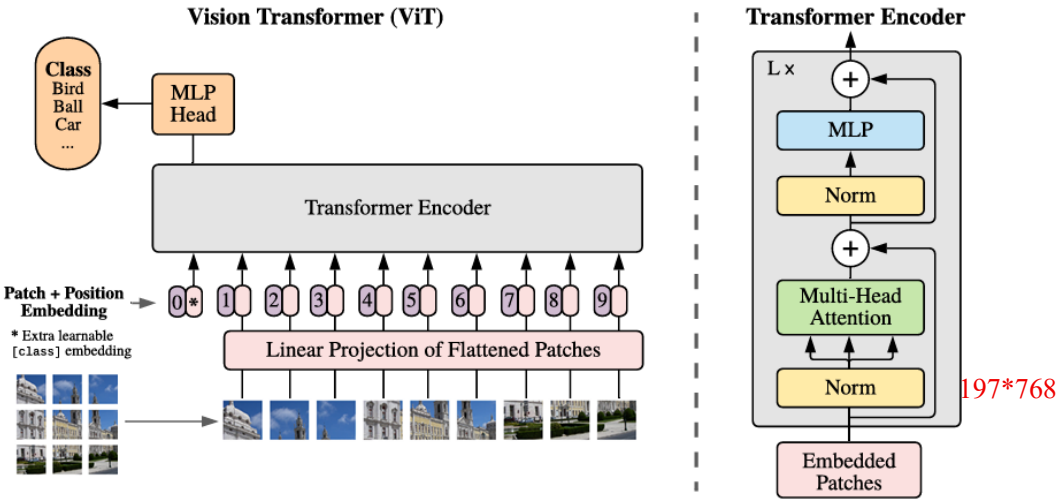


图 1-2 Vision Transformer 结构

Tips: layer norm 层，其核心思想是对单个样本的不同特征维度进行归一化，而非 Batch Norm 那样对批量样本的同一特征维度归一化（适应变长输入、稳定梯度）。

1.3 论文结论

作者使用在 JFT 数据集上预训练的 ViT 模型迁移到下游任务，性能表现要好于基于 ResNet 的 BiT 和基于 EfficientNet 的 Noisy Student，且预训练时间更少。

2．对比学习

2.1 一些对比学习模型

对比学习（Contrastive Learning）是一种自监督学习范式，核心思想是通过拉近相似样本（正样本对）的距离，推远不相似样本（负样本对）的距离，从而学习数据的有效表示。以下是其核心原理和关键组件的详细对比分析。

对比学习模型中主要的不同在于如何定义正负样本，难点在于字典过小或者特征一致性改变。

InstDisc（使用 memory bank 存储字典信息，随机抽取负样本）

SimCLR（端到端学习、负样本较少）

MoCoV2（MLP、数据增强、cos 学习率调度、增加训练轮次）

SimCLRV2（更大的模型、加深 projection head、动量编码器）

SwAV（通过聚类中心来比） 使用 multi crop 多采样提升效果

BYOL（不需要负样本，将判别任务改为预测任务）

SimSiam（简单的孪生网络，引入停止梯度（stop-gradient）来避免模型坍塌）

MoCoV3、DINO（引入 Vision Transformer）

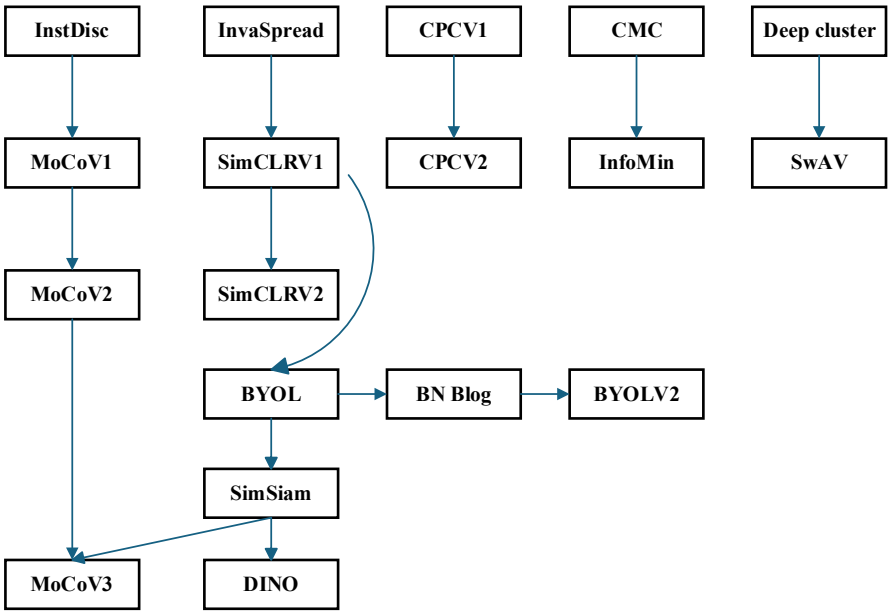


图 2-1 对比学习模型联系

2.2 MoCo（MomentumContrastiveLearning）

1) 输入处理:

对同一图像生成两个增强视图x_q和x_k，再分别经过不同的编码器提取特征。

2) 正负样本确定:

将正负样本类比为字典查询问题。

数据经过键编码器后统一储存在一个队列中，一般队列的第一个是正样本x_k的编码结果。这里巧妙的用队列维护一个字典。在字典很大的情况下，如果同时更新的话，计算代价大。于是改为用很多的 mini batch 逐次更新，先进先出，保证字典容量的同时减小计算成本。

3) 参数更新方式:

动量编码器（理解成惯性）

为了保持队列中特征的一致性（避免快速变化的特征破坏对比学习），MoCo 使用动量编码器生成队列特征:

$$\theta_k \leftarrow m\theta_k + (1-m)\theta_q \tag{2-1}$$

其中θ_k和θ_q分别为键编码器和查询编码器的参数

4) 损失问题:

作者使用的是 InfoNCE loss

$$L_{InfoNCE} = -\log\frac{\exp\left(sim(q,k^{+})/\tau\right)}{\sum_{i=0}^{K}\exp\left(sim(q,k_i)/\tau\right)} \quad (2\text{-}2)$$

其中τ为温度系数，用于调节损失的尖锐程度

Question 2: 为什么伪代码中仍然使用交叉熵?

Answer 2:

CrossEntropyLoss 实际上是 LogSoftmax+NLLLoss 的组合。它会先对 logits（相似度分数）计算 softmax，然后最大化正确类别的概率（这里是正样本的相似度）。

结合算法 1，则其计算公式为

$$L_{Cross} = -\log\frac{\exp\left(l\_pos\right)}{\exp\left(l_{pos}\right)+\sum_{i=0}^{K}\exp\left(l\_neg_i\right)} \quad (2\text{-}3)$$

用这样的方式方便直接使用内置函数来计算损失 CrossEntropyLoss，比手动实现更高效，且数值稳定性更好。

表 4-1 数据集中各病害样本数量

算法 1	**MoCo 伪代码**
1.	***f_k.params = f_q.params***
2.	***for x in loader***
3.	***x_q = aug(x)***
4.	***x_k = aug(x)***
5.	***q = f_q.forward(x_q)***
6.	***k = f_k.forward(x_k)***
7.	***k = k.detach()****#不自动更新参数*
8.	***l_pos = bmm(q.view(N,1,C),k.view(N,C,1))****#正样本距离*
9.	***l_neg = mm(q.view(N,C),queue.view(C,K))****#负样本距离*
10.	***logits = cat([l_pos,l_neg],dim=1)***
11.	***labels = zeros(N)***
12.	***loss = CrossEntropyLoss(logits/t,lables)***
13.	***loss.backward()***
14.	***update（f_q.params）***
15.	***f_k.params = m*f_k.params+(1-m)*f_q.params***
16.	***enqueue（queue,k）****#更新字典*
17.	***dequeue(queue)***

3．CLIP（Contrastive Language-Image Pre-Training）

3.1 核心思想

CLIP（Contrastive Language-Image Pre-Training）模型是一种多模态预训练神经网络，由

OpenAI 在 2021 年发布，是从自然语言监督中学习的一种有效且可扩展的方法。

该模型的核心思想是使用大量图像和文本的配对数据进行预训练，以学习图像和文本之间的对应关系。

3.2 模型架构

CLIP 模型有两个模态，一个是文本模态，一个是视觉模态，包括两个主要部分：

1. Text Encoder：用于将文本转换为低维向量表示-Embeding。

2. Image Encoder：用于将图像转换为类似的向量表示-Embedding。

目标在于直接从文本中获得监督信号，以此随心所欲的找到自己想要的类，逃脱固定的输出类别设定并获得良好的迁移效果。

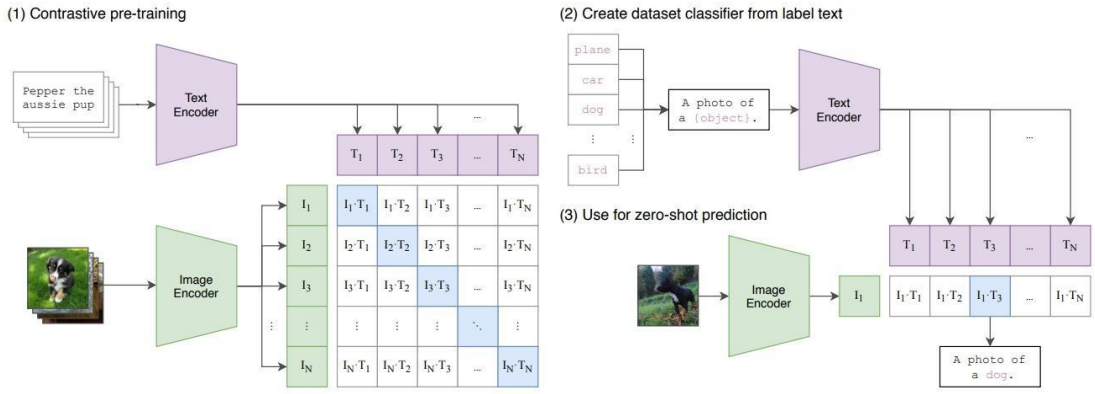


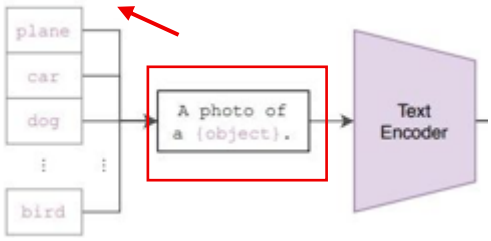
图 3-1 CLIP 结构

相似度计算：CLIP 计算图像特征和文本特征之间的余弦相似度（cosine similarity）

对于图像I_i和文本T_i，对角线上的元素均为正样本，即S_{ii}，其余部分为负样本。

利用正负样本进行 InfoNCE 损失计算，实际上这里是对称的算 loss，最后取平均值。

prompt engineering and prompt ensembling



上图是作者进行的创新，加入了 prompt，这里可以理解为提示，主要想利用一些先验知

识帮助模型更好的做出判决。图片中的示例是{A photo of a(object)}，那么实际作者ensembling了80个类似的词条供模型使用，推理时可取它们的平均输出。

作者构建了一个4亿个图像-文本对的数据集WIT数据集来进行预训练，以此保证CLIP强大的迁移学习能力。

3.3 实验结果

说到Zero-shot（零样本学习）是指模型在没有针对特定任务进行任何专门训练（即零训练样本）的情况下，直接完成该任务的能力，作者对这一点比较看重。实验结果表明，CLIP模型在跨数据集迁移任务中展现出较强的泛化能力，尤其在常规物体分类任务上表现优异。在面对细粒度纹理识别、图像中物体数量统计以及医学影像中的肿瘤分类等场景等更具挑战性的视觉任务时，该模型的性能则出现明显下降。可以看出预训练不是万能的，某些领域仍然要求模型要一定的先验知识。

同时作者还花大量的篇幅讲了Few-shot以及使用全部训练数据时的下游任务效果，均表现出不俗的性能，这里不在赘述。

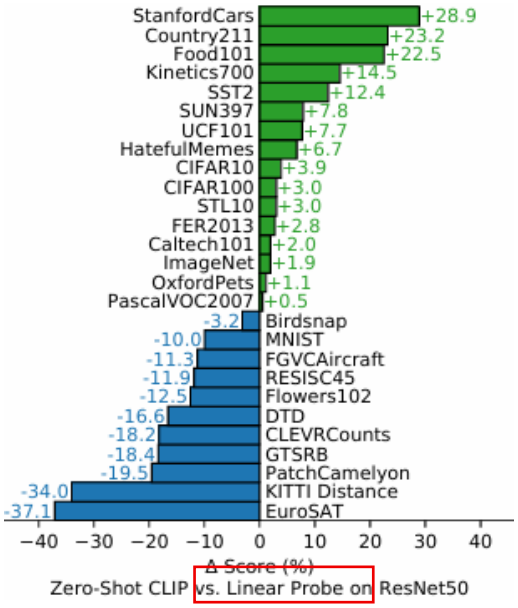


图3-2 CLIP在Zero-shot下的各数据集效果对比（与ResNet50）

Tips：作者使用Linear Probe，仅仅训练线性分类层，以此来直接评判CLIP的特征提取能力

3.4 一些限制和不足

- CLIP在真正意义上的跳出预训练范围上的数据集中性能表现不好，比如在MINIST中。
- 能否让CLIP还是在跳出有限的类别中进行对比、推理，直接的生成新的概念，实现完全的自动化。
- 在训练和挑选CLIP模型以及研发过程都与文中用27个数据集息息相关。无形之中已经带入偏见了或者信息泄露。
- 同时在视觉任务或者概念很难用text来表达，在计数、意义等抽象的文字描述下CLIP的表现并不好。