**模式识别与机器学习 实验报告**

**Lab4 Transformer模型的应用**

1. **CLIP图片分类代码逻辑**

CLIP的代码库主要由以下几个部分组成：

1.clip.py：这是库的主要文件，包含了CLIP模型的实现。它定义了CLIP类，该类包含了模型的主要部分，包括视觉和文本转换器，以及用于从这些转换器生成图像和文本特征的方法。此外，它还包含了一个load函数，用于加载预训练的模型。

2.model.py：这个文件包含了构成CLIP模型的各个组件的实现，包括视觉和文本转换器，以及它们的各个子层。

3.simple\_tokenizer.py：这个文件包含了一个简单的分词器，用于将文本转换为模型可以理解的形式。

4.clip\_tokenizer.py：这个文件包含了一个更复杂的分词器，它使用了Byte-Pair Encoding (BPE) 算法。

在进行零样本图片分类时，CLIP的工作流程如下：

首先，使用clip.load函数加载预训练的模型和分词器。

然后，将类别标签（如"狗"，"猫"等）和待分类的图片输入到模型中。类别标签被分词器转换为模型可以理解的形式，图片则直接输入到模型中。

模型会为每个类别标签和图片生成一个特征向量。这些特征向量然后被用来计算图片和每个类别标签之间的相似度。

最后，模型会选择相似度最高的类别标签作为图片的预测类别。

这种方法的关键在于，CLIP模型能够理解类别标签和图片之间的语义关系，从而能够在没有看到任何类别的训练样本的情况下，正确地将图片分类到正确的类别。这就是所谓的"零样本"分类。

1. **数据集处理**

由于本实验使用的训练数据较少，将数据集按1：4的大小进行了划分。



对于后文需要训练的CoOp，分别根据opt参数中的shot搭建n-shot训练集：从训练数据中按随机种子抽取每个类n张图放入data中用于训练。





1. **Clip zero-shot测试**

这段代码的主要流程如下：

1.加载CLIP模型：使用clip.load函数加载预训练的CLIP模型。

2.定义图像预处理：定义一个图像预处理流程，包括将灰度图像转换为RGB图像，调整图像大小，并将图像转换为张量。

3.加载数据集：加载Caltech101数据集，获取所有图像的路径和对应的类别标签。

4.划分数据集：使用train\_test\_split函数将数据集划分为训练集和测试集。

5.生成类别描述：对于测试集中的每个类别，生成一个描述（例如，对于类别"狗"，生成描述"A photo of a dog"）。

6.提取类别描述的特征：使用CLIP模型的encode\_text方法提取类别描述的特征。

7.分类测试集中的图片：对于测试集中的每张图片，先进行预处理并提取特征，然后计算图片特征和每个类别描述特征之间的相似性，选择相似性最高的类别作为预测结果。

8.计算正确率：比较预测结果和真实类别，计算总的正确率。

这段代码的主要是使用CLIP模型进行零样本图片分类。在零样本分类中，模型需要根据类别的语义描述来对图片进行分类，而不是依赖于类别的训练样本。这是一种非常强大的方法，可以应用于许多不同的任务和领域。





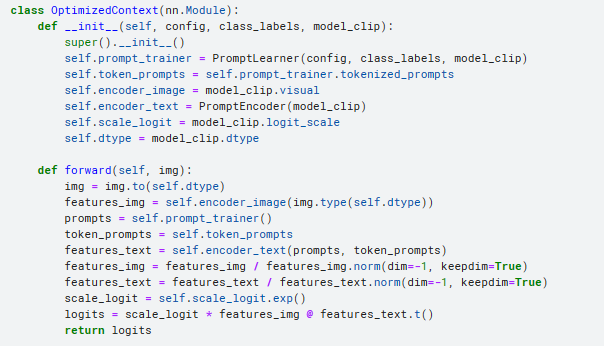
选择随机数0，233，666做三次测试，得到结果如下图：



平均测试结果为0.738%

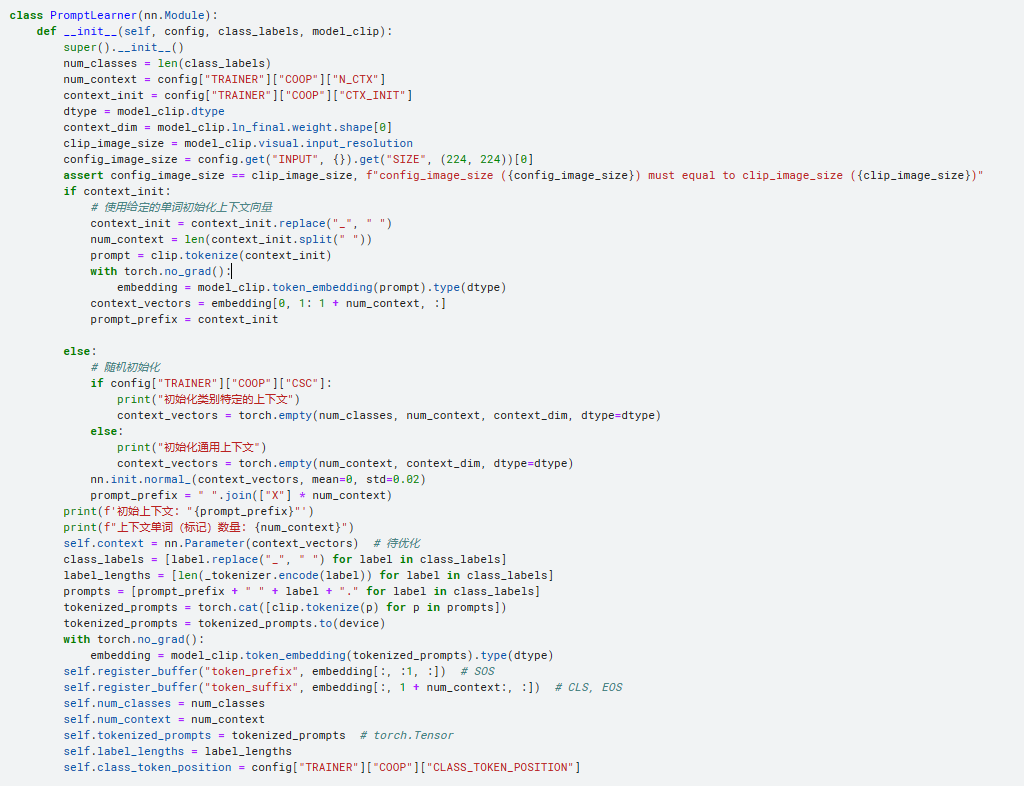
1. **CoOp实现**

基于clip实现可训练的promptlearner。首先改写原clip的textencoder：



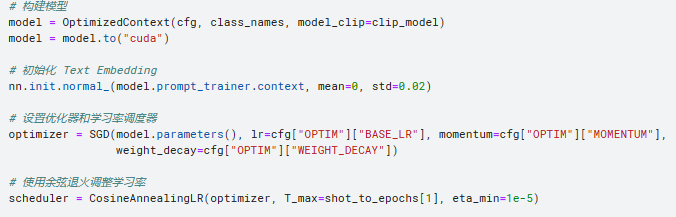
如图可知，在前向过程中，将已训练的prompt与位置编码进行加和，其他参数维持clip原样不做修改即可。

下图为PromptLearner的实现，根据opt将ctx大小设置为16，声明self.ctx为可学习的参数，并使用高斯分布初始化。



1. **CoOp训练**

搭建数据集、加载clip模型后，如下图初始化CoOp的encoder和learner。设置优化器仅优化promptlearner的参数，使用余弦退火调整学习率，损失函数使用交叉熵。



训练过程中，将学习率设置为0.002，其它参数如图

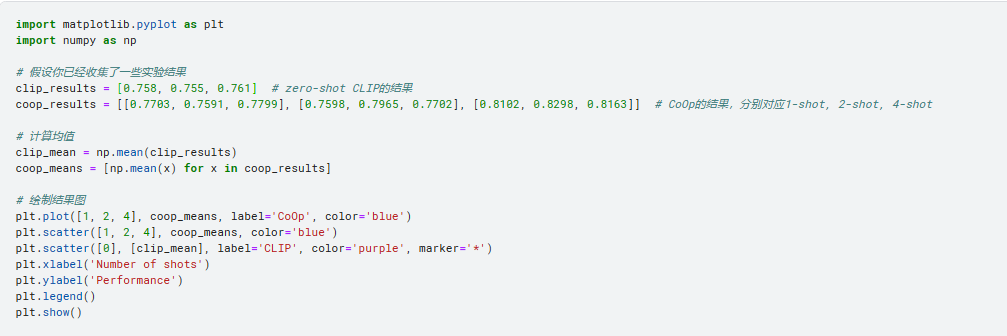


对于每个n-shot任务，选取0，233，666三个随机数进行训练得到的测试集结果.



1. **测试结果**

将训练的模型加载并在测试集上进行测试，随机数选择同上。



最终结果为：

（1）1shot：avg=76.98%

（2）2shot：avg=77.55%

（3）4shot：avg=81.88%

将上述正确率求取平均后得到每组的avg\_score，使用matplotlib进行画图得到下图结果：

