毕业设计记录

## 目标：

借鉴RCNN算法的思路实现图像内物体检测与Bounding Box的确定，并制作一个网页前端，可以上传图片进行计算并展示。

## 环境：

* 1. GPU：GTX 960M 4G
  2. CPU：i5-6300HQ
  3. 内存：8G

系统：Ubuntu 18.04.1 LTS

CUDA：V9.0.176

NVIDIA驱动：390.87

Python：2.7

数据集：Pascal VOC 2012

MXNet：1.3.1

没有设置cudNN。

## 关于数据集：

* 1. 一共含有17125张图片。

ImageSets文件夹中：

其中Action下存放的是人的动作（例如running、jumping等等）

Layout下存放的是具有人体部位的数据（人的head、hand、feet等等）

Main下存放的是图像物体识别的数据，总共分为20类。

Segmentation下存放的是可用于分割的数据。

主要使用Main文件夹中的用来训练。

## 实现的大致思路：

* 1. 对于一张图片，使用选择性搜索（Selective Search）会产生一些候选框，再对这些候选框进行简单的筛选后（除去长或宽小于32的框），把每个候选框都重新resize后输入到训练好的AlexNet中得到输出分类以及得分，再进行非极大值抑制（NMS）之后把该候选框在图片上画出，并展示在网页前端。
  2. 对于训练，根据Annotations文件夹中的xml文件，把JPEGImages中的每张图片中的每个object都裁剪出来并resize到AlexNet的标准输入尺寸（227x227），把这些图片制作成RecordIO数据对AlexNet进行训练。

## 生成训练用数据集：

首先，根据Annotations文件夹中的xml文件，把JPEGImages中的图片中的object都裁剪出来并resize到（224x224）保存在ResizedObjects文件夹中，这部分工作由DataSetObjectExtraction.py完成。

然后，使用产生随机数的方式将这些图片以7:2:1的方式划分为训练集、验证集和测试集，这部分工作由GenerateImgList.py完成，生成的lst文件存放在Data/RecordIO中。

再然后，用mxnet自带的im2rec.py生成rec文件，在控制台中输入如下命令即可：sudo python utils/im2rec.py --num-thread=4 Data/RecordIO/train.lst .。

至此，就完成了数据集的制作。

## 搭建网络并训练：

参考AlexNet标准网络结构，并根据Pascal VOC数据集一共有20个类别的特性，将第三个全连接层的单元数设置为20。

迭代器ImageRecordIter设置的参数有：

data\_shape=(3, 227, 227),

batch\_size=50,

mean\_r = 128,

mean\_g = 128,

mean\_b = 128

其他为默认。

训练使用的学习策略为MXNet中默认的FactorScheduler，参数为每500次迭代学习率衰减为原来的0.95。

训练总epoch为40，每10个epoch保存一次模型。

搭建的网络结构图为AlexNet.gv.pdf。

## 迷你数据集训练：

在进行大数据集训练之前，应准备一个迷你数据集先测试一下，因此取了训练集中的前30个图片和验证集的前20个图片作为迷你的训练集和验证集。主要是为了验证网络结构的正确性，对于如此小的数据集，应当能在没有正则化的情况下轻易达到过拟合。

## 训练记录：

1.训练总用时5835.39844489s，Train-accuracy一直保持在0.43左右，Train-cross-entropy一直保持在2.27–2.29之间，Train-mse一直保持在75.3多，Validation的也差不多，也是基本保持不变。

## 测试和评估：

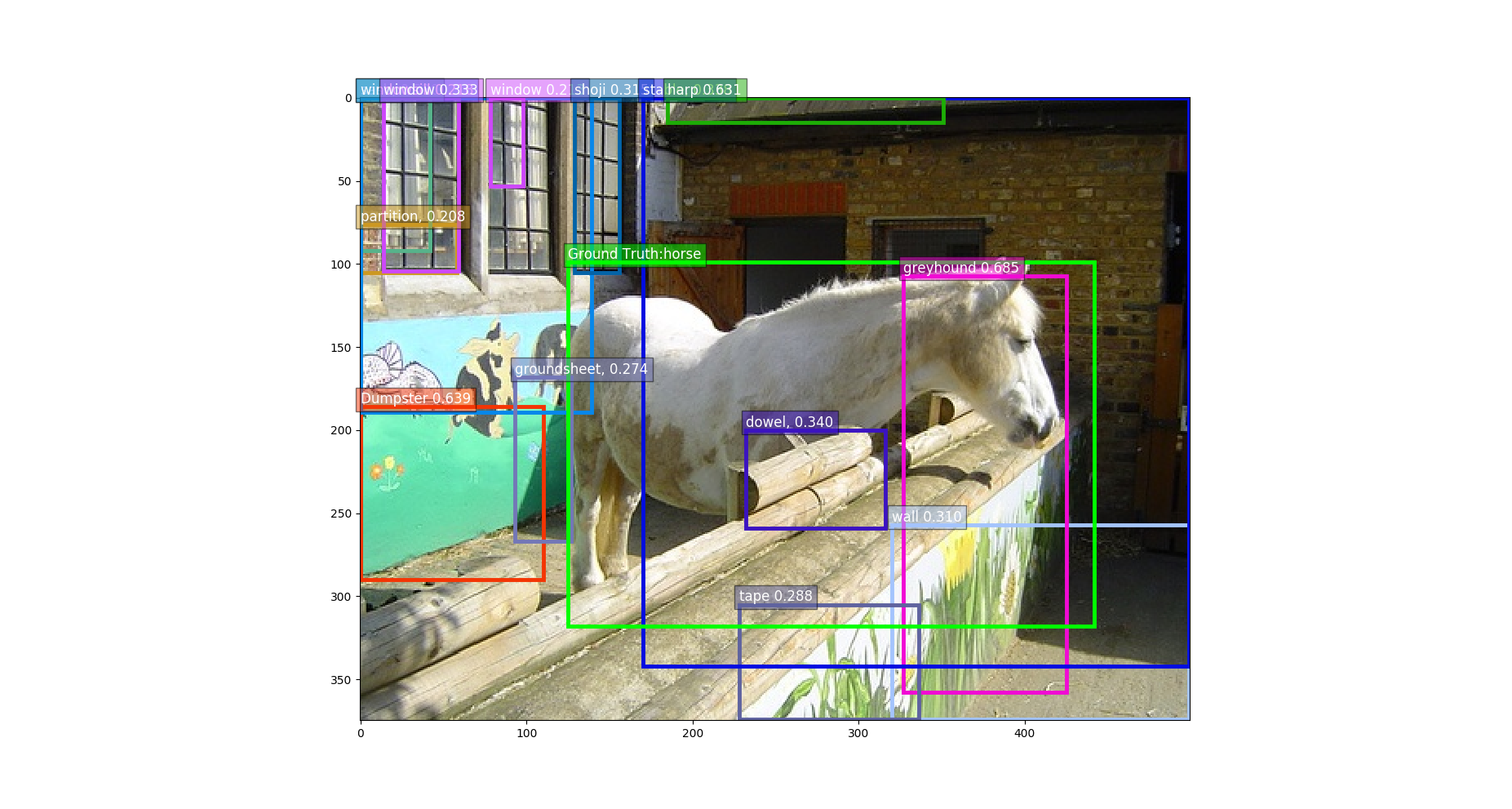
接下来要对训练过后的模型进行测试和评估。

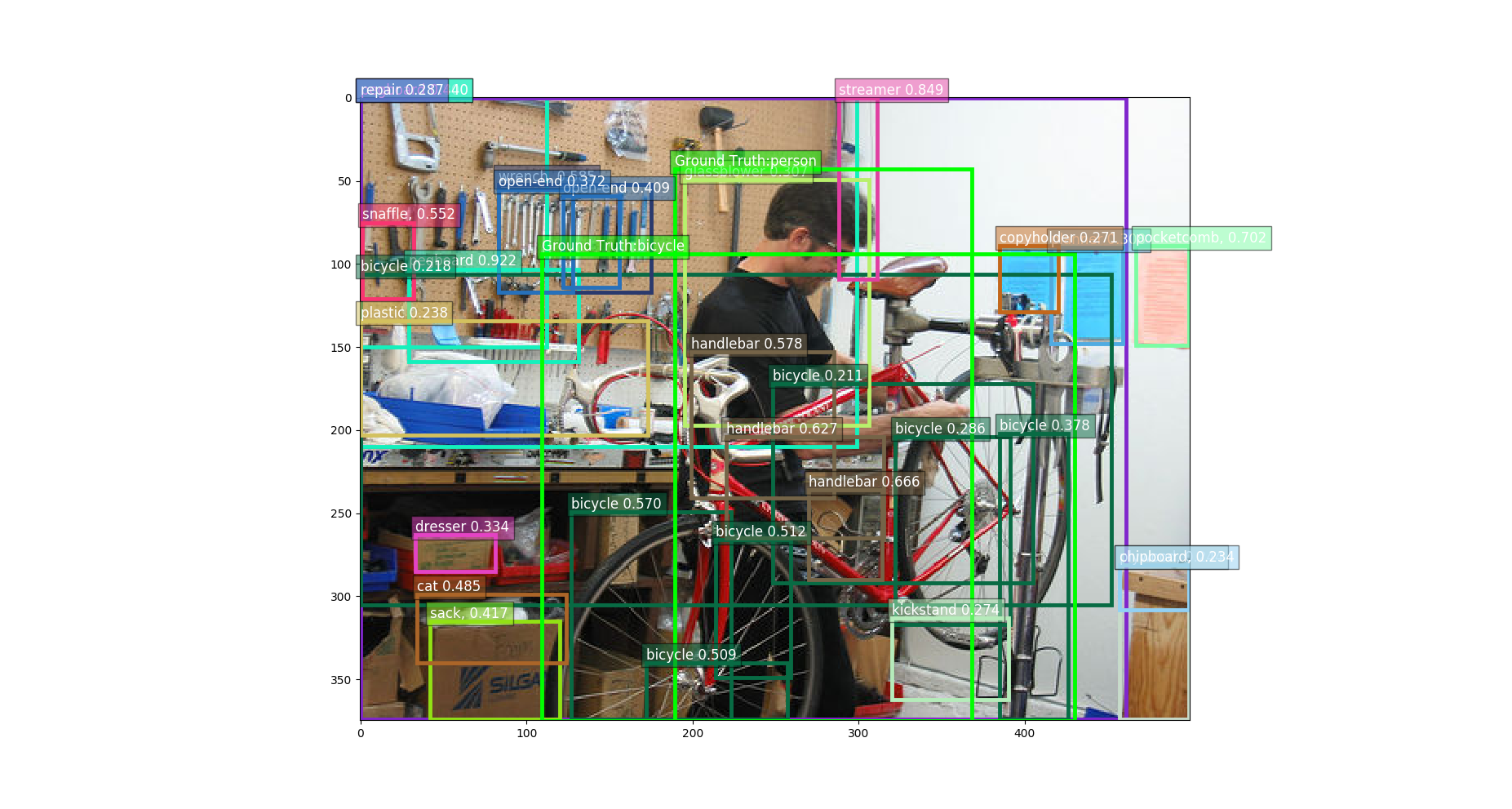
首先是测试，这里使用之前分出来的test测试集中的数据进行测试，载入的参数为训练时保存下来的最后一组参数，也就是编号为35的params文件，使用的指标为accuracy，即准确率。

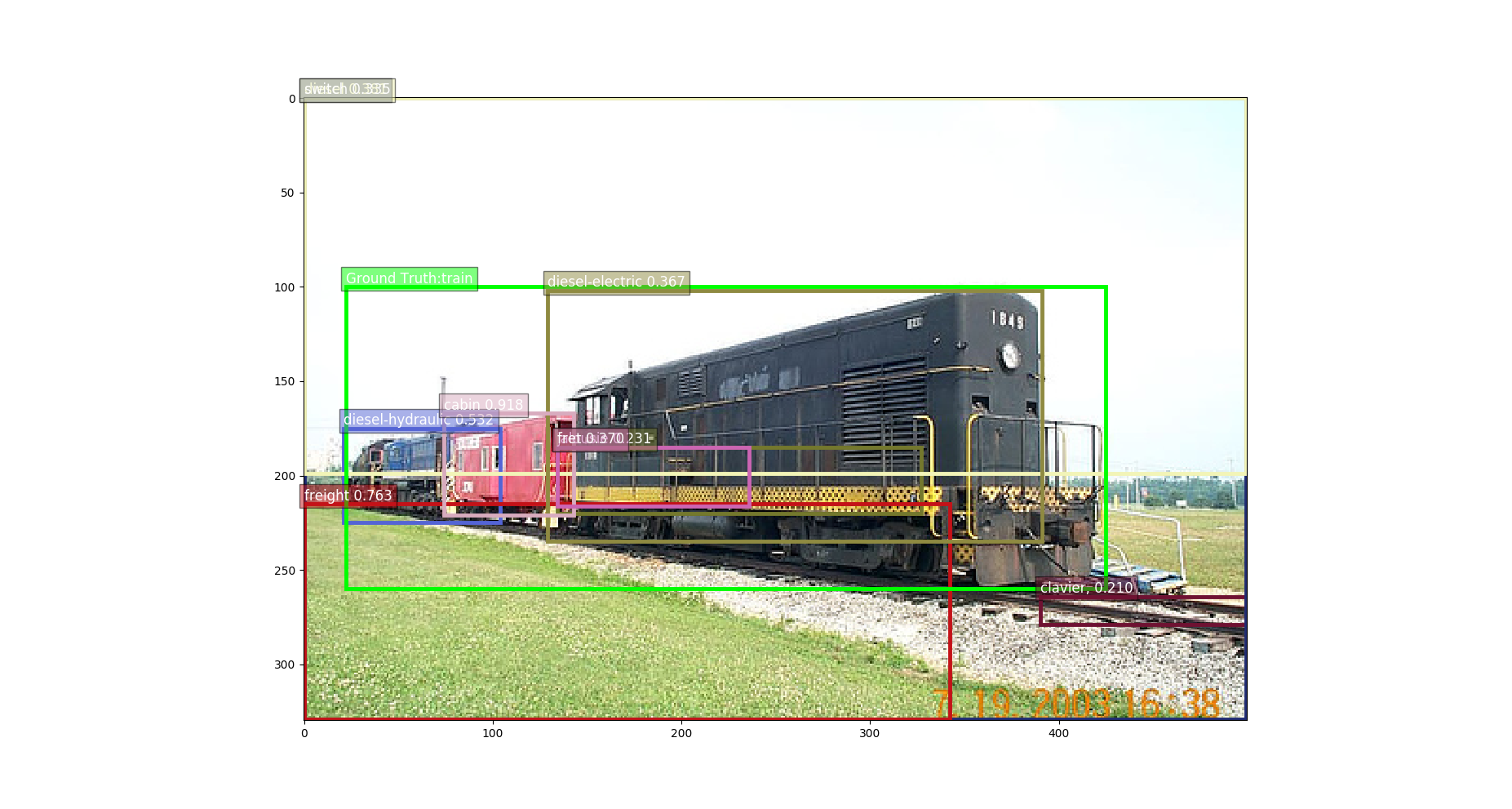
（接下来的步骤都是使用在ImageNet上训练好的ResNet152模型进行的）

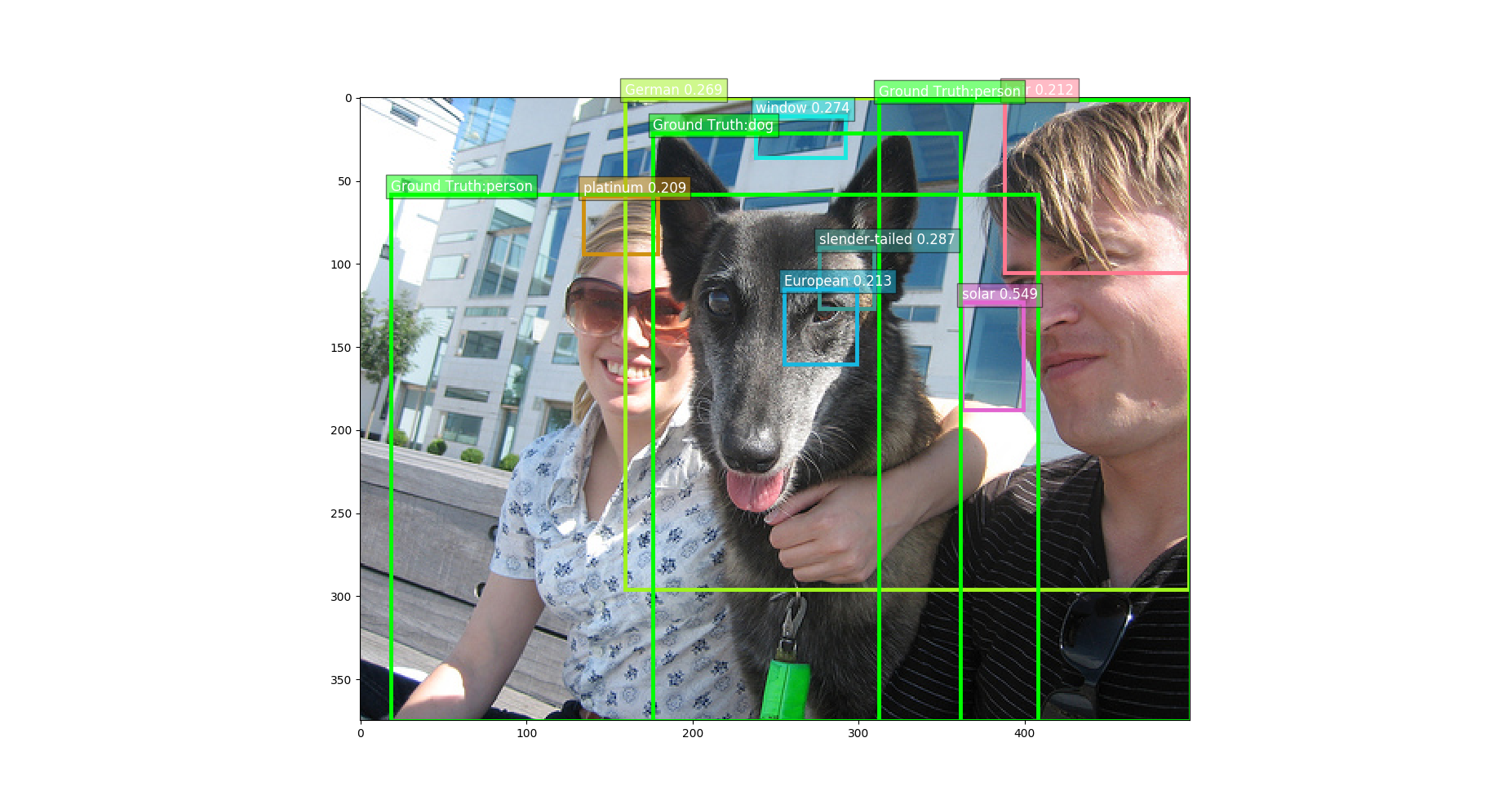
首先用ResizedObjects中的图片输入到预训练好的模型中，发现分类虽然大多数时候是没问题的，但是probability都不是太高，大多数就0.3-0.4左右，最高就0.5多，这部分在test/pretrainedModelTest.py中完成。

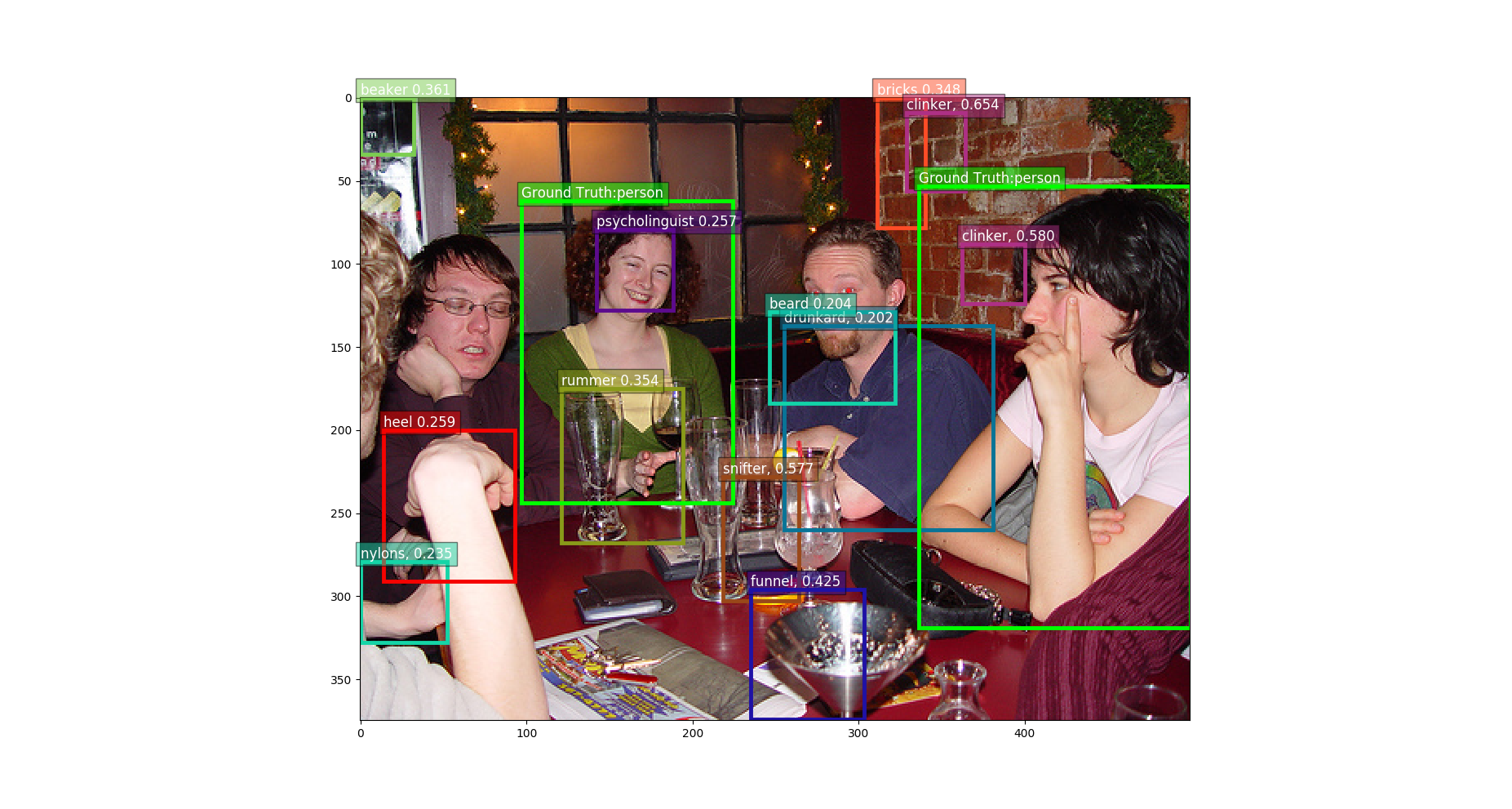
将Pascal VOC数据集中的某些图片直接整张输入，用selective search之后经过NMS，再将候选框resize之后输入到ResNet152之中，最后将结果画在图上，这部分在test/testPredictorInterface.py中完成（筛选掉了其中prob低于0.2的候选框）



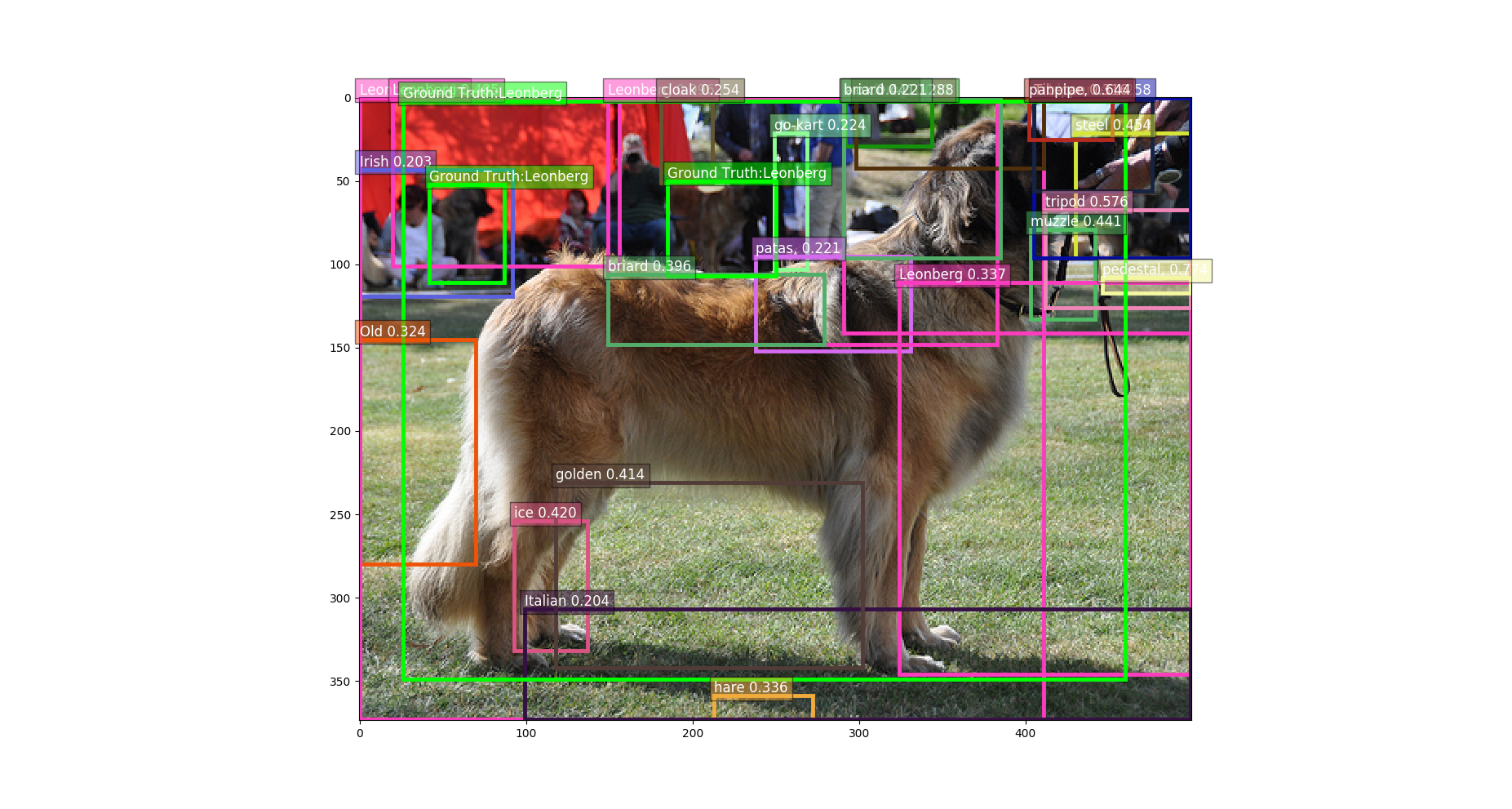


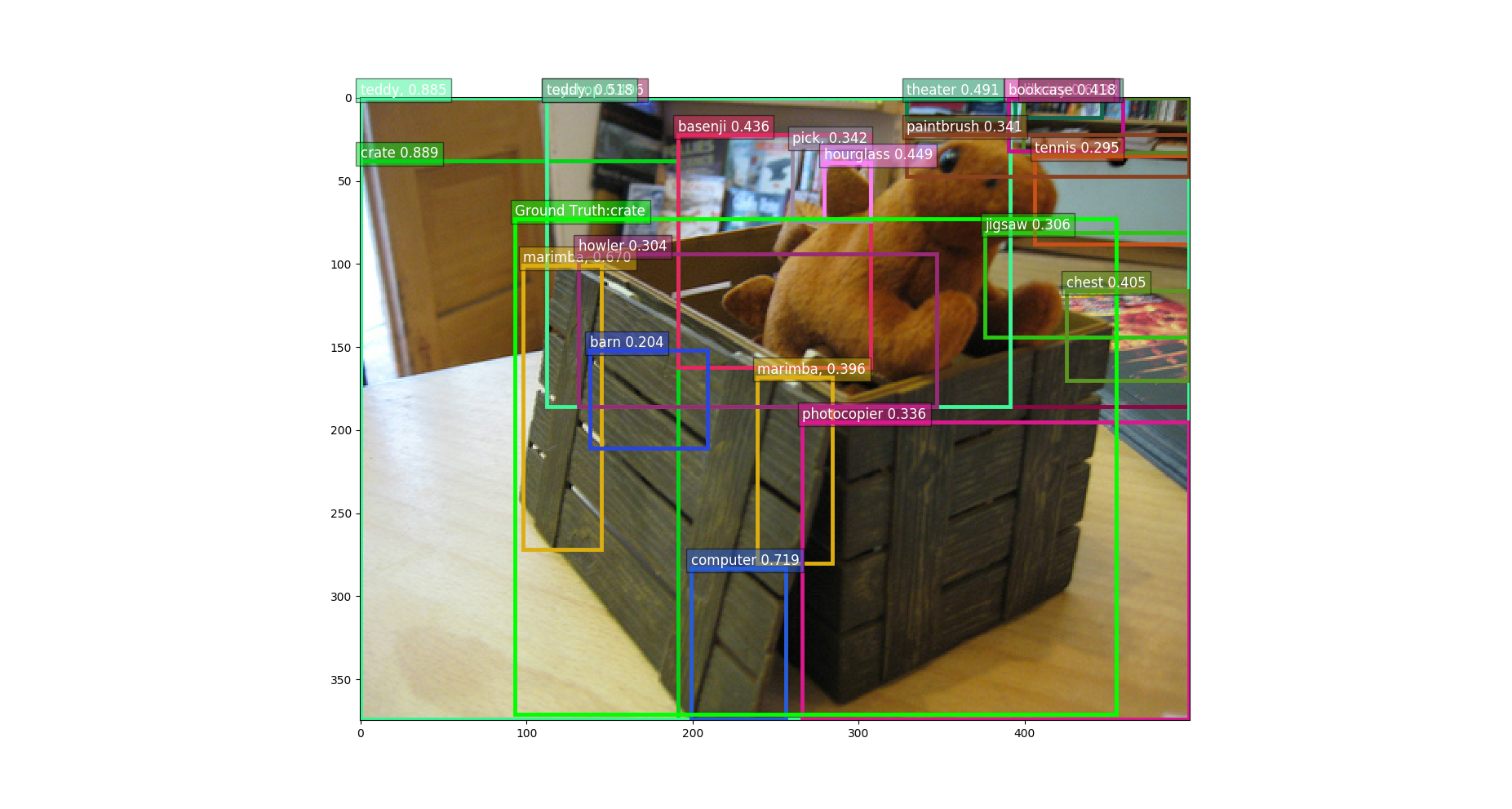


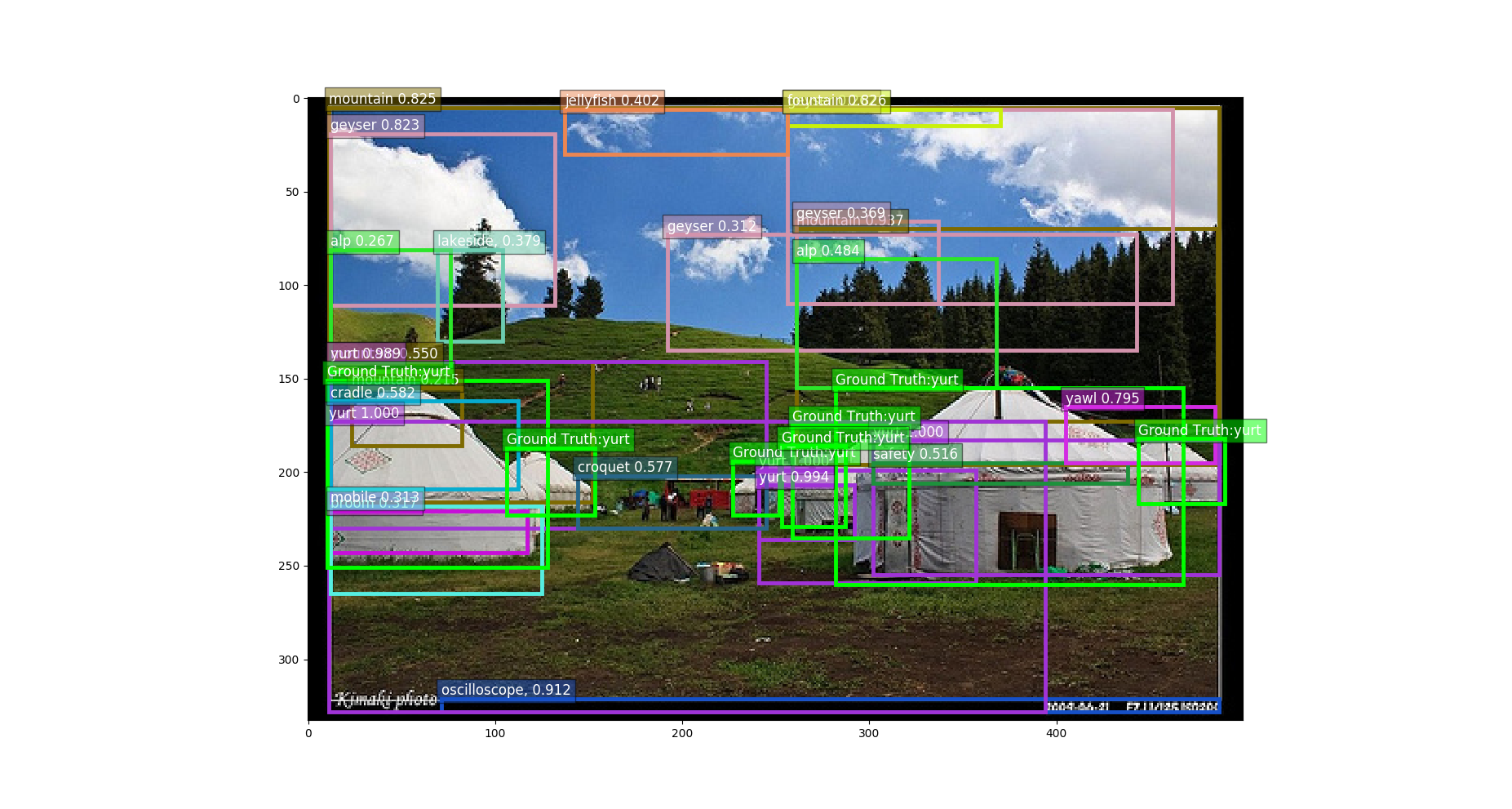




因为这个模型是在ImageNet上训练的，所以类别会不一样，看起来效果就不太好。接下来是在ImageNet的val上测试的结果：







其中绿色的框为ground truth，其他框为程序识别结果。

可以看到效果并不是很好，有些分类是错误的，有些该框的没有框出来，有些框小了，并没有包住整个物体，有些大框直接套着小框，有些框在了乱七八糟的地方经过模型却依旧会有比较高的prob。

而且在将threshold设置为0.3之后直接会漏掉一大堆该识别出来的物体，说明模型对该识别出来的物体把握不高。

在RCNN的论文中有对这些框做Bounding Box Regression以获得更好的框的效果，它的想法是训练一个模型，学习的是结果框和Ground Truth之间的一种映射，训练完了之后对每个结果框做一次这个映射便可以达到调整。

评估：

本来想用MAP和IOU这类指标，对这个预训练好的模型做个评估，但是我也不会再去训练这个模型了，就会一直这样用着了。先去写前端了，后面再考虑评估和自己训练或是对这个做优化吧。

## 图片的检索：

* 1. （1）考虑根据ImageNet的1000个类别，建立1000个文件，再根据ground truth把ImageNet中的图片中的物体切割并通过模型计算出特征，在对应的文件中记录下这个特征，检索的时候根据输入的图像先识别出其中物体的类别，再找到对应的文件，对其中的每个特征做一个计算，选出其中最匹配的前N个物体，并把它们对应的图片找出来，画出ground truth后显示在前端，另外输入的图片经过模型可能会识别出很多物体，应当选出其中的主要物体吧，不能全都去检索。

（2）直接对整张图片提取特征，在图片级别进行检索。

首先就需要有一个工具可以提取出一个图片的特征，这里对于预训练模型ResNet152来说，通常是取全连接层之前的那层的输出作为特征。

## 前端：

至于前端，使用apache2作为服务器的启动，想要把RCNN以及它的各种改进都各自展示出来

## 错误记录：

1.**利用list文件产生rec文件时发生的错误：**

**hongyigeng@hongyigeng-GL552VW:~/PycharmProjects/ImageSearch$ sudo ~/mxnet/bin/im2rec train.lst Data/ train.rec**

**/home/hongyigeng/mxnet/bin/im2rec: error while loading shared libraries: libcudart.so.9.1: cannot open shared object file: No such file or directory**

检查过环境变量没有问题。

尝试以下3条命令：

$ sudo cp /usr/local/cuda/lib64/libcudart.so.9.0 /usr/local/lib/libcudart.so.9.0 && sudo ldconfig

$ sudo cp /usr/local/cuda/lib64/libcublas.so.9.0 /usr/local/lib/libcublas.so.9.0 && sudo ldconfig

$ sudo cp /usr/local/cuda/lib64/libcurand.so.9.0 /usr/local/lib/libcurand.so.9.0 && sudo ldconfig

无效

解决办法：

使用mxnet/tools文件夹内的im2rec.py即可

sudo python ~/mxnet/tools/im2rec.py --num-thread=4 train.lst .

2.**运行训练脚本时发生错误：**

**RuntimeError: simple\_bind error. Arguments:**

**data: (50, 3L, 227L, 227L)**

**softmax\_label: (50,)**

**[19:36:47] src/storage/storage.cc:119: Check failed: e == cudaSuccess || e == cudaErrorCudartUnloading CUDA: unknown error**

尝试重装CUDA9.1以替换CUDA9.0：

首先卸载：sudo /usr/local/cuda/bin/uninstall\_cuda\_9.0.pl

运行安装程序：sudo sh cuda\_9.1.85\_387.26\_linux.run –override

安装之后试图运行CUDA-examples中的deviceQuery出错，显示：

./deviceQuery Starting...

CUDA Device Query (Runtime API) version (CUDART static linking)

cudaGetDeviceCount returned 30

-> unknown error

Result = FAIL

解决办法：

重启。。。

3.**import mxnet的时候出现了OSError libcudart 9.0什么的错误**，直接卸载原来的mxnet装上mxnet的GPU版本mxnet-cu91

4.**运行训练脚本的时候出现如下错误：**

**Check failed: static\_cast<index\_t>(res.rows) >= param\_.data\_shape[1] && static\_cast<index\_t>(res.cols) >= param\_.data\_shape[2] input image size smaller than input shape**

**应该是数据集中的有些图片尺寸小于AlexNet的输入图片尺寸要求277\*277。**

解决办法就是要把数据集中的图片全部都resize一下。

## 问题记录：

1.训**练出来的模型的准确率太低，测试了测试集和验证集，准确率都只有40%。**查看了Annotations中的xml文件，发现很多图片标注出来的物体都不止一个，而我做lst文件时只是取了第一个物体做为该图片所属的类别，而第一个物体未必是该图片中最明显的，可能这个会是导致准确率低的原因吧。

试着去找单标签的数据集，即图片只有一个标签，图片中只有一个明显的主要物体，有CIFAR-10和CIFAR-100，但是这个数据集中的图片分辨率都太低，只有32x32，很不行。因此考虑还是使用Pascal VOC数据集，但是要带bounding box作为输出，考虑实现一个RCNN。

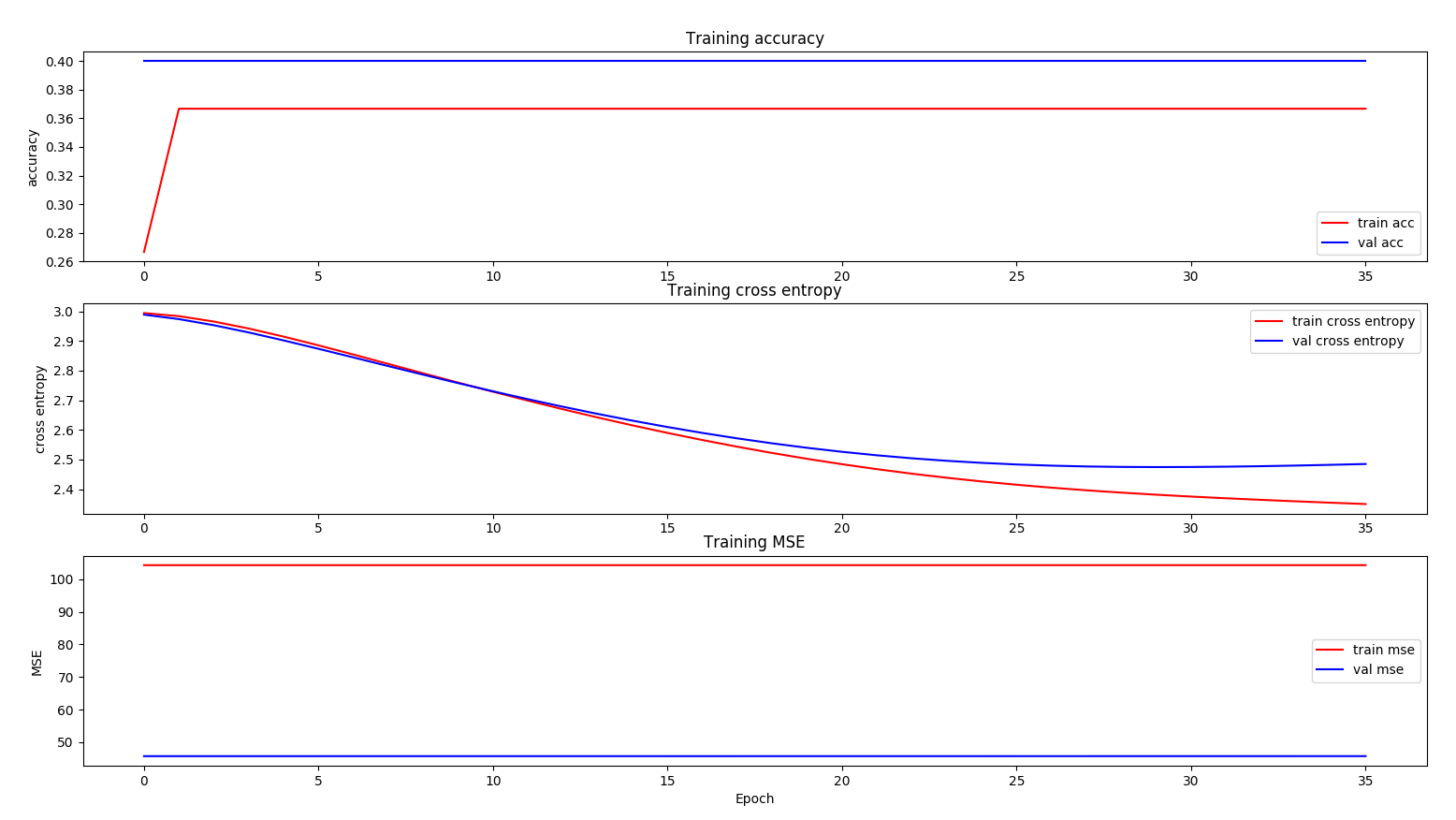
2.**在尝试RCNN的算法思想之后，用AlexNet训练框中的物体效果也不好。**

**测试之后发现准确率还是只有42.65%.。。。。**

**测试发现模型的预测全部都是person，而person占了测试集中的40%多。**

**查看训练日志发现训练时的loss根本就没有降。。。**

3.**在迷你数据集上的训练发现训练的效果居然也是一样！甚至还更差劲，Train和val的acc都超过40%。**



在把dropout层去掉之后效果依旧没有更好。

我现在尝试去训练一下CIFAR-10数据集，看是否正常，这部分内容写在test/testTrainingOnCifar10.py中：

除了optimizer\_params为自己原来使用的之外，其他部分包括数据的载入以及网络结构参考网上https://www.cnblogs.com/Mu001999/p/6221100.html，训练的效果如下：

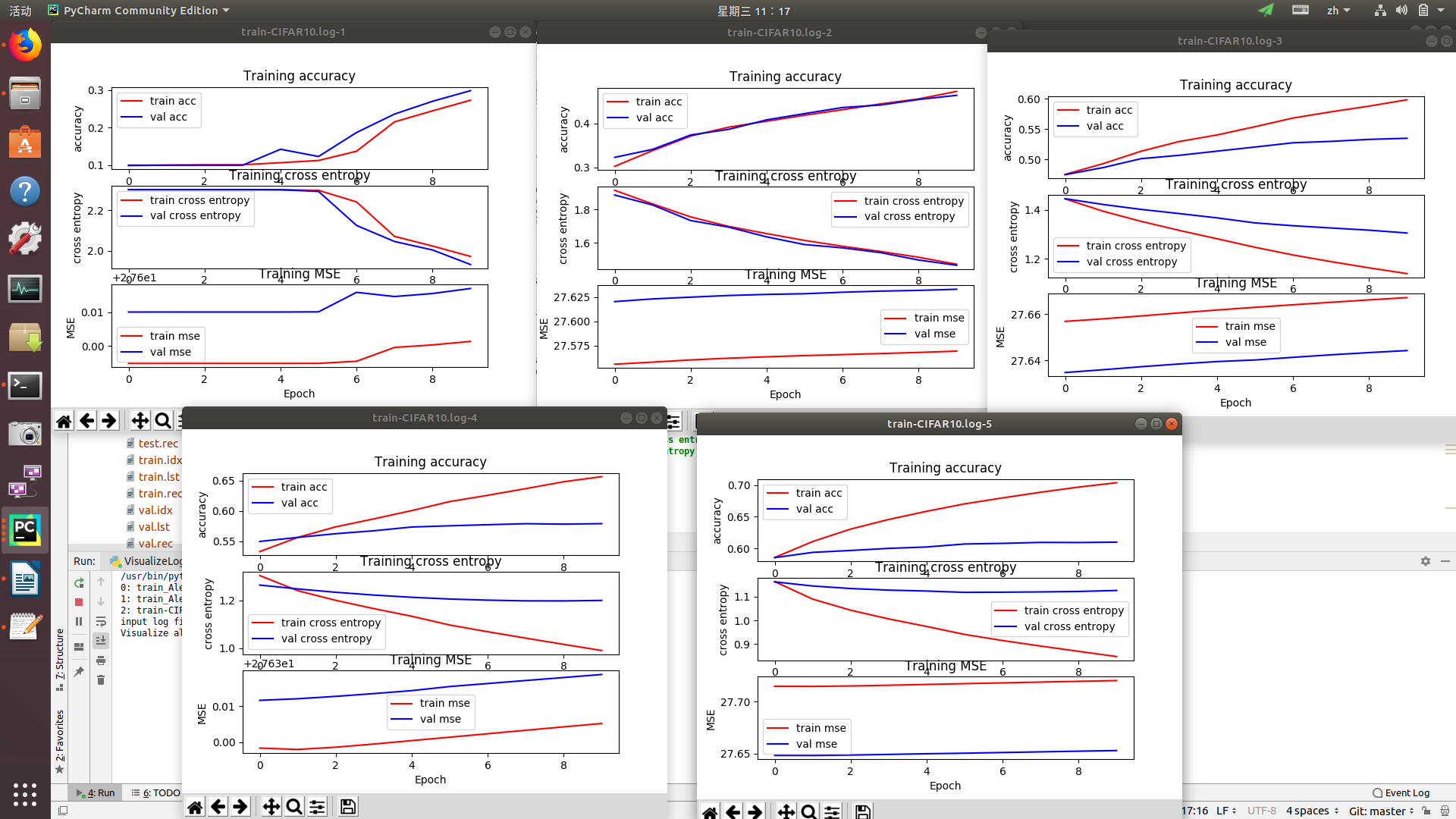


网络结构为MLP，现在考虑照搬那个博客写的，再看看效果如何：

直接用博客中的代码居然验证集的准确率一直都是0.1，数据集有10类，也就是说根本就没有训练效果！loss也一直不降！把激活函数换成sigmoid也没有用。

将网络结构换成lenet再来训练CIFAR10之后：

（lenet的结构来自mxnet/example/image-classification/train\_mnist.R）

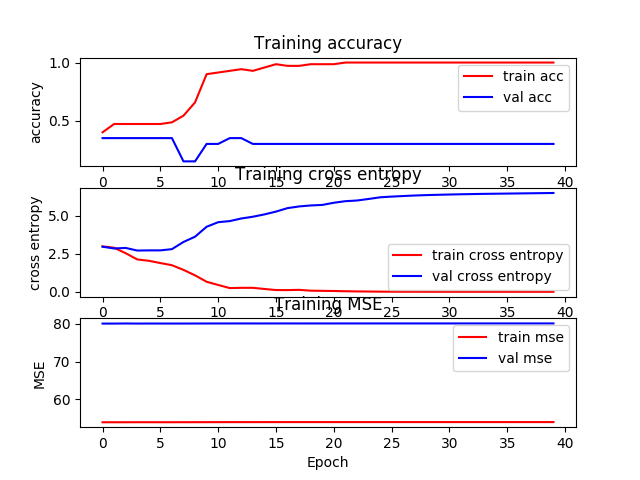
居然有效果了！！val\_acc训练结束时到达了60%多！（均方误差这个指标估计有点问题）

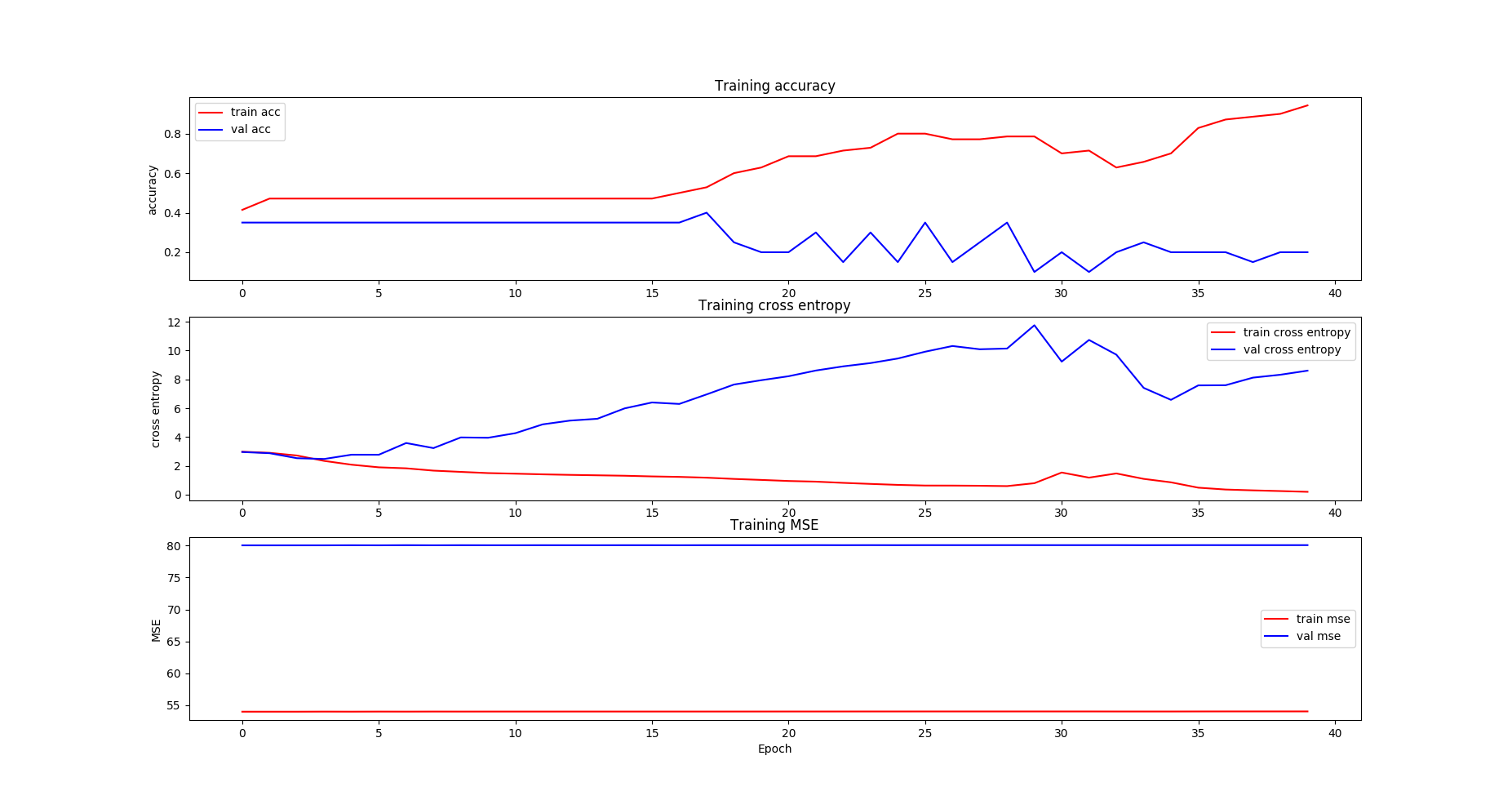
于是又返回到原来的AlexNet结构上做改动：

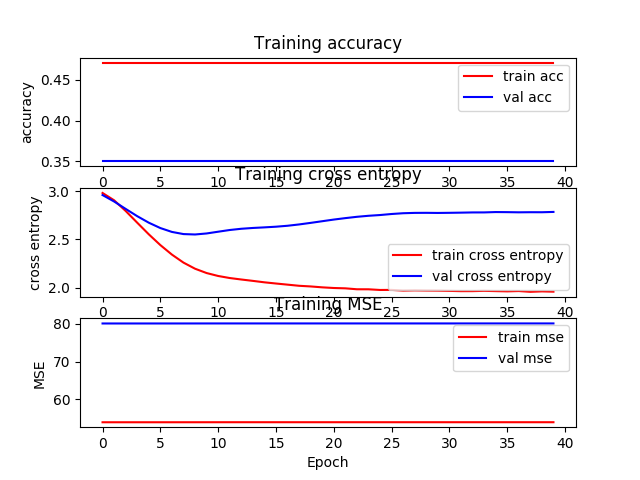
将relu改为tanh：没用

去掉LRN和dropout：没用

把网络结构换成LeNet：没用

把网络结构换成一个简单的mlp：没用

把网络结构换成11层的vggNet：没用



算了，先去下载个训练好的模型来进行下一步了，完事儿后再来纠结训练的问题。

4.**进行apache2配置时，在设置/etc/apache2/sites-enabled/000-default中的DocumentRoot为项目中的web后，访问localhost之后显示permission denied。**

解决办法为在/etc/apache2/apache2.conf中加入如下语句：

<Directory /home/hongyigeng/PycharmProjects/ImageSearch/>

Options Indexes FollowSymLinks

AllowOverride None

Require all granted

</Directory>

和这段语句不一样：

<Directory "/home/hongyigeng/PycharmProjects/ImageSearch/web">

AllowOverride None

Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch

Require all granted

AddHandler cgi-script cgi .py .pl .cgi

</Directory>

5.**访问Index.py时出现Internal Server Error，查看错误日志/var/log/apache2/error.log时看见如下语句：**

**[Mon Dec 24 12:36:46.649987 2018] [cgid:error] [pid 9465:tid 139988085394368] (13)Permission denied: AH01241: exec of '/home/hongyigeng/PycharmProjects/ImageSearch/web/Index.py' failed**

**[Mon Dec 24 12:36:46.650336 2018] [cgid:error] [pid 9049:tid 139987748124416] [client 127.0.0.1:42516] End of script output before headers: Index.py, referer:** [**http://localhost/**](http://localhost/)

解决办法：

这是对所要执行的脚本权限不够的原因，需要有执行权限，因此只需要对这个脚本赋予权限即可 sudo chmod 777 ~/PycharmProjects/ImageSearch/web/Index.py

6.**测试上传图片时，将路径从/tmp/改为其他路径时出现InternalServer Error，查看错误日志发现如下错误报告：[Mon Dec 24 14:56:43.236419 2018] [http:error] [pid 9048:tid 139987984267008] [client 127.0.0.1:46382] AH02429: Response header name '<!--' contains invalid characters, aborting request, referer:** [**http://localhost/html/**](http://localhost/html/)

使用https://docs.python.org/2.0/lib/node246.html这个网站上的方法可以输出python脚本出现的错误信息，得到的错误信息为：Traceback (most recent call last): File "/home/hongyigeng/PycharmProjects/ImageSearch/web/python/process.py", line 20, in open('/tmp/ImageSearch/' + fn, 'wb').write(fileitem.file.read()) IOError: [Errno 13] Permission denied: '/tmp/ImageSearch/2007\_000323.jpg'

7.**每次保存文件到tmp文件夹中都会转存到类似systemd-private的文件夹中**

解决办法：

这是因为目录名称中提到的服务启用了安全Tmp系统功能。

输入这个命令后：grep -R PrivateTmp /etc/systemd/可以看到哪些服务启用了PrivateTmp功能，之后编辑对应服务的配置文件，设置PrivateTmp=false之后，systemctl daemon-reload再重启服务即可。

8.**在网页上上传图片时，process.py报错，提示的尽是“No module named xxxx”或者是”No such file or directory”之类的错误。**

“No module named xxxx”的问题是因为sys.path没有设置好，python寻找的所有module都是从这些路径里面去找的。  
”No such file or directory”的问题所在是import的predictor脚本中的文件路径都写成了相对路径，测试的时候写的相对路径没问题，但是实际使用时是在不同的路径下，这样被import时导致找不到文件。

9.**在后台运行process.py时出现TclError: no display name and no $DISPLAY environment variable**

这个问题是因为matplotlib要切换成后端模式

# do this before importing pylab or pyplot

Import matplotlib

matplotlib.use('Agg')

**import** matplotlib.pyplot asplt

10.**多次上传发现图片并未改变**

这是因为图片的缓存机制，只要在src的后面加一个随机的参数，使得src每次都会不同，那么就不会从缓存中读取图片了，像这样image.attr('src', data.message+'?'+Math.random());

11.**git reset –hard之后想找回文件**

使用git reflog命令可以看到reset之前的一次操作，复制那个操作的id，git reset –hard 那个操作的id 就可以恢复了。

12.**cgi程序没有错误还是会出现Internal Server Error**

因为cgi程序没有运行权限，需要用chmod改一下权限。

13.**使用pickle.load出现EOFError**

因为pickle.load会一直读到文件的结尾，但是无法事先判断，百度后还是只能用try except来实现pickle文件的读取，觉得这样写很丑。。。