# 使用OpenCV和Caffe结合进行分类

本实例在http://docs.opencv.org/trunk/d5/de7/tutorial\_dnn\_googlenet.html上进行改写，因为我按照原实例进行实验时出现一些问题，经修改后运行正确，把步骤记下来，供以后参考。

这里使用OpenCV的opencv\_dnn模块与Caffe model zoo中已训练好的GoogLeNet网络进行分类，使用的测试图像如下：



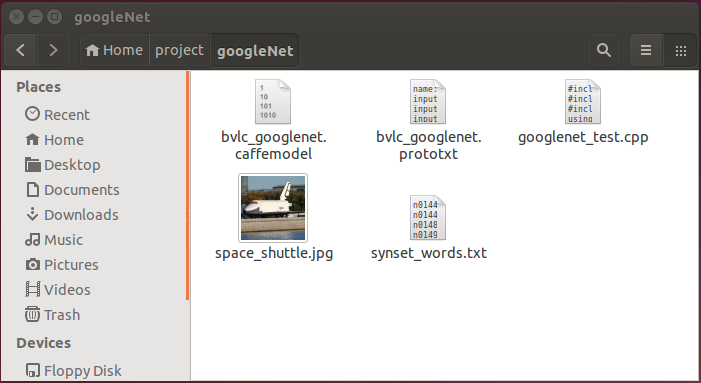
首先下载GoogLeNet模型文件，bvlc\_googlenet.prototxt和bvlc\_googlenet.caffemodel以及类名文件synset\_words.txt，下载地址为：

<https://raw.githubusercontent.com/ludv1x/opencv_contrib/master/modules/dnn/samples/bvlc_googlenet.prototxt>

<http://dl.caffe.berkeleyvision.org/bvlc_googlenet.caffemodel>

<https://raw.githubusercontent.com/ludv1x/opencv_contrib/master/modules/dnn/samples/synset_words.txt>

下载后将文件与测试图像放在同一文件夹下：



googlenet\_test.cpp是测试代码文件，代码如下：

#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 = "bvlc\_googlenet.prototxt";

String modelBin = "bvlc\_googlenet.caffemodel";

String imageFile = (argc > 1) ? argv[1] : "space\_shuttle.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;

std::cerr << "bvlc\_googlenet.caffemodel can be downloaded here:" << std::endl;

std::cerr << "http://dl.caffe.berkeleyvision.org/bvlc\_googlenet.caffemodel" << 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(224, 224)); //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：

g++ -o googlenet\_test googlenet\_test.cpp -lopencv\_dnn -lopencv\_highgui -lopencv\_imgcodecs -lopencv\_imgproc -lstdc++ -lopencv\_core -L/usr/local/lib

运行：

./googlenet\_test

结果：

