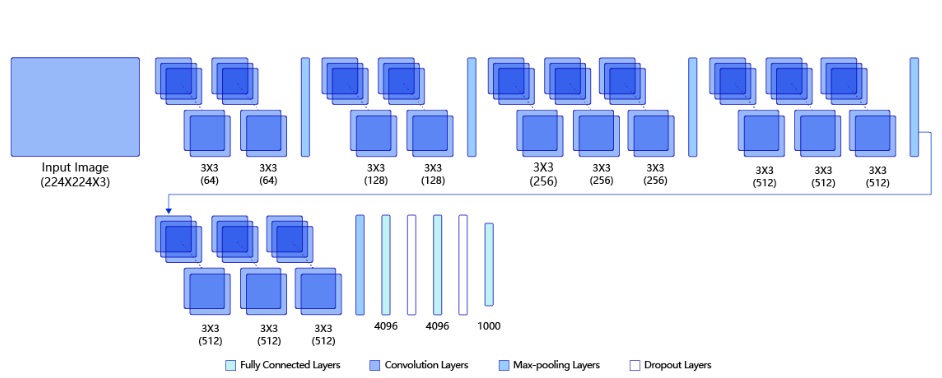
**构建VGG网络模型实现手写字体识别**

**一、VGG模型**

VGG的Classification模型从原理上并没有与传统的CNN模型有太大不同。大家所用的Pipeline也都是：训练时候：各种数据Augmentation（剪裁，不同大小，调亮度，饱和度，对比度，偏色），剪裁送入CNN模型，Softmax，Backprop。测试时候：尽量把测试数据又各种Augmenting（剪裁，不同大小），把测试数据各种Augmenting后在训练的不同模型上的结果再继续Averaging出最后的结果。



特点：小卷积核：将卷积核全部替换为3x3（极少用了1x1）；小池化核：相比AlexNet的3x3的池化核，VGG全部为2x2的池化核；层数更深特征图更宽：基于前两点外，由于卷积核专注于扩大通道数、池化专注于缩小宽和高，使得模型架构上更深更宽的同时，计算量的增加放缓；全连接转卷积：网络测试阶段将训练阶段的三个全连接替换为三个卷积，测试重用训练时的参数，使得测试得到的全卷积网络因为没有全连接的限制，因而可以接收任意宽或高为的输入。

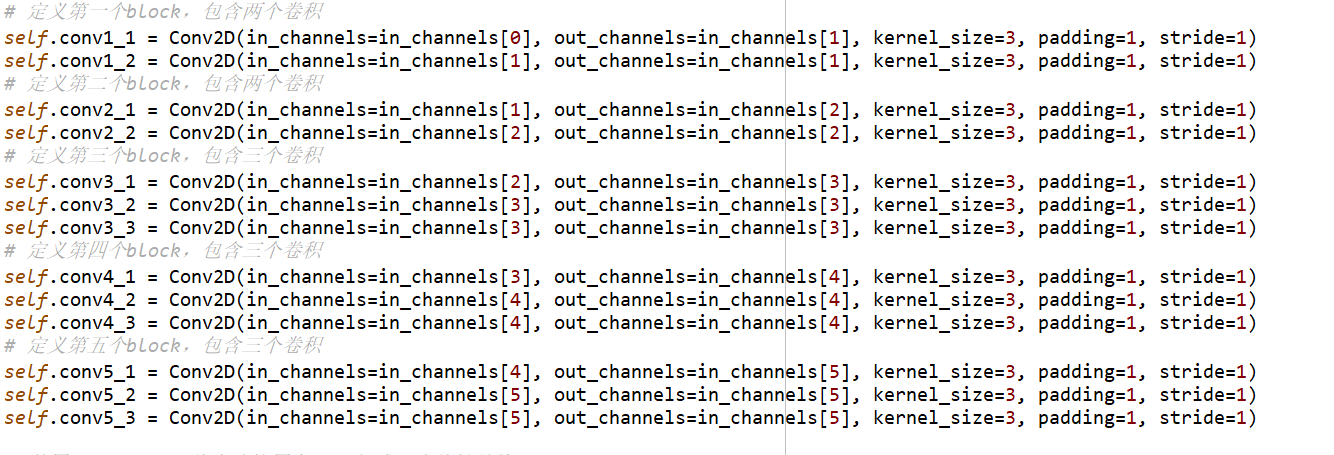
**二、数据预处理**

使用的数据集：MNIST数据集，一个开源的手写数字数据库，它一共有7万张图片。其中，6万张用于训练神经网络，1万张用于测试神经网络，每张图片都是28\*28像素点的。则定义图像的总尺寸为28\*28，标签的种类数，训练的总周期，一个批次的大小。则其包含60000个训练集和10000测试数据集，分为图片和标签，图片是28\*28的像素矩阵，标签为0~9共10个数字。

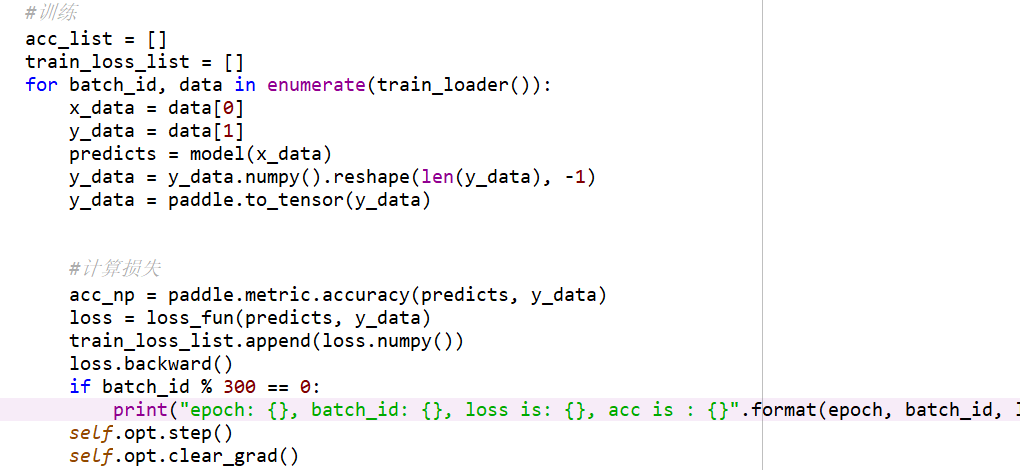
先导入需要用到的库，再加载数据，对数据进行预处理。做数据增强处理，调整饱和度、对比度、亮度和色调，亮度调整范围：[max(0，1 - brightness), 1 + brightness]，对比度调整范围大小：[max(0，1 - contrast), 1 + contrast]，饱和度调整范围大小：[max(0，1 - saturation), 1 + saturation]，色调调整范围大小：[-hue, hue]，再对数据进行膨胀和腐蚀操作。

**三、训练VGG网络模型**

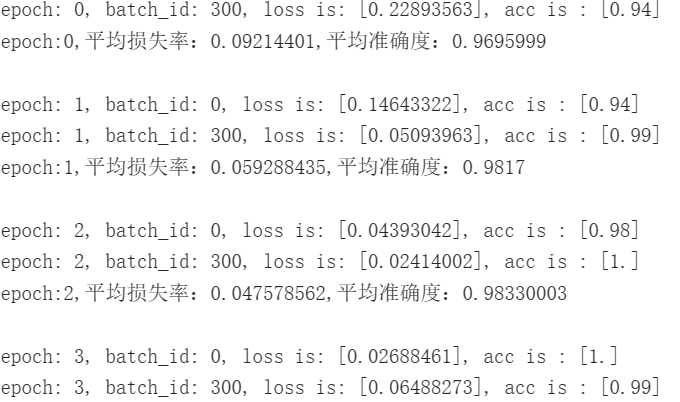
定义五个block，第一个block包含两个卷积、第二个block包含两个卷积、第三个block包含三个卷积、第四个block包含三个卷积、第五个block包含三个卷积，使用Sequential 将全连接层和relu组成一个线性结构，使得当输入为224x224时，经过五个卷积块和池化层后，特征维度变为[512x7x7]。



下面开始训练模型，定义损失函数，损失函数可以帮助我们衡量模型的，模型预测的好坏，用来表现预测与实际数据的差距程度。



最终结果会表达准确率和损失图。



**四、拓展——用此模型识别纸上手写的数字**

首先要进行数据图像预处理，手机拍到的照片和MNIST数据集中的图片会有很大的差别，是不能直接把我们手机上拍到的照片直接让训练好的网络模型进行识别的，所以需要对数据图像进行预处理。

图像大小的转换：MNIST数据集中的图像是28\*28的，要将拍摄到的照片转换成28\*28大小的；MNIST数据集中的图像是灰度图，所以需要将照片灰度化；为了使识别结果准确性更高，可以用阈值控制的方法把拍摄的照片的背景转换成黑色，照片中数字的颜色转换成白色，也可以用边缘检测把数字包裹住，将数字之外的背景清零（没有实现），如果将数字部分的值都设置为255的话，识别的效果应该不理想，这里，想到，可以采用255-原本数字的灰度值，这样，应该就实现了数字边缘浅、中间深的变换。

对照片中的手写字体进行识别，加载训练好的网络模型，得到待识别的图片，放在网络模型中进行预测，最后，打印对应的预测结果。