实验主题： 基于卷积神经网络的图像分类

# 实验目的

图像分类，顾名思义，是一个输入图像，输出对该图像内容分类的描述的问题。本实验需要设计一个类似AlexNet的卷积神经网络，并在CIFAR-10数据集上训练，以达到图像分类的目的。

# 实验环境与数据准备

## 实验环境

* + 1. 硬件环境

PC机一台

* + 1. 软件环境

Anaconda 3; Python 3.7; PyTorch 1.5; TensorBoard 1.14

## 数据准备

* + 1. 数据下载

数据地址：

官方（建议迅雷）：<https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz>

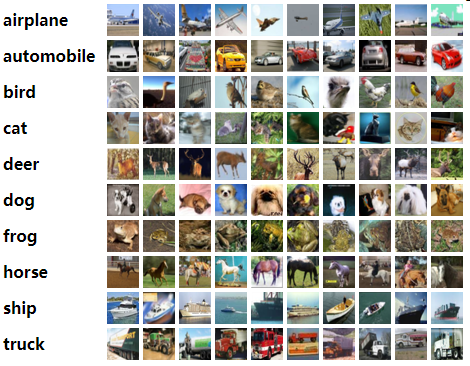
百度网盘：

链接：<https://pan.baidu.com/s/19vfeFfb_vcYm5rnyF2xS8Q>

提取码：tu61

* + 1. 数据介绍

CIFAR-10是由Hinton的学生Alex Krizhevsky和Ilya Sutskever 整理的一个用于识别普适物体的小型数据集。共包含 10 个类别的 RGB 彩色图片：飞机（airplane）、汽车（automobile）、鸟类（bird）、猫（cat）、鹿（deer）、狗（dog）、蛙类（frog）、马（horse）、船（ship）和卡车（truck）。图片的尺寸为 32\*32，数据集中一共有50000张训练图片和10000张测试图片。图片实例如下图所示：



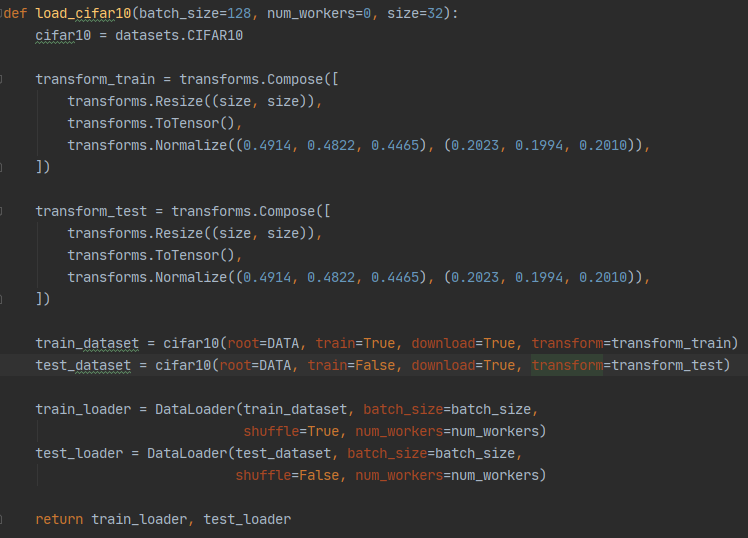
# 实验操作步骤（占60%）

## 相关准备

下载并解压工程：<https://github.com/AuMgLi/exp1_image_classification>

## 数据集加载及数据预处理

1. 在工程根目录（./）下新建文件夹data，即./data/，将下载好的数据集文件cifar-10-python.tar.gz移动到该文件夹中。
2. 在datasets.py中加载数据集并进行数据预处理：



## 搭建模型

模型主要基于AlexNet，做了相应修改。

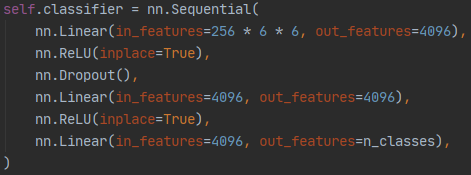
1. 设置各卷积层和池化层：



1. 设置平均池化层：

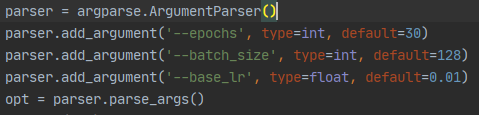


1. 设置各全连接层：

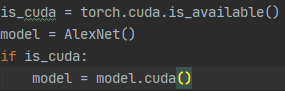


## 训练网络

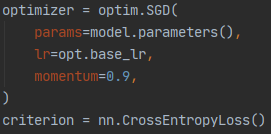
1. 设置各超参数值：



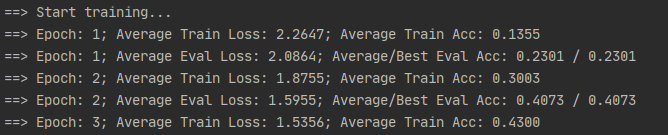
1. 判断GPU是否可用



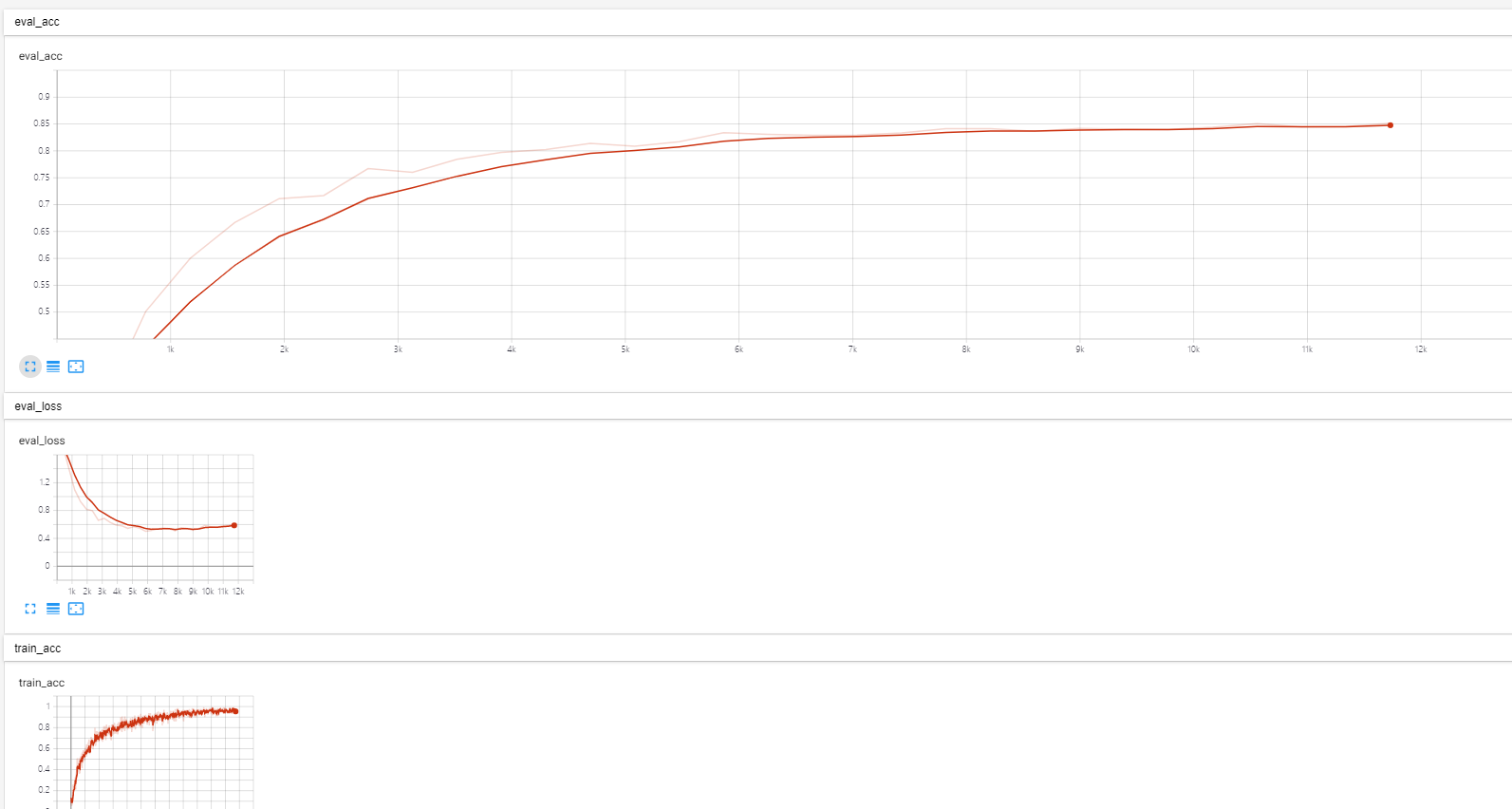
1. 设置损失函数和优化器：



1. 工程根目录终端输入：python train.py --epochs 10 --batch\_size 128，开始训练：

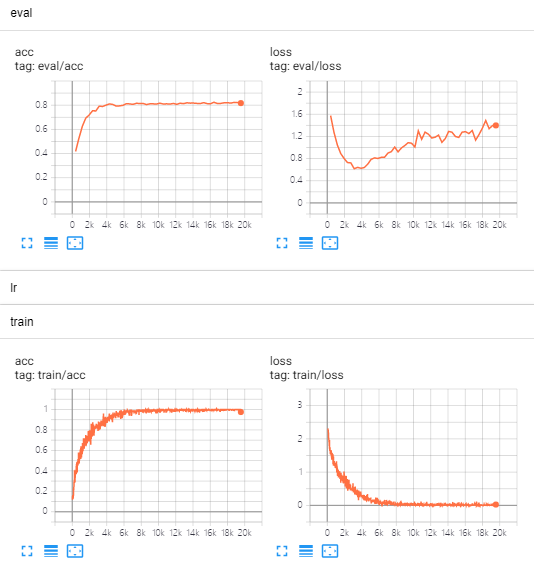


1. 工程根目录终端输入：tensorboard --logdir=./runs，浏览器打开：localhost:6006，监控训练过程：



# 拓展实验（占40%）

1. 我们搭建的模型相比原版AlexNet修改了哪些地方？为什么要做这样的修改？
2. 在上述默认配置下训练50个epochs左右时在TensorBoard中出现下图所示的情况：



从图中可见，训练集的accuracy接近100%，loss不断下降，已接近0；而验证集的accuracy在0.8附近震荡，loss仍大于0.4且出现重新上升趋势，可以判断此时出现过拟合（Overfitting）现象。下面尝试通过几种方法来减轻过拟合现象：

1. 减小模型复杂度，减小模型各卷积层和全连接层的output维度，例如第一层改为：

nn.Conv2d(in\_channels=3, out\_channels=32, kernel\_size=5, stride=1, padding=2) # out\_channels由64改为32

1. 在模型全连接层前添加Dropout（nn.Dropout()）；
2. 添加正则化项，例如将训练时计算loss的代码改为：

loss = criterion(outputs, targets) + reg\_lambda \* calc\_reg(model)

其中reg\_lambda为正则化系数，自已定义合适的值。给出calc\_reg()中的代码，计算模型各卷积层和全连接层参数的正则化项，具体实现是L1或L2正则化项皆可尝试；

1. 在datasets.py中基于torchvision.transforms做数据增强，例如transforms.RandomHorizontalFlip()；
2. 使用optim.lr\_scheduler.StepLR()实现学习率衰减；
3. 其他方法。

完成相应修改后重新训练，阐述你做了哪些有意义的尝试，分别具体使用了哪些方法，得到了什么样的结果，给出修改的关键代码和类似下图所示的TensorBoard对比图（图中有3次实验，分别是橙色过拟合、深蓝欠拟合和浅蓝相对正确拟合的情况）。

