# 一：介绍

必须安装好OpenCV。

SeetaFace人脸识别引擎包括了搭建一套全自动人脸识别系统所需的三个核心模块，

即：人脸检测模块SeetaFace Detection、面部特征点定位模块SeetaFace Alignment以及人脸特征提取与比对模块 SeetaFace Identification。

主要功能：

1.人脸检测模块(SeetaFace Detection): 采用了一种结合传统人造特征与多层感知机（MLP）的级联结构，在FDDB上达到了84.4%的召回率（100个误检时），并可在单个i7 CPU上实时处理VGA分辨率的图像。

2.面部特征点定位模块(SeetaFace Alignment): 通过级联多个深度模型（栈式自编码网络）来回归5个关键特征点（两眼中心、鼻尖和两个嘴角）的位置，在AFLW数据库上达到state-of-the-art的精度，定位速度在单个i7 CPU上超过200fps。

3.人脸识别模块(SeetaFace Identification): 采用一个9层的卷积神经网络（CNN）来提取人脸特征，在LFW数据库上达到97.1%的精度（注：采用SeetaFace人脸检测和SeetaFace面部特征点定位作为前端进行全自动识别的情况下），特征提取速度为每图120ms（在单个i7 CPU上）。

# 二.下载

1）<https://github.com/seetaface/SeetaFaceEngine>

unzip SeetaFaceEngine-master.zip

2) 下载cmake

需要保证cmake版本在3.1以上。

<https://cmake.org/download/> 下载了Linux x86\_64

tar zxvf cmake-3.1.3-Linux-i386.tar.gz #解压自己下的版本

在bin目录下有cmake的可执行文件，可以直接使用

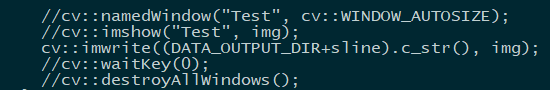
# 三、编译

## 3.1 编译测试FaceDetection

1.修改代码，不然会报段错误（可以使用Xshell，不会出问题）

[root@host233186 FaceDetection]# vi src/test/facedetection\_test.cpp

std::string DATA\_OUTPUT\_DIR = "./data/output/"; //新增



2.编译

[root@host233186 FaceDetection]#mkdir build

[root@host233186 FaceDetection]#cd build

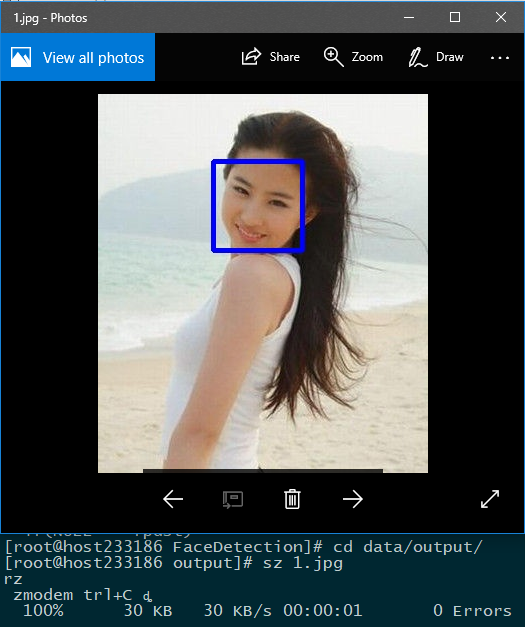
[root@host233186 build]# ../../cmake-3.9.1-Linux-x86\_64/bin/cmake ..

[root@host233186 build]# make

//test

[root@host233186 FaceDetection]./build/facedet\_test imagefilePath ./model/seeta\_fd\_frontal\_V1.0.bin #imagefilePath是图片的路径

3.查看处理后的图片



## 3.2 编译测试FaceAlignment

1.编译之前，将FaceDetection中的/include/face\_detection.h和/build/libseeta\_facedet\_lib.so

拷贝到/FaceAlignment/build文件夹下。

然后在FaceAlignment>src>test中face\_alignment\_test.cpp中

|  |
| --- |
| int main(int argc, char\*\* argv)  {  // Initialize face detection model  seeta::FaceDetection detector("./build/seeta\_fd\_frontal\_v1.0.bin"); //修改路径  //修改成这个路径，把seeta\_fd\_frontal\_v1.0.bin放到build中。原版是基于windows的。  detector.SetMinFaceSize(40);  detector.SetScoreThresh(2.f);  detector.SetImagePyramidScaleFactor(0.8f);  detector.SetWindowStep(4, 4); |

2.

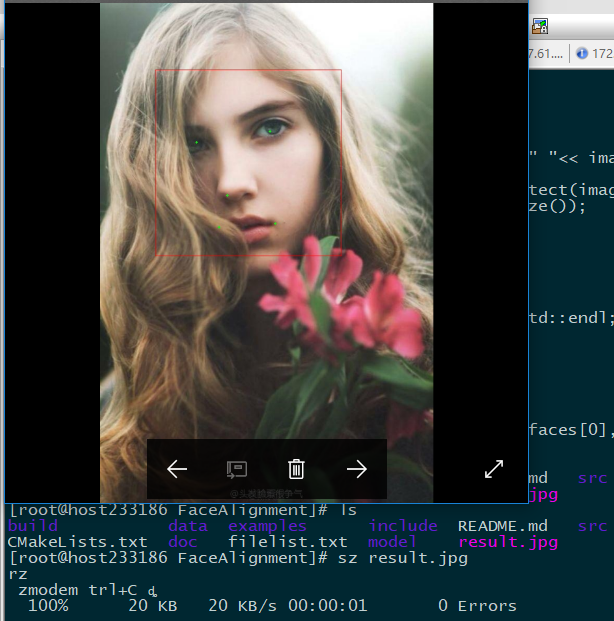
[root@host233186 FaceAlignment]#mkdir build

[root@host233186 FaceAlignment]#cmake ..

[root@host233186 build]#make

//test

[root@host233186 FaceAlignment]# ./build/fa\_test



## 3.1 编译测试FaceIndentification

1）添加文件

将libseeta\_facedet\_lib.so（/FaceDetection/build），libseeta\_fa\_lib.so（FaceAlignment/build），

seeta\_fd\_frontal\_v1.0.bin（/FaceDetection/model），seeta\_fa\_v1.1.bin（FaceAlignment/model），添加到FaceIndentification/build中

将face\_alignment.h（FaceAlignment/include），face\_detection.h（/FaceDetection/include）添加到FaceIndentification/include中

2）修改文件

打开FaceIdentification/src/test/CMakeLists.txt将文件替换成如下内容

|  |
| --- |
| aux\_source\_directory (. SRC\_LIST)  link\_directories(${PROJECT\_BINARY\_DIR})  message(${SRC\_LIST})  # add external libraries  find\_package(OpenCV REQUIRED)  include\_directories(${seeta\_facedet\_lib\_INCLUDE\_DIRS} ${seeta\_fa\_lib\_INCLUDE\_DIRS})  list(APPEND seeta\_fi\_lib\_required\_libs ${OpenCV\_LIBS} seeta\_facedet\_lib seeta\_fa\_lib)  enable\_testing ()  foreach (f ${SRC\_LIST})    string(REGEX REPLACE "[.]cpp" ".bin" BIN ${f})    add\_executable(${BIN} ${f})    #target\_link\_libraries(${BIN} viplnet ${OpenCV\_LIBS} seeta\_facede\_lib seeta\_fa\_lib)    target\_link\_libraries(${BIN} viplnet ${seeta\_fi\_lib\_required\_libs})  endforeach () |

在 FaceIdentification/src/test/test\_face\_recognizer.cpp文件中添加如下内容头文件

|  |
| --- |
| #include <opencv2/opencv.hpp>  #include <opencv2/highgui/highgui.hpp>  #include <opencv2/imgproc/imgproc.hpp> |

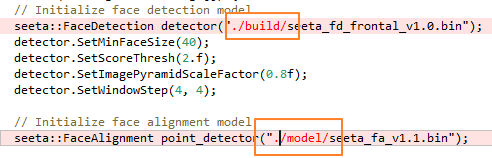
在 FaceIdentification/src/test/test\_face\_verification.cpp文件中添加如下内容头文件

|  |
| --- |
| #include <opencv2/opencv.hpp>  #include <opencv2/highgui/highgui.hpp>  #include <opencv2/imgproc/imgproc.hpp> |

解压 FaceIdentification/model/seeta\_fr\_v1.0.part1.rar，并上传

clipboard.png

修改代码[root@host233186 FaceIdentification]# vi src/test/test\_face\_verification.cpp（很重要，否则会引起段错误）：



3）编译

mkdir build cd build cmake .. make #如出错，请检查文件添加好

4）运行

./build/src/test/test\_face\_recognizer.bin #3个单元测试函数 ./build/src/test/test\_face\_verification.bin #比较两个图像相似度

效果：检测的是FaceIdentification/data/test\_face\_recognizer/images/compare\_im下的图片相似度

0.680086

改进：

1.增加一个文件Test\_face\_feature.cpp，增加一个文件列表，批量处理文件列表中的所有文件，输出每张图片的特征。.

std::string DATA\_DIR = "./data/";

std::string DATA\_OUTPUT\_DIR = "./data/output/";

增加代码段：

//argv[1] is a filelist

FILE \*fpdst = fopen(argv[1],"r");

if(NULL == fpdst)

{

fprintf(stderr,"Error at create file:%s\n",argv[1]);

return 0;

}

char sline[1024];

while(fgets(sline,sizeof(sline),fpdst))

{

if(NULL == sline)

break;

sline[strlen(sline)-1] = 0;

std::cout<<sline<<std::endl;

const char\* img\_path = (DATA\_DIR + sline).c\_str();

......................................code.....................................

cv::imwrite((DATA\_OUTPUT\_DIR+sline).c\_str(), img);

}

fclose(fpdst);

修改后的版本，这样运行

./build/src/test/test\_face\_feature.bin ./data/test\_face\_recognizer/zhaoliying.txt ./dat/zhaoliying.dat ./dat/zhaoliying.list