# Caffe的finetune及测试

下面的例子在bvlc\_reference\_caffenet.caffemodel模型的基础上进行finetune，对5个类别进行分类。

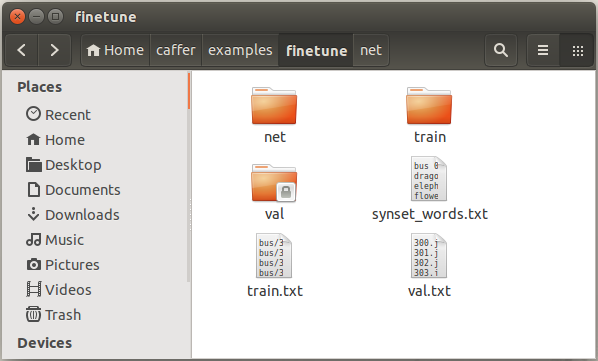
## 准备数据

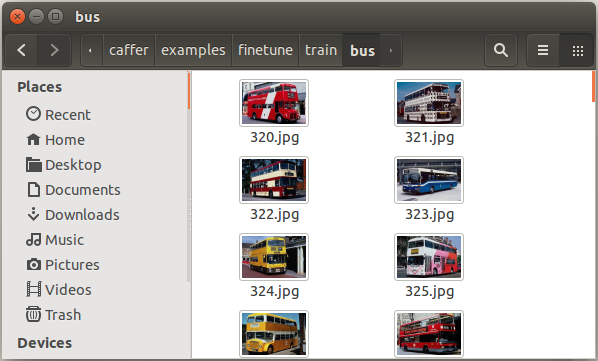
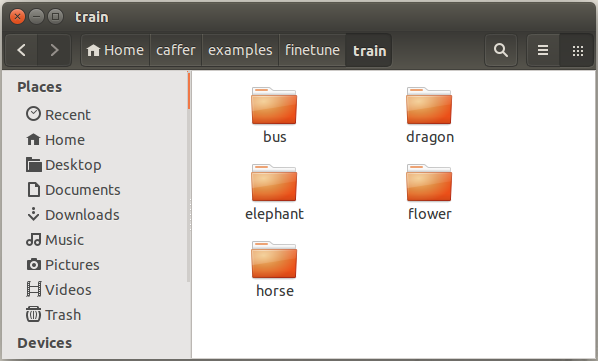
在examples目录创建一个finetune文件夹，再在其中创建3个子文件夹net,train,val，其中net用于存放与网络相关的文件，train存放训练图像，val存放测试图像，将训练及测试图像的相对路径及相应的标签保存到train.txt和val.txt文件中，synset\_words.txt保存标签与相应类名。

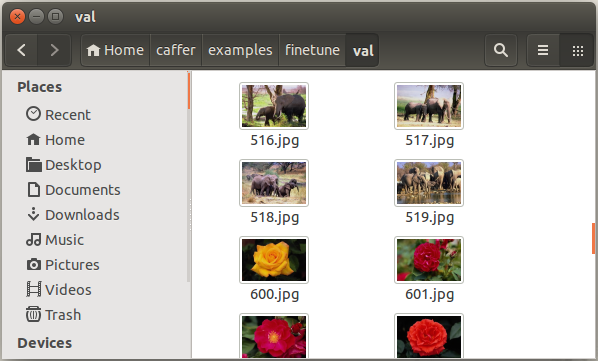
net文件夹虽的文件包括：

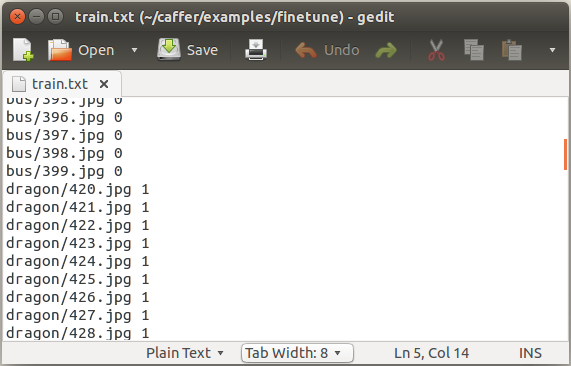
1. bvlc\_reference\_caffenet.caffemodel文件，1000类的网络参数模型文件，用于对模型参数初始化，拷贝自models/bvlc\_reference\_caffenet目录。
2. deploy.prototxt文件，1000类的测试网络文件，修改后用于我们的测试任务，拷贝自models/bvlc\_reference\_caffenet目录。
3. Solver.prototxt文件，网络超参数文件，修改后用于训练我们的网络，拷贝自models/bvlc\_reference\_caffenet目录。
4. train\_val.prototxt文件，1000类网络结构文件，修改后作为我们的网络，拷贝自models/bvlc\_reference\_caffenet目录。
5. create\_lmdb.sh文件，用于生成lmdb数据的脚本文件，拷贝修改自example/imagenet/create\_imagnet.sh
6. make\_finetune\_mean.sh文件，用于生成均值文件的脚本文件，拷贝修改自example/imagenet/make\_imagenet\_mean.sh
7. train\_caffenet.sh文件，用于训练网络的脚本文件，拷贝修改自example/imagenet/train\_caffenet.sh

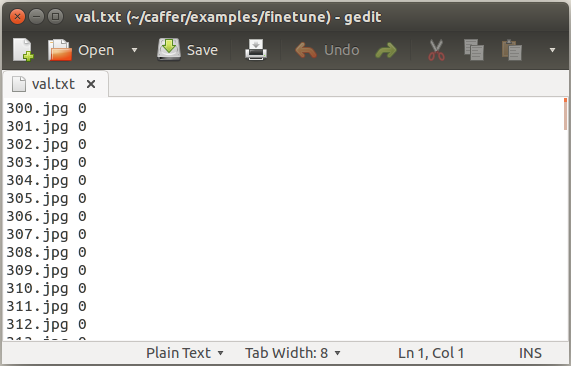
目录结构及文件内容可参考下图：

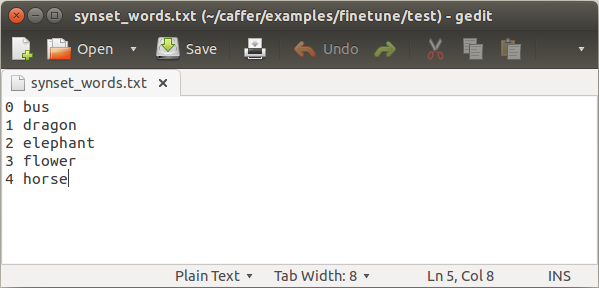


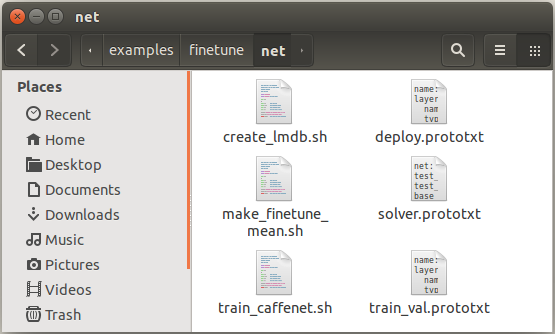




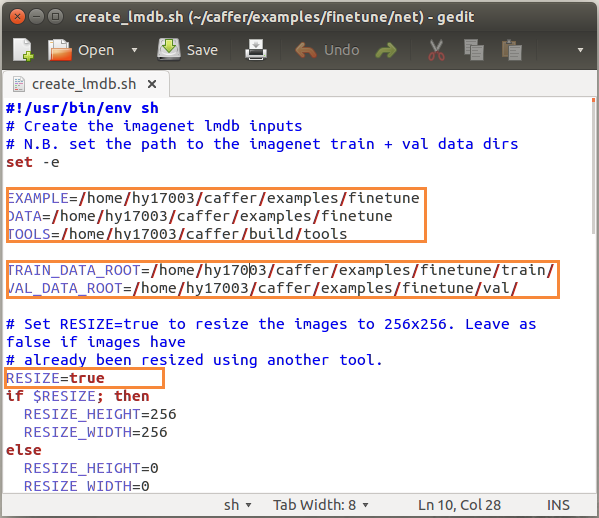


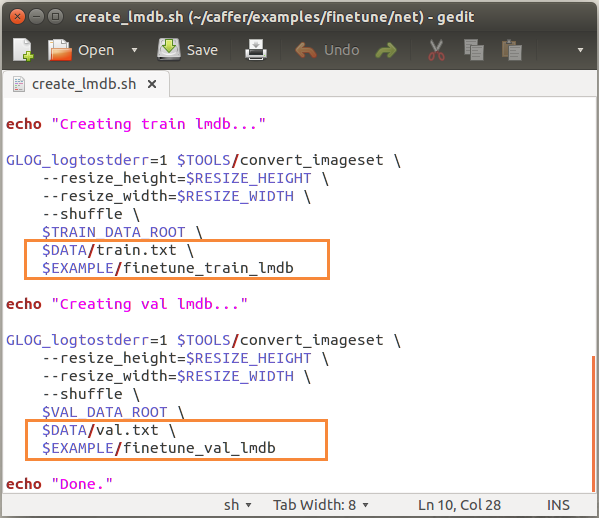


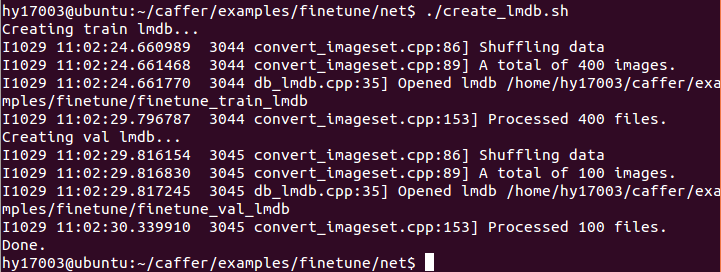


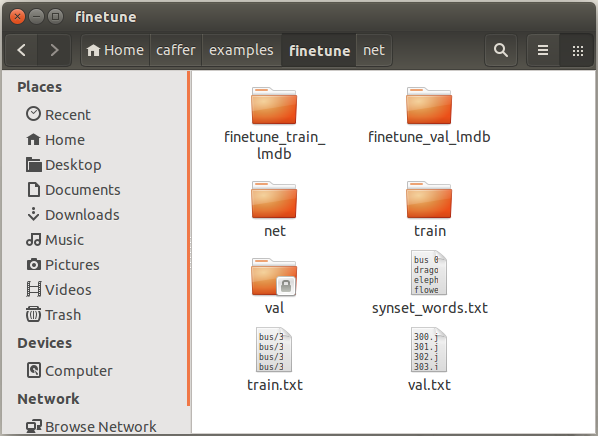


## 生成lmdb文件





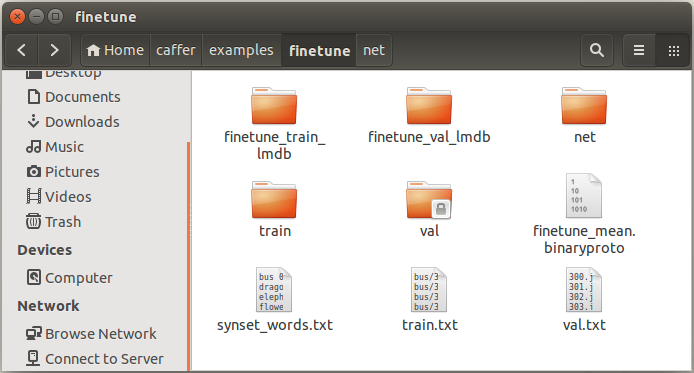




## 均值文件

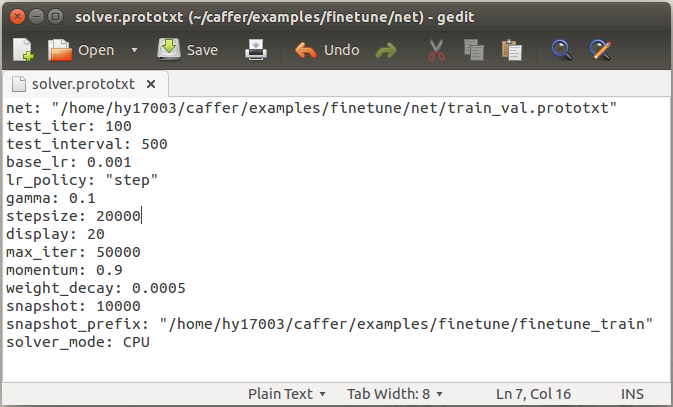




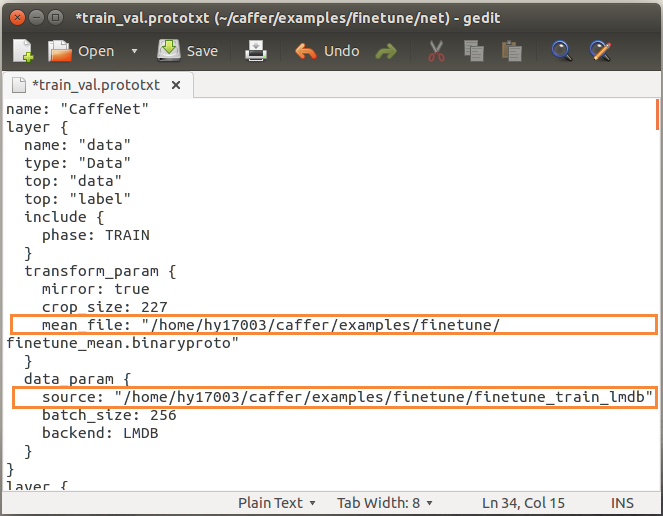


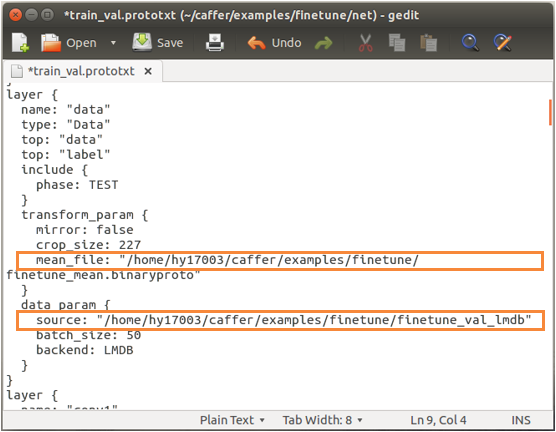
## 网络设置

### Solver.prototxt

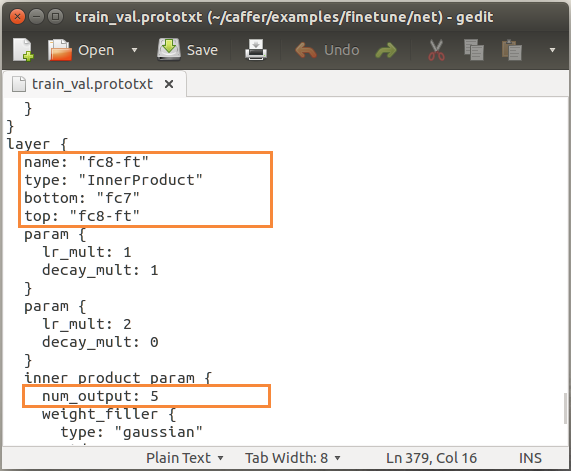


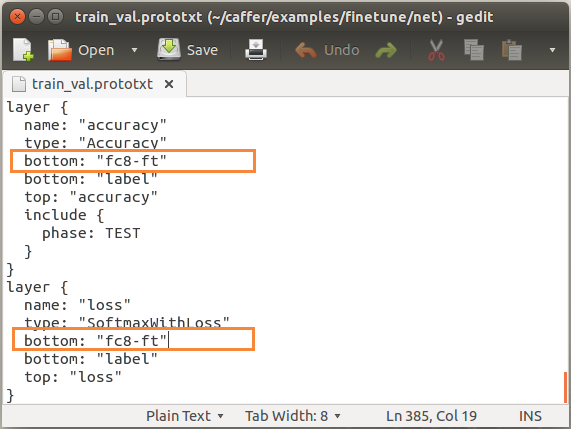
### train\_val.prototxt



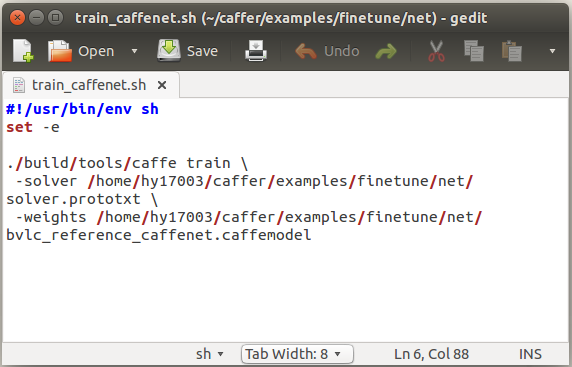


**注意！注意！注意：这里把全连接层的输出改成5，因为我们要对5个类进行分类，另外还要把最后一个全连接层改一个名字，这里改成“fc8-ft”,随意改，不能用原来的名字，因为在使用原有的训练好的模型对网络进行初始化的时候要拷贝原模型的参数，由于全连接层层已经改变，不能用原有的参数，所以，因而不能使用原有模型的全连接层名称。**

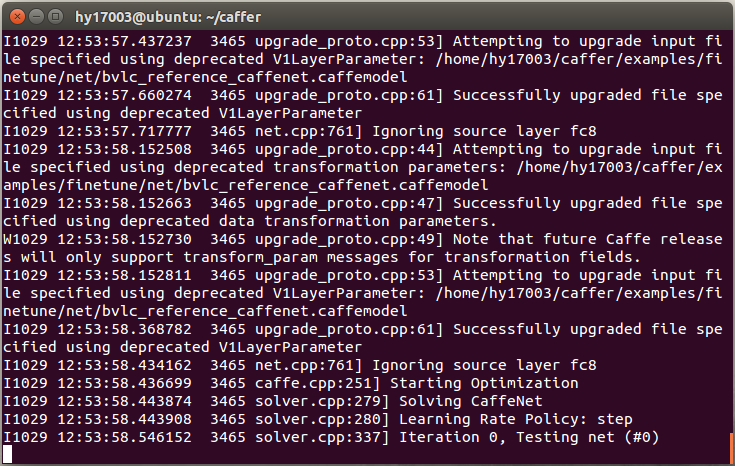




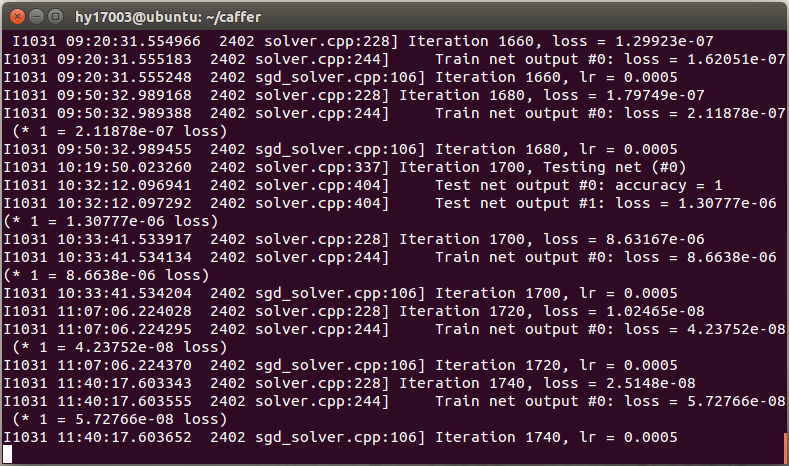
### train\_caffnet.sh

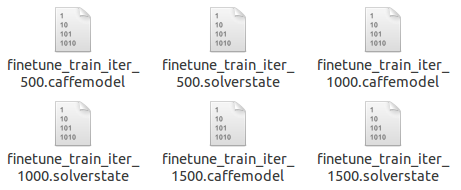


**注意，到caffe根目录下执行./example/finetune/net/train\_caffenet.sh**



训练时间十分漫长，经过近两天两夜的训练后，迭代到了1740次：



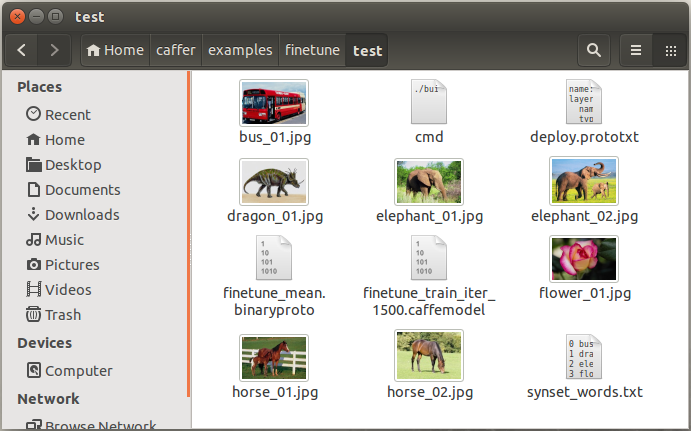


另，如果想把训练过程以日志的形式保存，以方便后续对训练过程可视化，可使用如下命令：**./example/finetune/net/train\_caffenet.sh >& train.log**

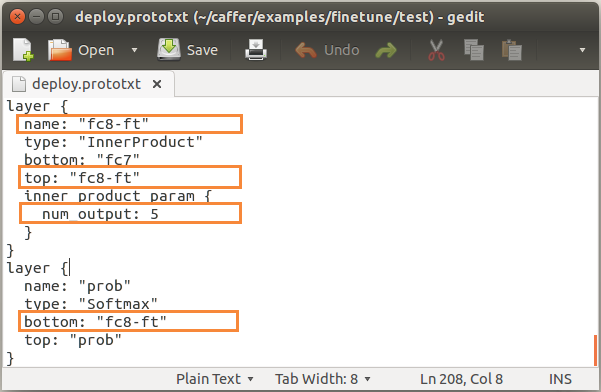
## 测试

取1500次迭代的结果进行测试。

新建一个文件夹test，将deploy.prototxt文件，finetune\_mean.binaryproto文件，finetune\_train\_iter\_1500.caffemodel文件，synset\_words.txt文件及部分测试图像文件拷贝到该文件夹下。

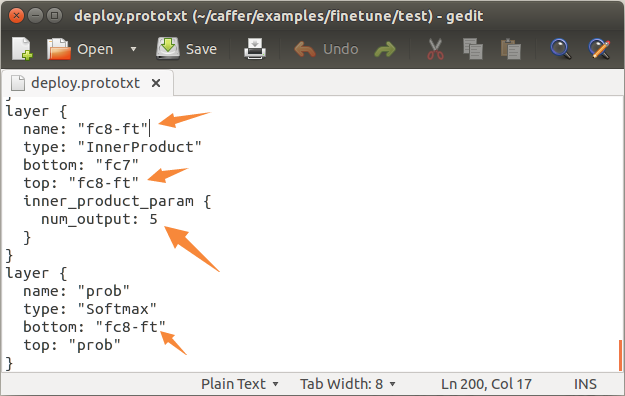


对deploy.prototxt文件进行修改:



### 使用caffe的classification工具测试

修改deploy.prototxt文件：

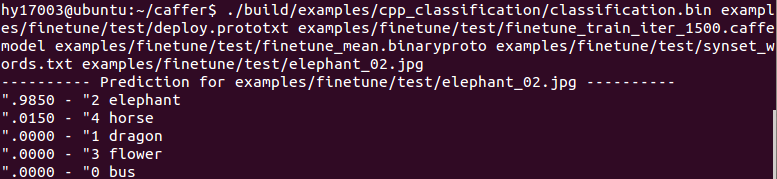


退到caffe根目录下，对elephant\_02.jpg图像进行测试，这里使用caffe自带的classification工具，执行以下命令：

./build/examples/cpp\_classification/classification.bin \ examples/finetune/test/deploy.prototxt \ examples/finetune/test/finetune\_train\_iter\_1500.caffemodel \ examples/finetune/test/finetune\_mean.binaryproto \ examples/finetune/test/synset\_words.txt \ examples/finetune/test/elephant\_02.jpg

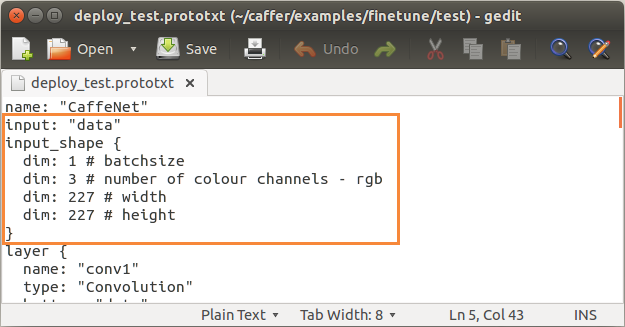


结果列出了前5项预测结果，其中elephanat的可能性为0.9850，预测结果完全正确！



### 编写代码进行测试

拷贝一份deploy.prototxt文件，重命名为deploy\_test.prototxt，删除其data层，添加图中代码：



创建一个finetuneTest.cpp文件，输入如下代码：

#include "opencv2/dnn.hpp"

#include "opencv2/imgproc.hpp"

#include "opencv2/highgui.hpp"

using namespace cv;

using namespace cv::dnn;

#include <fstream>

#include <iostream>

#include <cstdlib>

using namespace std;

/\* Find best class for the blob (i. e. class with maximal probability) \*/

void getMaxClass(dnn::Blob &probBlob, int \*classId, double \*classProb)

{

Mat probMat = probBlob.matRefConst().reshape(1, 1); //reshape the blob to 1x1000 matrix

Point classNumber;

minMaxLoc(probMat, NULL, classProb, NULL, &classNumber);

\*classId = classNumber.x;

}

std::vector<String> readClassNames(const char \*filename = "synset\_words.txt")

{

std::vector<String> classNames;

std::ifstream fp(filename);

if (!fp.is\_open())

{

std::cerr << "File with classes labels not found: " << filename << std::endl;

exit(-1);

}

std::string name;

while (!fp.eof())

{

std::getline(fp, name);

if (name.length())

classNames.push\_back( name.substr(name.find(' ')+1) );

}

fp.close();

return classNames;

}

int main(int argc, char \*\*argv)

{

cv::dnn::initModule(); //Required if OpenCV is built as static libs

String modelTxt = "deploy\_test.prototxt";

String modelBin = "finetune\_train\_iter\_1500.caffemodel";

String imageFile = "elephant\_01.jpg";

//! [Create the importer of Caffe model] 导入一个caffe模型接口

Ptr<dnn::Importer> importer;

importer = dnn::createCaffeImporter(modelTxt, modelBin);

if (!importer){

std::cerr << "Can't load network by using the following files: " << std::endl;

std::cerr << "prototxt: " << modelTxt << std::endl;

std::cerr << "caffemodel: " << modelBin << std::endl;

exit(-1);

}

//! [Initialize network] 通过接口创建和初始化网络

Net net;

importer->populateNet(net);

importer.release();

Mat img = imread(imageFile);

if (img.empty())

{

std::cerr << "Can't read image from the file: " << imageFile << std::endl;

exit(-1);

}

resize(img, img, Size(227, 227)); //GoogLeNet accepts only 224x224 RGB-images

dnn::Blob inputBlob = cv::dnn::Blob(img);

net.setBlob(".data", inputBlob); //set the network input

net.forward(); //compute output

dnn::Blob prob = net.getBlob("prob"); //gather output of "prob" layer

int classId;

double classProb;

getMaxClass(prob, &classId, &classProb);//find the best class

std::vector<String> classNames = readClassNames();

std::cout << "Best class: #" << classId << " '" << classNames.at(classId) << "'" << std::endl;

std::cout << "Probability: " << classProb \* 100 << "%" << std::endl;

return 0;

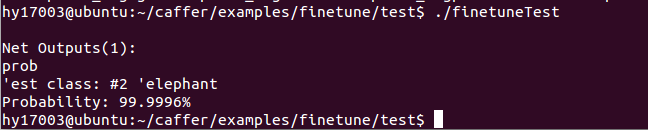
}

编译：

hy17003@ubuntu:~/caffer/examples/finetune/test$ g++ -o finetuneTest finetuneTest.cpp -lopencv\_dnn -lopencv\_highgui -lopencv\_imgcodecs -lopencv\_imgproc -lstdc++ -lopencv\_core -L/usr/local/lib

运行：

hy17003@ubuntu:~/caffer/examples/finetune/test$ ./finetuneTest



在这段代码中，没有使用到均值文件，而在5.1中却使用到，不知道此处如何使用均值文件。