# 第一周-图像识别-PRML小组项目报告

组员：叶帆帆（浙江大学）、梁宵（浙江大学）

#### 项目成绩





图 1项目成绩（NO.2 & NO.12）

#### 项目内容及目的

本周主要对手写汉字进行识别分类，旨在锻炼深度学习构建网络，处理数据，进行训练与测试的实战能力。我们**同时使用Caffe和****TensorFlow框架进行项目实战**。Caffe使用的是略微修改的Alexnet，TensorFlow使用的是自己设计的一个网络。下面我们将分别介绍两种框架下的具体实现过程。

#### 项目测试环境及数据来源

系统环境：科赛K-Lab&线下服务器（GPU：1080Ti）

数据来源：科赛平台提供的数据+中科院自动化所提供的补充数据

<http://www.nlpr.ia.ac.cn/databases/handwriting/Application_form.html>

#### 实战过程

* **基于caffe框架的实现过程（线下初步训练，线上测试）**

**数据预处理**

1. 给数据集打标签

我拿到任务后第一反应是先了解下数据集的形式，看完数据集以后非常感动。科赛给我们提供的数据集竟然是已经分好类的图片形式的数据集，这给作为深度学习新手的我节约不少时间。于是我就愉快的给数据集打起了标签，几行代码就轻松搞定。

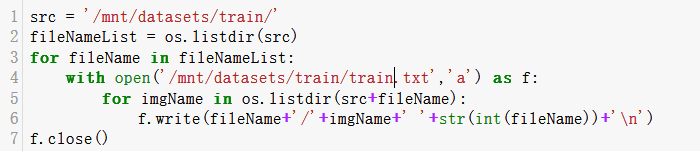


图 2打标签代码截图

1. 将图片数据及转化成leveldb格式的数据集

Caffe提供非常多适用的工具和接口，在处理完数据的标签之后，将train和validation的数据转化成leveldb格式的数据集，这一步只需要调用caffe提供的接口函数即可，具体如下：

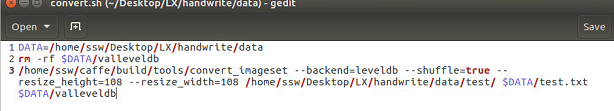
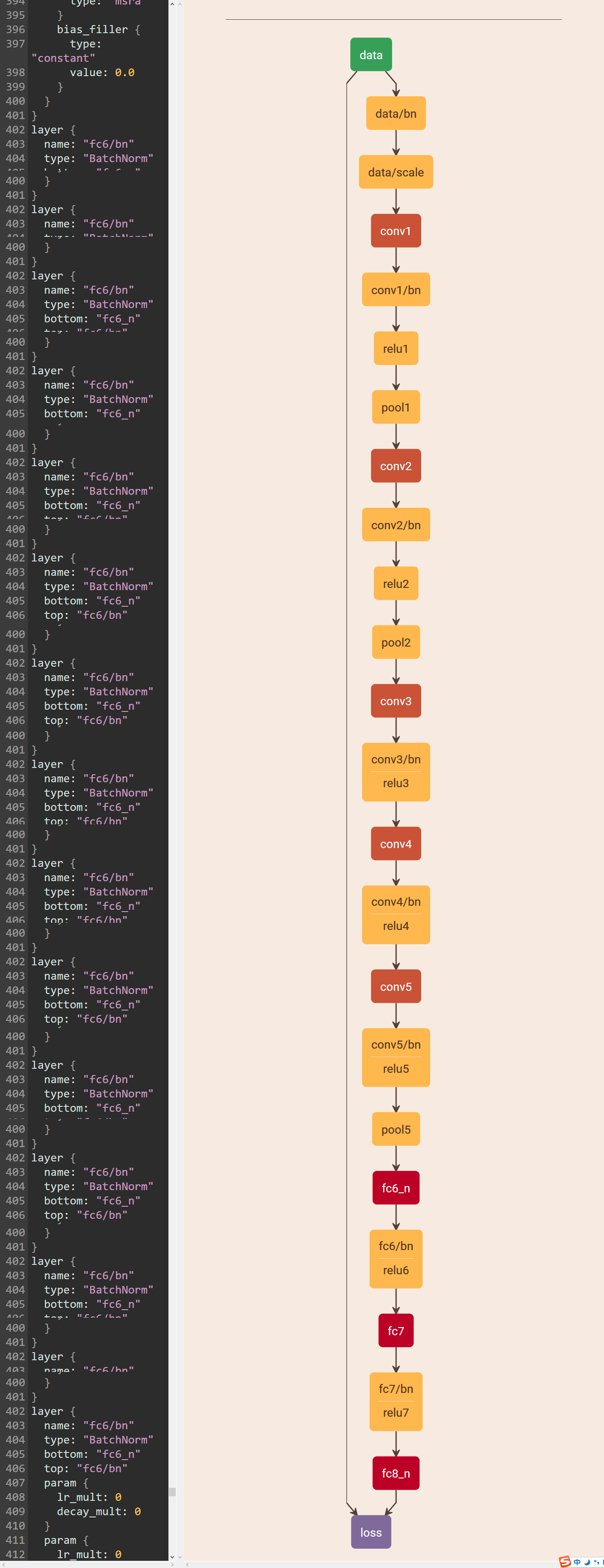


图 3数据转化脚本

转化的过程中将数据resize为108\*108（网络经验值）。**值得一提的是，在制作leveldb的时候混洗（shuffle）一定要打开，不然很可能不收敛。**

**网络结构选择**

根据经验结合图像大小和内容复杂度，我和我的队友一致认为不需要很深的网络应该就能比较好的解决该问题。由于提供的数据集图片大小较小，故也不能再网络中加太多的pooling层，几番思考后，我决定用AlexNet。网络结构图如下：



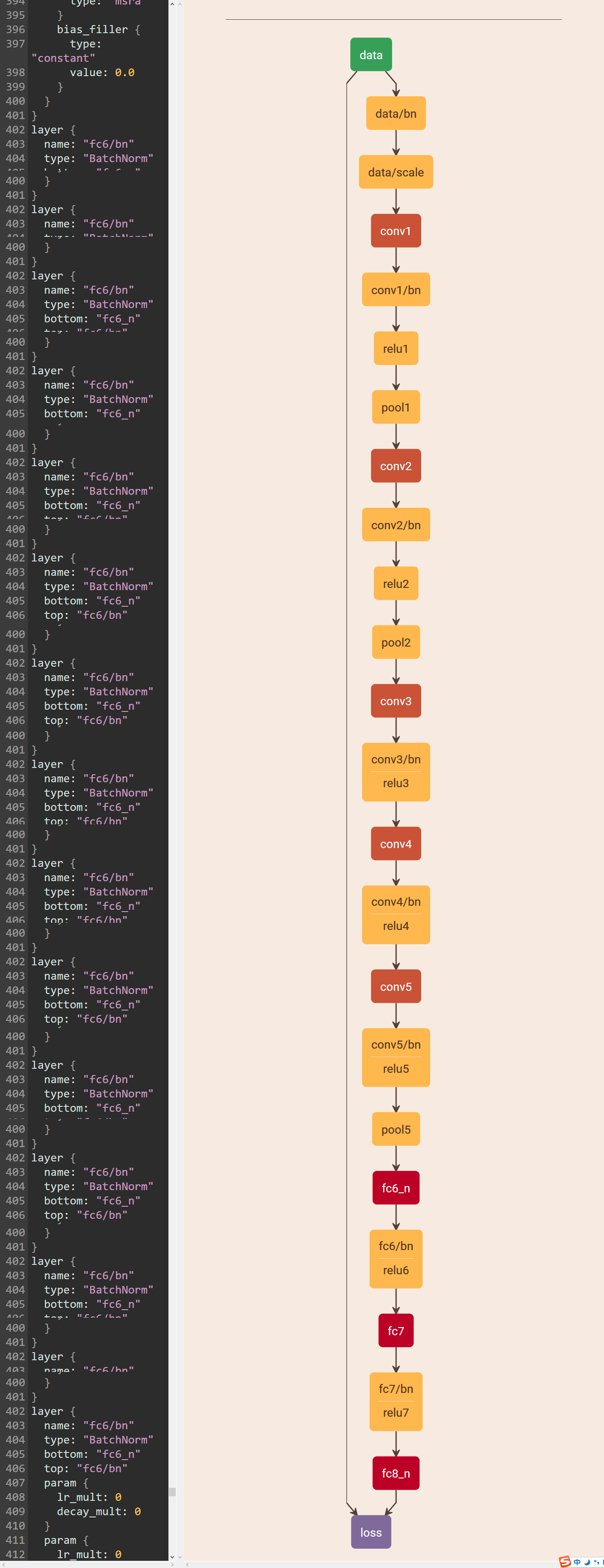
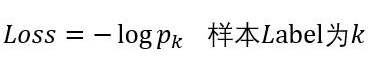
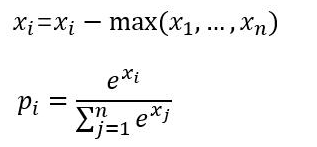


图 4 AlexNet 网络结构

**Loss函数选择**

对于多分类问题，loss函数就非SoftMaxWithLoss莫属了，其核心公式如下：



caffe中也提供了SoftMaxWithLoss层的实现，可以说是很方便了。

**训练参数详细**

这一部分是最关键也是最费时的。首先我采用最简单的SGD优化器，而且没有使用预训练模型，而且没有shuffle我的训练数据集。结果可想而知，网络训练了一整夜，也没有收敛。后来我查阅相关文献后，将优化器改为Adam，重新制作shuffle后的数据集，也加入了预训练模型，果然很快收敛，但是在loss=1附件来回震荡，验证准确率也在70%左右徘徊。后来又将学习速率lr减小，重新训练。经过数十个小时的训练后，得到了还不错的结果，验证准确率到了95%左右，loss也到了0.03左右。详细的参数设置如下：

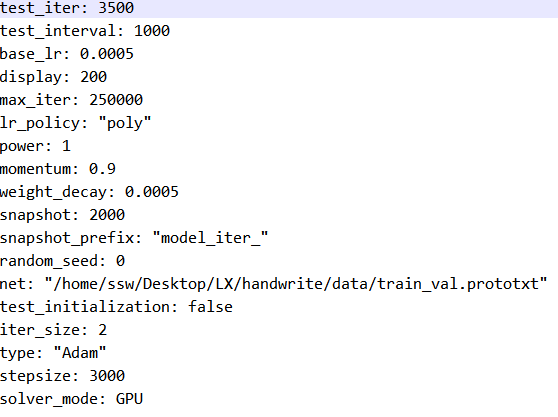
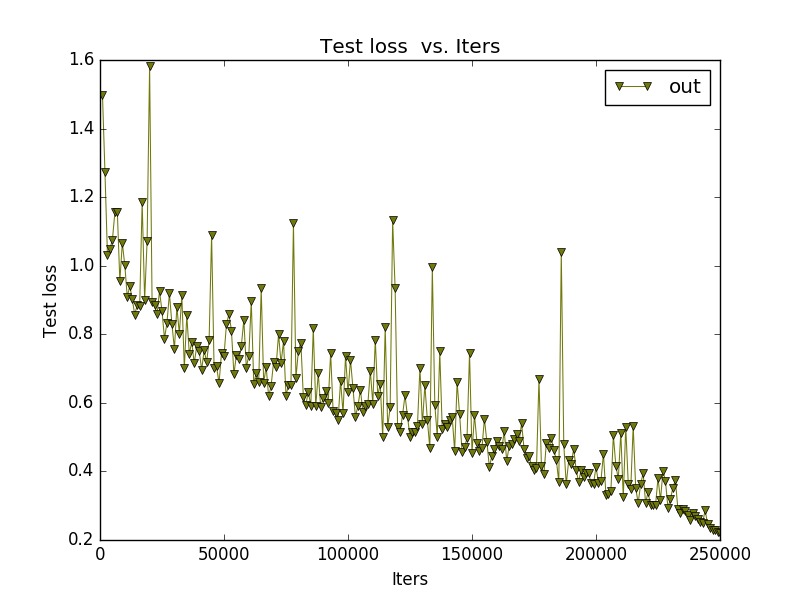
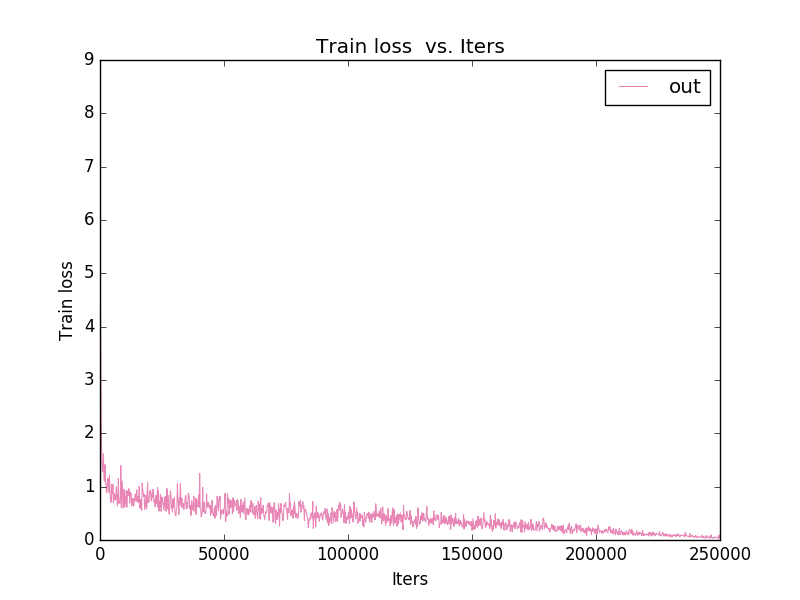


图 5 训练参数详细

**训练测试结果**

根据caffe训练的日志文件，绘制训练loss和accuracy图如下：



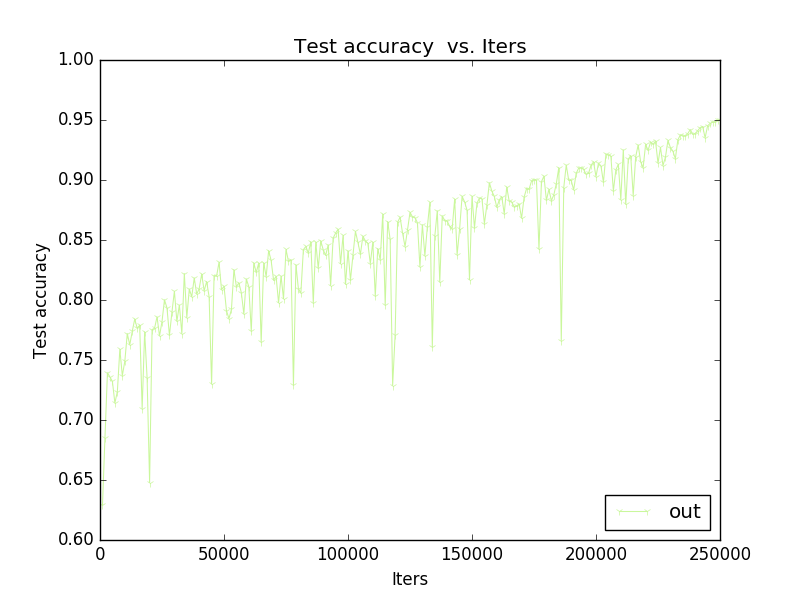


图 6分别为训练loss、测试loss和验证准确度图

**线上结果测试**

线上一共提供了39000+个测试集，对测试集中的输入图片逐一测试，并生成json格式的结果输入，具体代码如下：



图 7测试代码截图

最后线上的结果是94.4022%，第二名，是一个还可以的结果。由于时间紧迫也没有再做进一步提升，这是我感到遗憾的一点，希望下周能够继续加油。在此再次向志愿者和辅导员以及相关工作人员表示感谢。

* **基于TensorFlow框架的实现过程**

**数据预处理**

读取路径后，对数据进行混洗，防止过拟合，并按文件夹名对数据进行标记。

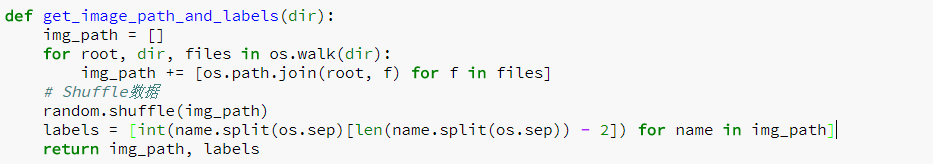


图 8 TensorFlow数据处理代码

由于数据集很大，对数据进行分批操作，对图片进行resize，大小我设为96\*96，将图片与label对应好后返回。

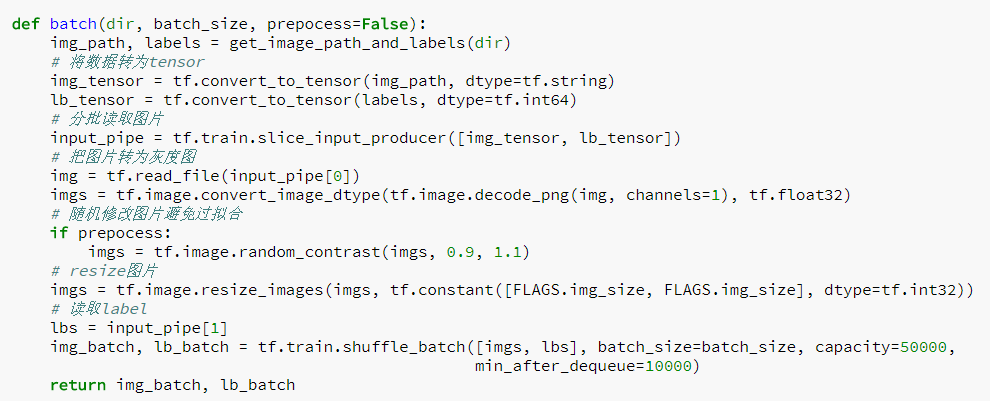


图 9 TensorFlow数据处理代码2

**搭建网络**

考虑到上传时模型不能大于100m，同时图片resize后较小，搭建了一个较浅的网络，由5个卷积层和两个全连接层构成。

第一层：5\*5的卷积核32个，步长为1；

第二层：3\*3\*32的卷积核64个，步长为1；

第三层：3\*3\*64的卷积核128个，步长为1；

第四层：3\*3\*128的卷积核256个，步长为1；

第五层：3\*3\*256的卷积核512个，步长为1；

其中每层经过一个relu层和2\*2的maxpooling层

最后是两个全连接层。

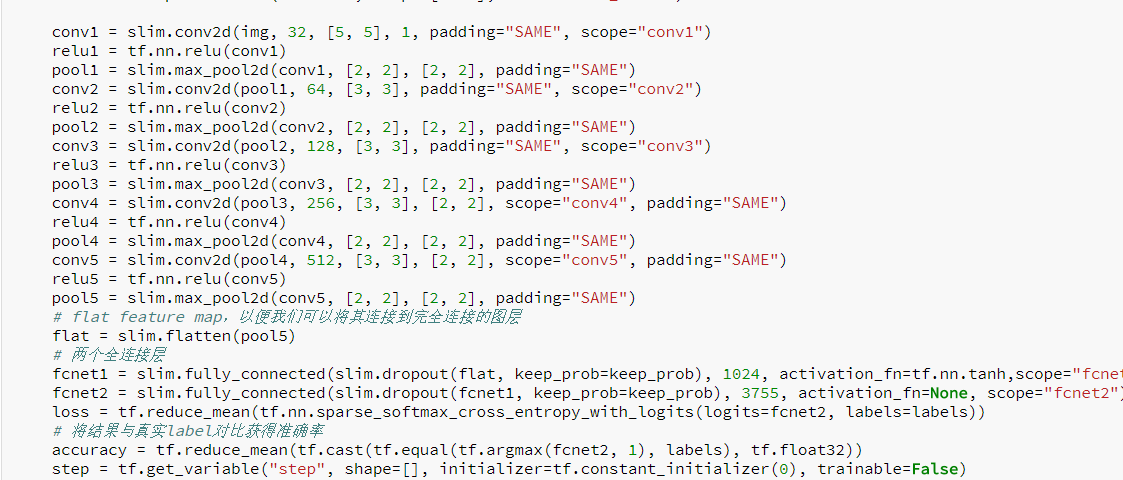


图 10TensorFlow模型构造代码

Tensorflow中的graph图如下图所示：

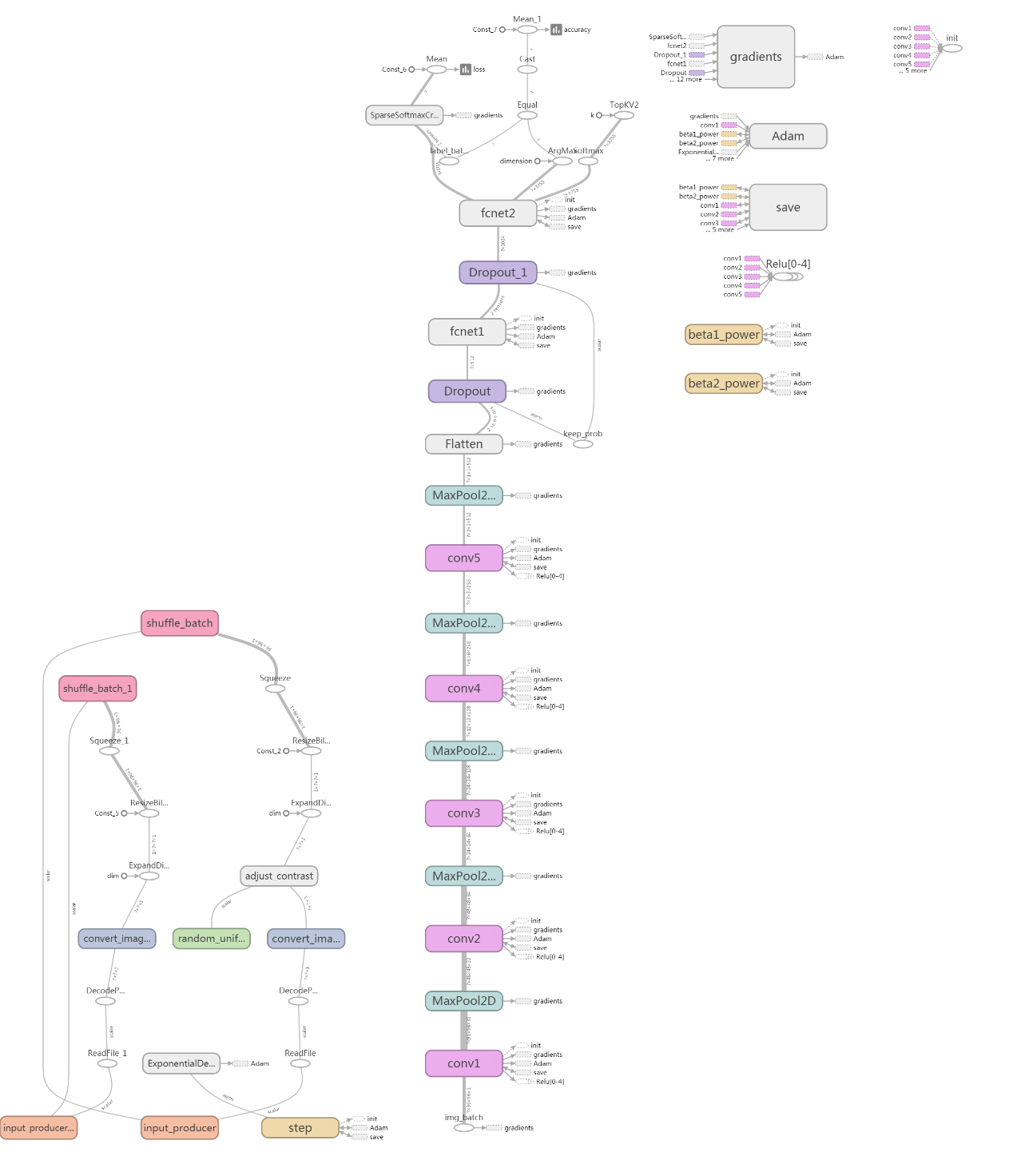


图 11TensorFlow模型graph图

**优化算法和参数设置**

使用Adam优化算法，初始学习率为2e-4，decat\_rate=0.97,decay\_steps=2000,指数部分采用整除策略，时间有限，具体参数还有待调整。

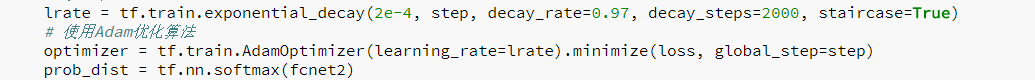


图 12 TensorFlow参数设置

**训练**

进行了20w步，每1w步保存一个模型， 每500步打印一次验证集的准确度，并输出log文件。

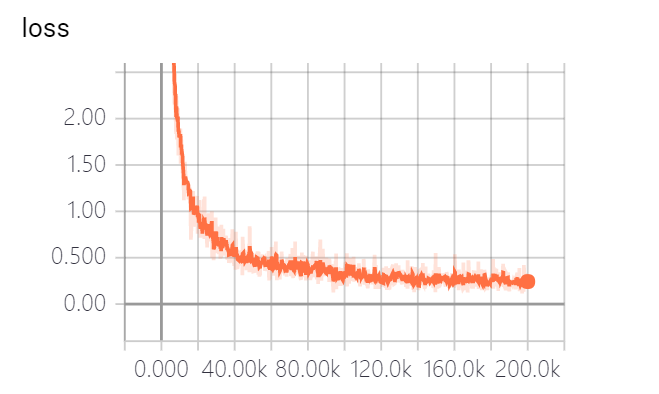


图 13TensorFlow训练Loss

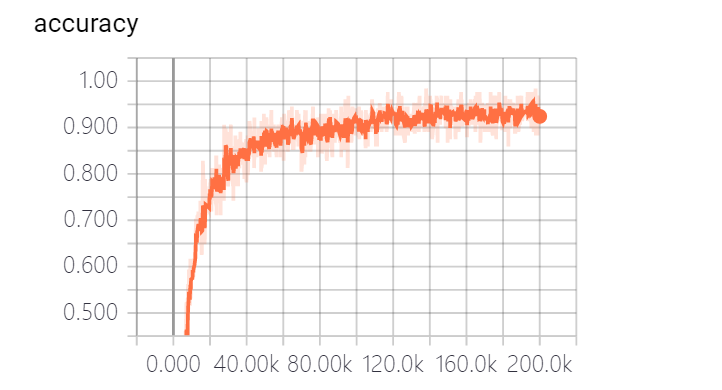


图 14TensorFlow训练准确率

可以看出loss在0.5附近不断震荡，准确度也在0.9附近震荡。由于时间有限，没有进行更多的尝试，我认为降低初始学习率也许能解决这个问题，因为在接近5w步的时候loss就一直在震荡了。

**测试**

测试使用的是平台的测试图片，采用的办法是一张张读图片进行label的输出，然后写入json文件。



图 15 TensorFlow测试代码

得到了如下所示的json文件：



图 16结果截图

将其提交，准确率在90.7%，不算太理想。

由于resize后图片大小与Alexnet的理想输入大小差的较大，再加上本身对tensorflow的不熟悉，没有用tf搭建Alexnet，在这个网络跑完后我们又尝试了个更浅的网络，过程与结果都大同小异。

#### 遇到的困难和思考

**梁宵：**在使用TensorFlow的过程中遇到了不少困难，因为之前我们使用的都是Caffe框架，没有接触过TensorFlow，因此会碰到不少低级错误，在老师和同学们的帮助下解决了。最大的困难还是到最后loss一直震荡，无法下降，我觉得可以从网络的搭建和参数的设置上入手，网络的搭建可以参考我们Caffe框架使用的网络，参数设置上，我认为可以适当调低学习率。在这里也感谢一下队友，能从对方身上学到很多东西。

**叶帆帆：**看似简单的项目完整做完还是会碰到很多困难。我在第一周的项目中遇到的主要困难是网络对参数的调节，在项目过程中也积累了一些经验，希望下周能少犯一些低级错误。第一个小项目的顺利完成，得益于工作人员以及志愿者的热心服务，再次表示感谢。

#### 希望和导师、同学交流的地方

希望能在群里多讨论一些代码实现方面的问题，这些都是较为薄弱的环节，有些时候讨论后能发现更加方便的实现办法，受益匪浅。