实验四：**Transformer**模型应用

by Sun Jan 10

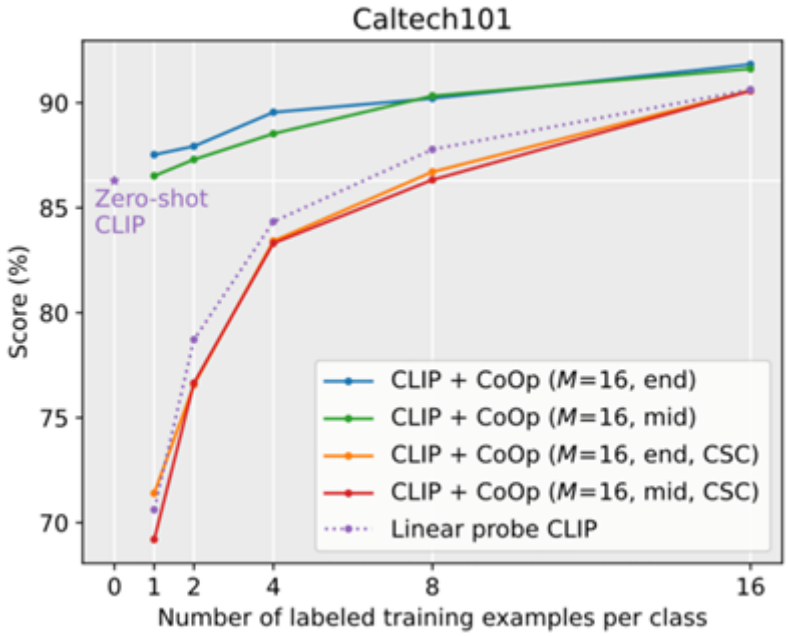
闯关弟子注意，本关考察你代码阅读与模型实现功夫

实验任务

1. 用Transformer模型(预训练的CLIP)实现对图像的zero-shot分类。
2. 将固定的提示词（prompt）依据《CoOp》中的方式替换为可学习的text embedding，再次对图像进行分类。

实验要求

1. 首先用原始的CLIP在celtech-101上进行分类，自己划分训练集、验证集、测试集。注：本次实验使用的CLIP的具体型号为“RN50”，即视觉模型为ResNet50的CLIP
2. 报告需介绍CLIP的代码运行原理，重点体现CLIP中的transformer结构及其作用，以及CLIP同时用视觉和语言模态进行分类的逻辑。
3. 在CLIP官方代码的基础上改成CoOp的prompt，在Caltech101上进行图片分类
   - M=16 (训练16个embedding)
   - end (class token位于可训练embedding后面)
   - 建议分别尝试1, 2, 4 shot
   - 其他设置不必与原论文保持一致
   - 预计需要约5.8G显存
4. 比较zero-shot CLIP与CoOp在Caltech101测试集(自己划分)上的性能，画出类似下图中的紫色五角星与蓝色折线，每个点都应是选取3个不同随机种子的实验结果的均值，具体如何设置随机种子，请参看下方文档。



5. 加分项，非必做：提出一种对transformer block的修改方法，注意仅需“修改方法”，只要性能不严重下降，修改有道理均可，期待大家自由发挥。

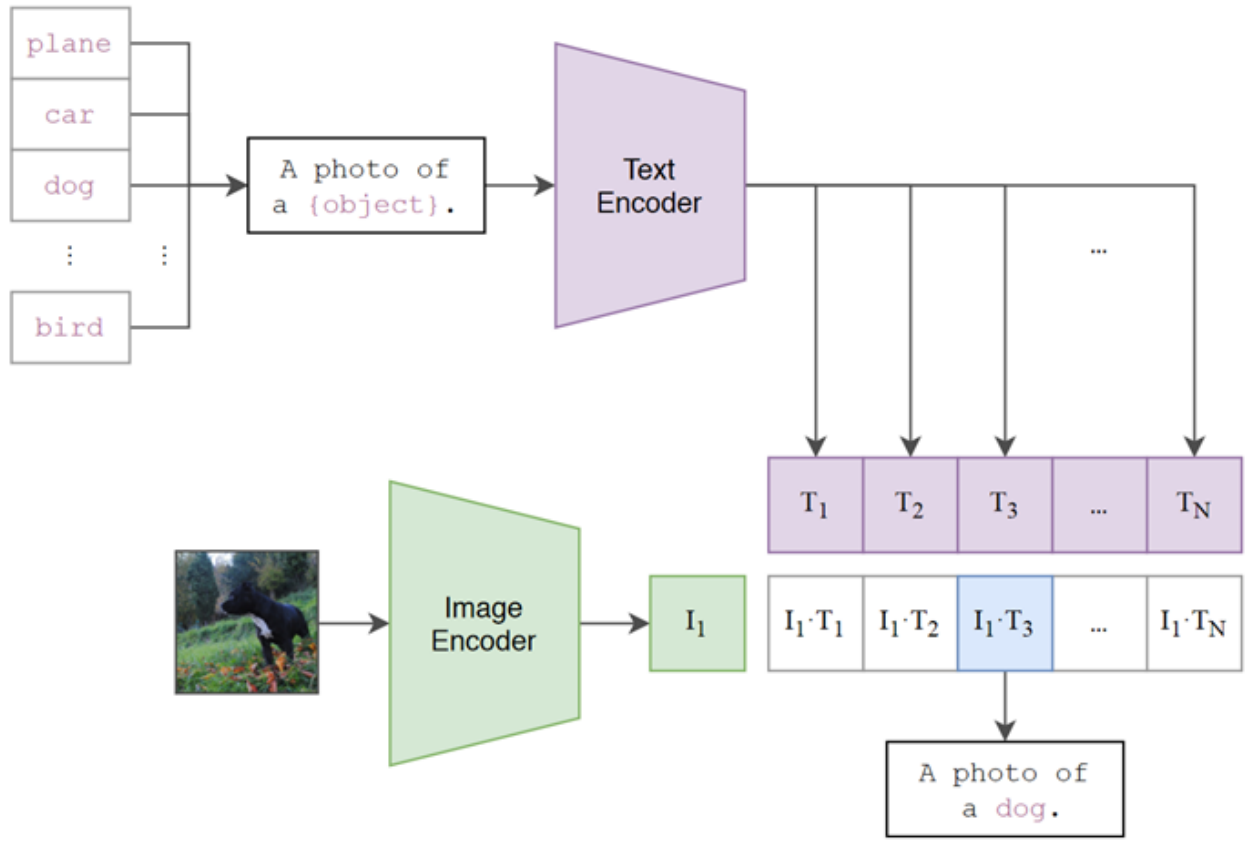
杂项

1. CLIP的官方代码库为https://github.com/openai/CLIP
2. CoOp的原paper链接为2109.01134，可以参看它learnable prompt部分的代码实现
3. 在已经安装好的pytorch环境中运行以下命令
   pip install ftfy regex tqdm # 安装CLIP依赖的包
   git clone https://github.com/openai/CLIP # 下载CLIP的代码
4. 本次实验可以通过更改./CLIP中的代码完成
5. 种子的设置例子：

```
seed = 42
random.seed(seed)
torch.manual_seed(seed)
if torch.cuda.is_available():
    torch.cuda.manual_seed(seed)
    torch.cuda.manual_seed_all(seed)
np.random.seed(seed)
torch.backends.cudnn.benchmark = False
torch.backends.cudnn.deterministic = True
```

Transformer与CLIP的知识

CLIP



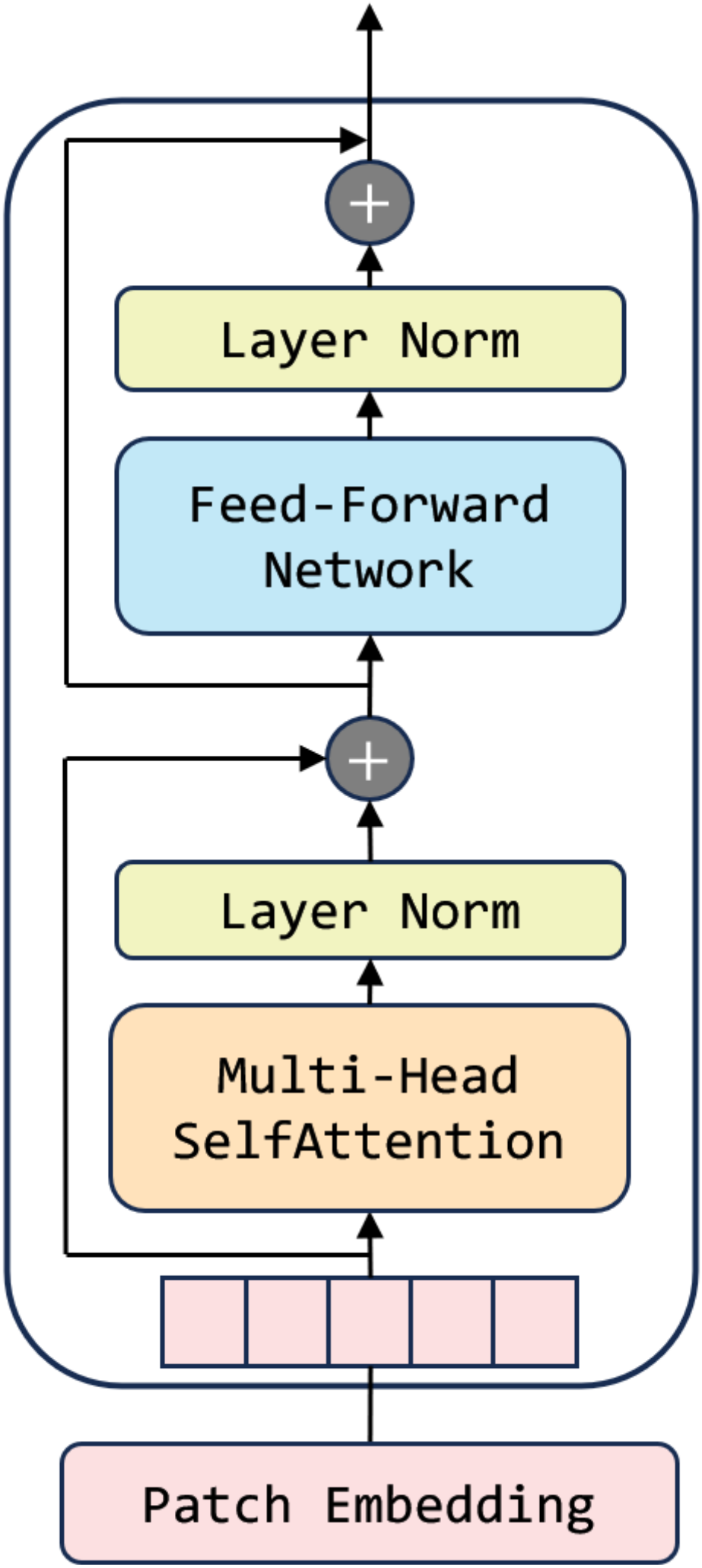
CLIP通过称为contrastive learning的方式对图像进行分类，因为引入了自然语言信息，分类的鲁棒性大幅提升，即使在2024年，也是最好用的多模态Transformer模型之一。

具体的实现原理我这里不以文字形式详解，以免直接被用在实验报告中。

基于**transfomer**的**encoder**设计

Transformer 是一种通过堆叠自注意力机制来构建的深度学习模型，可以有效地处理长距离依赖或是时序信息。它由多个Transformer 块堆叠而成，通过这些块的组合，文本或图像包含的信息被提取为包含其意义的特征。

Transformer 块由两个主要模块连接而成。首先是多头自注意力（multi-head Self-attention layer，以下简称为MHSA）层，后面连接着前馈神经网络（Feed-Forward Neural Network，以下简称为FFN）。


Layer Norm
Feed-Forward Network
Layer Norm
Multi-Head SelfAttention
Patch Embedding

多头自注意力机制基于自注意力机制（Self-attenetion，以下简称为SA）构建，自注意力网络接收一个词或图片向量$X:X1,X2,...,Xn$作为输入，拥有三个不同的权重矩阵WQ、WK、WV 用于对输入的向量进行处理。

代表查询（Query) 的矩阵Q, 代表关键词（Key）的矩阵K, 代表权值（Value）的矩阵V 三个矩阵均来自输入X 与权重矩阵的乘积，其中

$Q = X \times WQ$

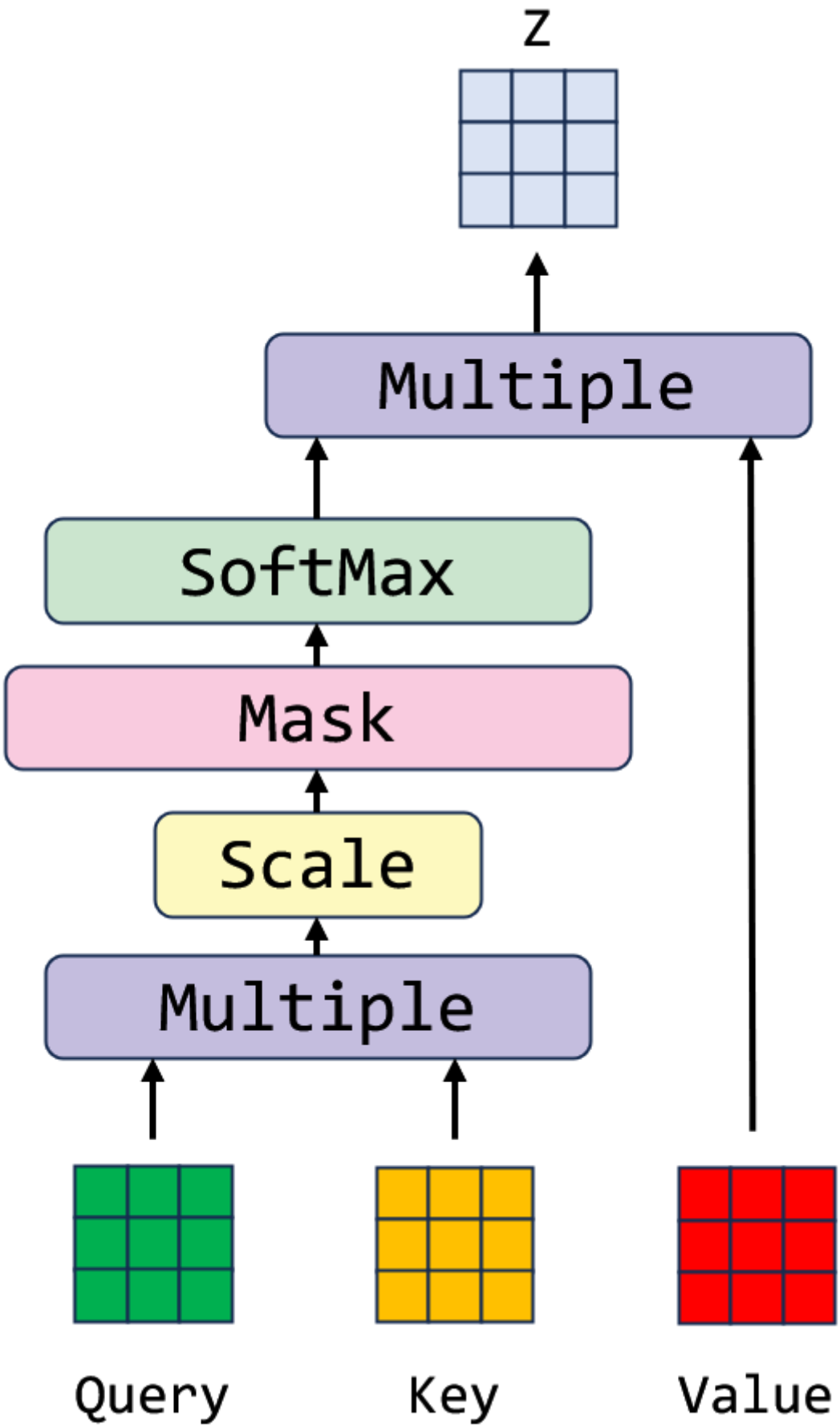
$K = X \times WK$

$V = X \times WV$

由此，输入中每个位置的attention score 定义为如下形式：

$$S = \frac{Q \times K^T}{\sqrt{d^k}}$$

其中$\sqrt{d_k}$ 为一个query 和key 向量的维度,S 为attention score。将结果除以dk 是为了防止注意力结果在某些维度上过大，相当于一次正则化操作。再将其结果通过Softmax 层归一化为概率分布，由此概率分布乘以权值矩阵V，即可得特征矩阵。此操作可以表示如下图所示：


Z
Multiple
SoftMax
Mask
Scale
Multiple
Query
Key
Value