# 基于python—opencv的人脸识别（修改版）

这篇用python语言开发的人脸识别的原文是国外一位大佬基于开源计算机视觉库opencv开发的，本人就基于源代码上稍微修改了一点点。这里我就没有再详细的写关于opencv的安装跟程序的详解，因为给出的原文的链接有说明，建议看此文章前可以看看。

不过值得一提的是将程序从我的笔记本上部署到树莓派，在树莓派上安装opencv时，我部署环境遇到的困难真的是一言难尽（基于python3.7），如果有人也想将程序部署到树莓派或其他终端上时也遇到安装opencv报错一类的问题可以留言互动，我尽我所能解答。

我将原来的代码文件与修改过的文件放在了我的GitHub上，欢迎有人修改或指明代码中的不足之处。

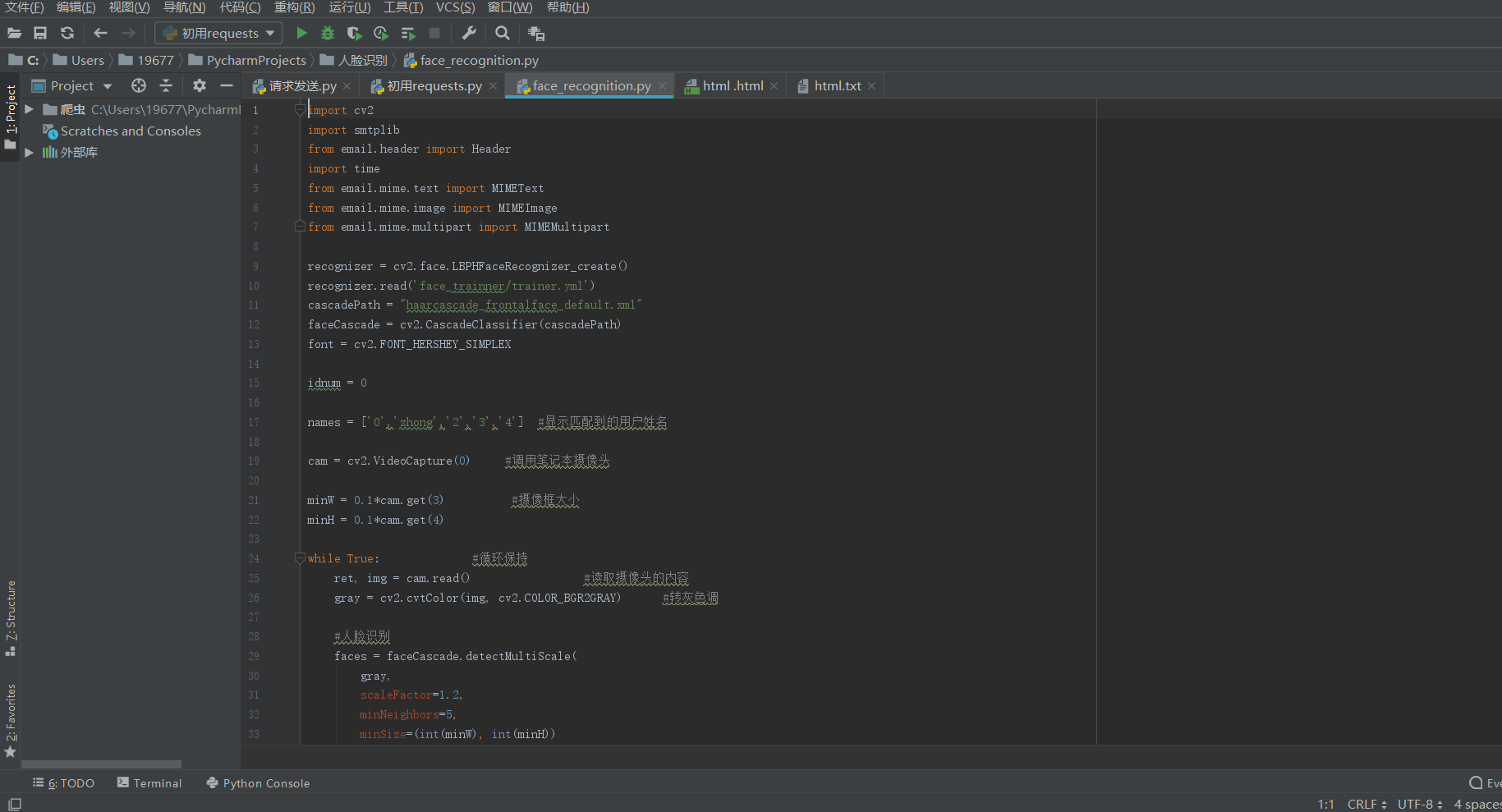
原文章

基于opencv的人脸识别

我的github地址：

添加的内容：

1. 若识别到未知人员的面孔，则拍下一张照片
2. 用时间戳命名相片保存到文件夹下。
3. 加入第三方邮件smtp库，利用qq邮箱来发送邮件，邮件中有刚排下人员的照片。



原文的程序有几个，我修改的程序文件为：face\_recognition.py

import cv2

import smtplib

from email.header import Header

import time

from email.mime.text import MIMEText

from email.mime.image import MIMEImage

from email.mime.multipart import MIMEMultipart

recognizer = cv2.face.LBPHFaceRecognizer\_create()

recognizer.read('face\_trainner/trainer.yml') #注意路径文件夹

cascadePath = "haarcascade\_frontalface\_default.xml" #注意路径

faceCascade = cv2.CascadeClassifier(cascadePath)

font = cv2.FONT\_HERSHEY\_SIMPLEX

idnum = 0

names = ['0','zhong','2','3','4'] #显示匹配到的用户姓名

cam = cv2.VideoCapture(0) #调用笔记本摄像头

minW = 0.1\*cam.get(3) #摄像框大小

minH = 0.1\*cam.get(4)

while True: #循环保持

ret, img = cam.read() #读取摄像头的内容

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY) #转灰色调

#人脸识别

faces = faceCascade.detectMultiScale(

gray,

scaleFactor=1.2,

minNeighbors=5,

minSize=(int(minW), int(minH))

)

for (x, y, w, h) in faces:

cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)

idnum, confidence = recognizer.predict(gray[y:y+h, x:x+w])

if confidence < 60: #如果相似值小于一百

idnum = names[idnum] #输出显示相似的用户名

confidence = "{0}%".format(round(100 - confidence))

else:

idnum = "unknown"

confidence = "{0}%".format(round(100 - confidence))

if idnum == "unknown":

#拍摄一张照片并储存，以便发送到邮箱

print("数据库人脸无匹配....正在发送邮件告知....")

now = time.strftime("%Y-%m-%d-%H\_%M\_%S")

cv2.imwrite('img/img'+ now +'.jpg', img)

# 第三方 SMTP 服务

msg = MIMEMultipart('related')

msg\_str = """

<p>发现未知人员....</p>

<img src="cid:123">

"""

msg.attach(MIMEText(msg\_str, 'html', 'utf-8'))

with open('img/img'+ now +'.jpg', 'rb') as f:

pic = MIMEImage(f.read())

pic.add\_header('Content-ID', '<123>')

msg.attach(pic)

msg['From'] = '邮箱@qq.com'

msg['To'] = '邮箱@qq.com'

msg['Subject'] = Header("树莓派人脸入侵检测系统",'utf-8').encode()

smtp = smtplib.SMTP\_SSL('smtp.qq.com')

smtp.login('发送邮箱', '密码开起的smtp服务的密码随机的')

smtp.sendmail('发送邮箱@qq.com', '接收邮箱@qq.com', msg.as\_string())

print("邮件发送成功")

time.sleep(5)

cv2.putText(img, str(idnum), (x+5, y-5), font, 1, (0, 0, 255), 1) #输出显示相似的用户名

cv2.putText(img, str(confidence), (x+5, y+h-5), font, 1, (0, 0, 0), 1) #输出相似度

cv2.imshow('camera', img) #保持画面

k = cv2.waitKey(10) #按esc退出

if k == 27:

break

cam.release() #关闭摄像头

cv2.destroyAllWindows()

运行结果：

可以看到能识别出人脸

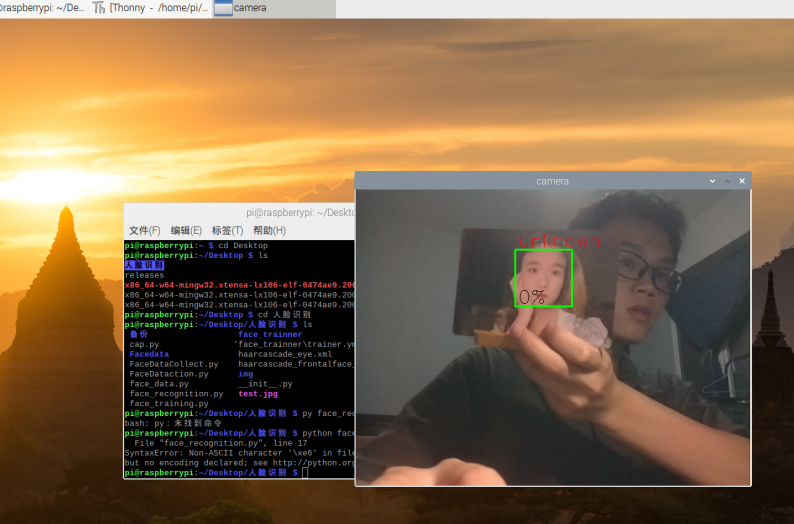


程序已经部署在树莓派上了，注意图的右下角



树莓派4

也能识别出陌生人脸



若识别到陌生人脸，则发送邮件通知（注意绿框中的陌生人脸）



当然，缺点是：

1.更改后运行人脸识别程序，一旦在摄像头范围内捕捉到陌生面孔，设置好的邮箱将收到邮件告警，但是邮件几乎是在不停的发送，原本设置了时间间隔，但是原本的识别对比那段程序是一个while循环，无奈本人的编程能力还是有限，不敢大动刀子修改程序。只在发邮件代码后面加了时间间隔time.sleep(),但会在循环中卡顿一会。

2.误判几率交高，可能是opencv的算法原因吧，且在两人在同一镜头下，陌生人还是会有一定几率显示已存在的人脸数据。目前需要的结果就到这里了，如果想要精确的人脸识别估计还得在算法上下功夫。