图像风格迁移实验手册

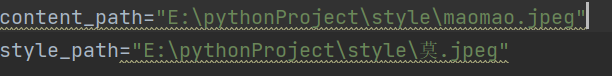
1. 导入实验所需要库



1. 图片预处理

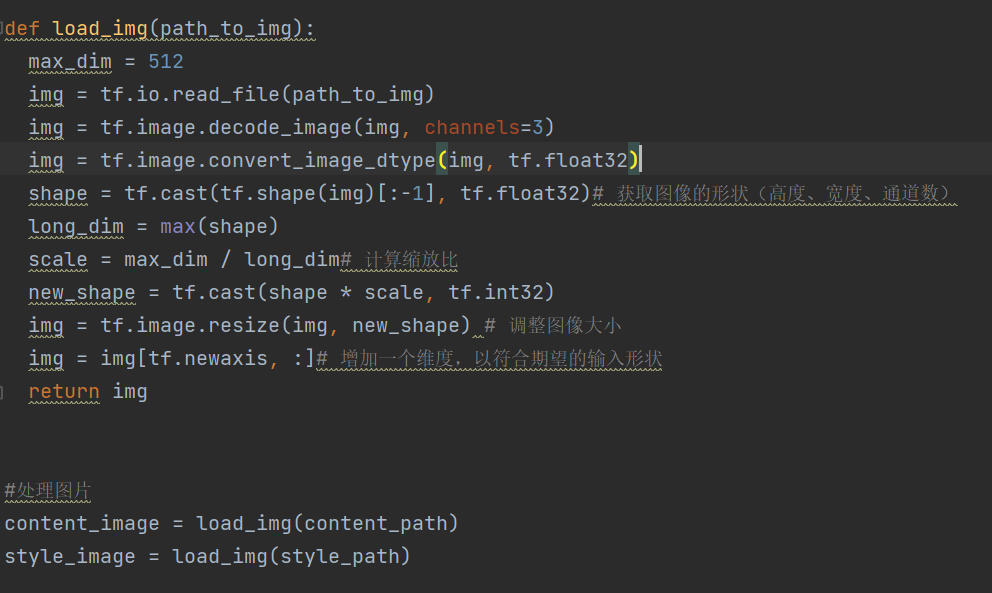
2.1 下载图片

首先在网上下好需要迁移风格的图像以及风格样式图片，并将地址赋值给content\_path和style\_path。



2.2 图片预处理

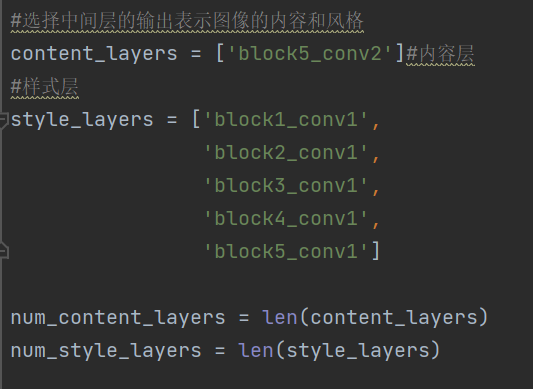
在将图像放入神经网络之前，需要对大小进行预处理，同时还需要增加一个维度以符合期望的输入。



3 建立模型

3.1 设置内容层和样式层

选择中间层的输出分别来表示图像的内容与样式，分别设置为内容层和样式层。



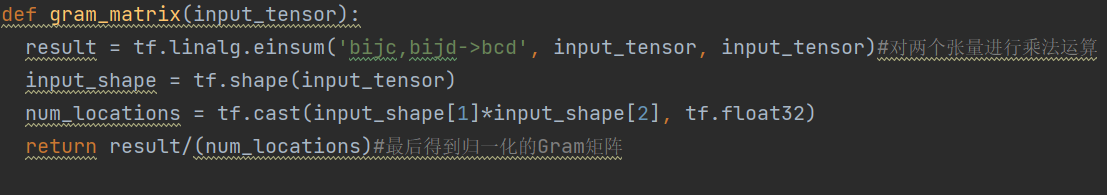
3.2 建立VGG模型

这里我们直接调用已经预训练好的模型VGG作为本次实验的模型，之后对于每个图像的样式层和内容层，分别分出一个模型。之后使用vgg\_layers(style\_layers)就可以提取一个图片通过神经网络获取的样式层的特征。在这里，我们首先先提取作为本次实验的风格样式图片的样式特征。



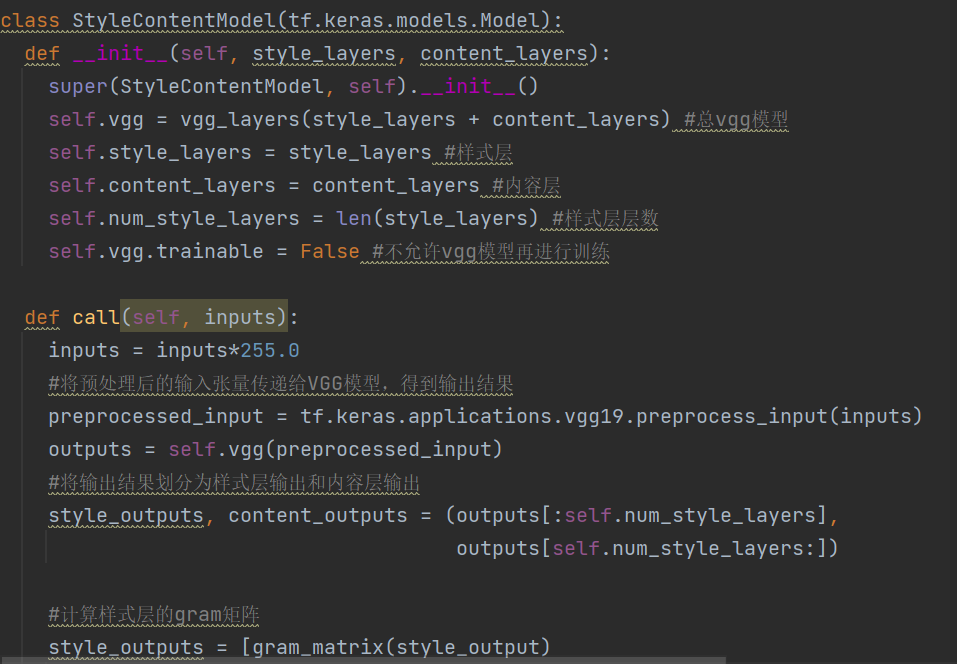
3.3 设置计算gram矩阵函数

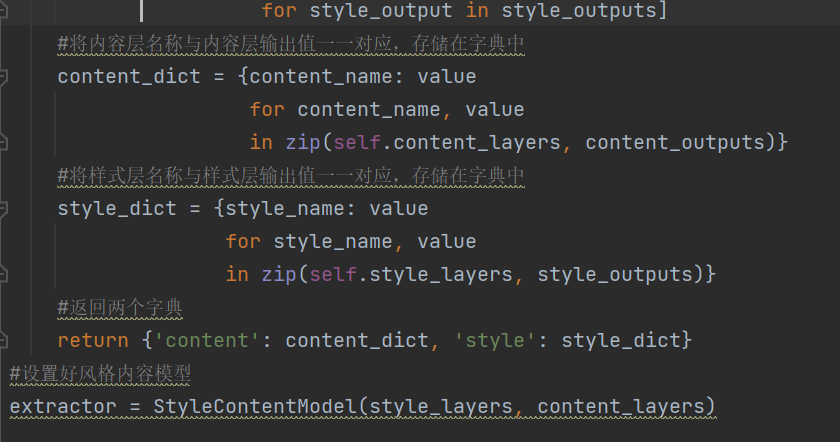
Gram矩阵可以很好地反应不同特征之间的组合，因此在本次实验中选用gram矩阵来反应两张图片的风格的相似度。因此这次实验中我们需要让原图像的风格特征向量算出来的gram矩阵更接近风格图片的特征向量对应的gram矩阵，下图便是gram矩阵的计算方法：



3.4 建立风格内容模型

之后我们需要建立一个模型，获取风格样式特征向量对应的gram矩阵以及内容层与样式层的输出。其中返回的字典就分别对应了图片内容层的输出和样式层的输出。

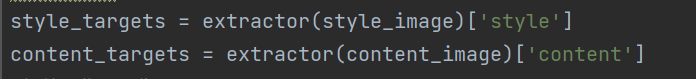




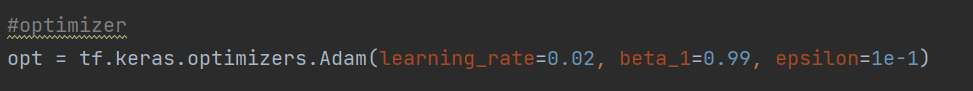
这样我们的基本模型就已经建立完成了，接下来就是训练的过程了。

1. 训练
   1. 参数设置

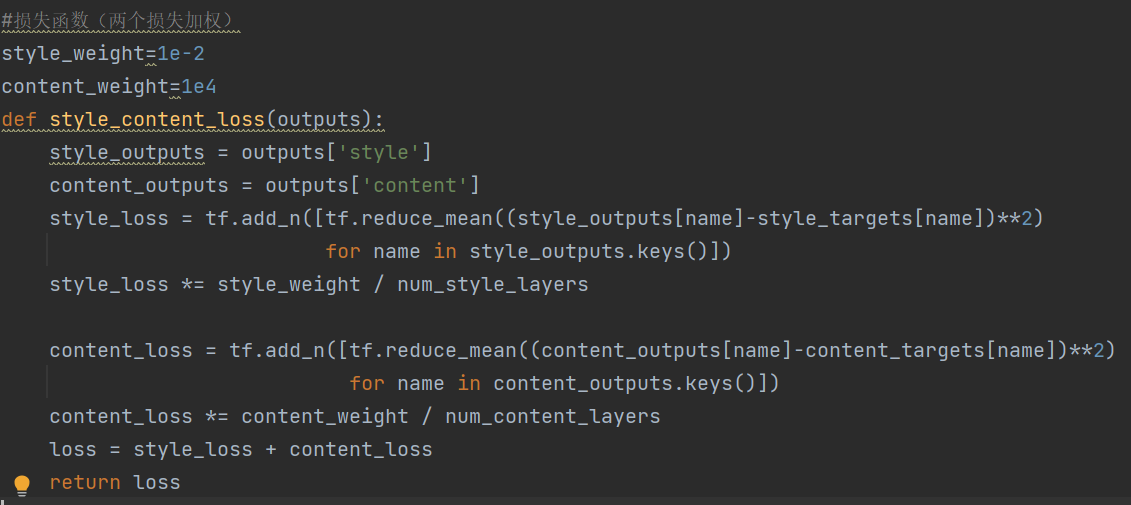
在本次实验中，我们需要完成图像风格迁移，因此我们希望让原图像的样式层输出的gram矩阵与风格图像样式层输出的gram矩阵尽可能相似，因此第一步我们需要设置本次训练的目标，希望最终输出图片的内容尽可能与原图像一致，而风格与风格图像尽可能一致。



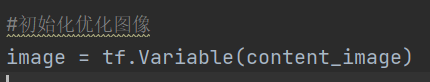
接下来设置梯度下降过程中所用到的优化器。



设置梯度下降过程中的风格内容损失函数，由于我们既要保证最终输出图片的内容尽可能与原图像一致，又要让风格与风格图像尽可能一致，因此在损失函数中我们设置为两个损失加权求和，由于我们需要保证图片的内容不能有大的改变，因此图片内容方面的loss权重需要设置尽可能大防止最终形成的图像与原图像完全不一样。



最后初始化优化图像，基本参数设置就完成了。

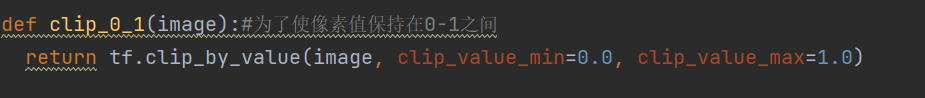


* 1. 训练

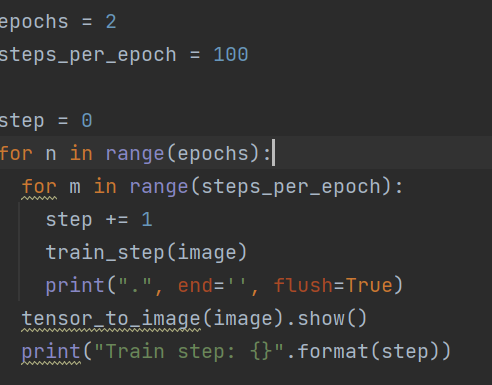
由于我们希望我们与原图像大体基本一致，因此我们不希望图像变化过大，因此最终的损失函数不仅包括了上面提到的风格内容损失，还加上了图像总变化，这样就可以防止图像变化过大。当然还需要加上之前设置的优化器，并且在训练的每一步最后需要让图像的像素值保证在0-1之间方便训练。



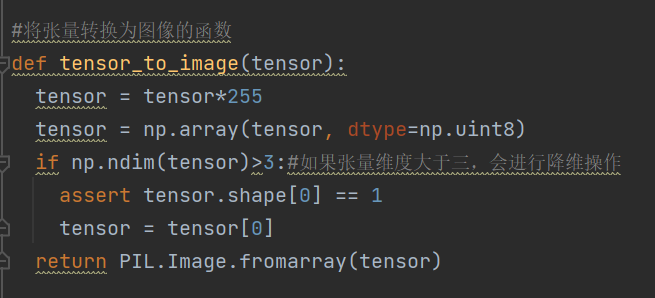
保证像素值在0-1之间的函数



最后设置epoch和每个epoch中的step就正式开始训练了。



在这里需要注意由于我们得到的是tensor类型，因此想要看到图像还需要把它转换成图像类型，函数如下图所示：



这样我们就可以看到每个epoch后的图像类型了。以上就是图像风格迁移实验。