写个神经网络 让她认得我`(๑•ᴗ•๑)(Tensorflow,opencv,dlib,cn人脸识别)

2018-05-07

Deep learning python

显示目录

文章目录

训练一个神经网络 能让她认得我

这段时间正在学习tensorflow的卷积神经网络部分，为了对卷积神经网络能够有一个更深的了解，自己动手实现一个例程是比较好的方式，所以就选了一个这样比较有点意思的项目。

项目的github地址: github 喜欢的话就给个Star吧。

想要她认得我，就需要给她一些我的照片，让她记住我的人脸特征，为了让她区分我和其他人，还需要给她一些其他人的照片做参照，所以就需要两组数据集来让她学习，如果想让她多认识几个人，那多给她几组图片集学习就可以了。下面就开始让我们来搭建这个能认识我的”她”。

运行环境

下面为软件的运行搭建系统环境。

系统: window或linux

软件: python 3.x 、 tensorflow

python支持库:

tensorflow:

```
pip install tensorflow      #cpu
pip install rensorflow-gpu  #gpu
```

numpy:

```
pip install numpy
```

opencv:

```
pip install opencv-python
```

dlib:

```
pip install dlib
```

获取本人图片集

获取本人照片的方式当然是拍照了，我们需要通过程序来给自己拍照，如果你自己有照片，也可以用那些现成的照片，但前提是你的照片足够多。这次用到的照片数是10000张，程序运行后，得坐在电脑面前不停得给自己的脸摆各种姿势，这样可以提高训练后识别自己的成功率，在程序中加入了随机改变对比度与亮度的模块，也是为了提高照片样本的多样性。

程序中使用的是dlib来识别人脸部分，也可以使用opencv来识别人脸，在实际使用过程中，dlib的识别效果比opencv的好，但opencv识别的速度会快很多，获取10000张人脸照片的情况

下，dlib大约花费了1小时，而opencv的花费时间大概只有20分钟。opencv可能会识别一些奇怪的部分，所以综合考虑之后我使用了dlib来识别人脸。

get_my_faces.py

```
import cv2
import dlib
import os
import sys
import random

output_dir = './my_faces'
size = 64

if not os.path.exists(output_di
    os.makedirs(output_dir)

# 改变图片的亮度与对比度
def relight(img, light=1, bias=0
    w = img.shape[1]
    h = img.shape[0]
    #image = []
    for i in range(0,w):
        for j in range(0,h):
            for c in range(3):
                tmp = int(img[j
                if tmp > 255:
                    tmp = 255
                elif tmp < 0:
                    tmp = 0
                img[j,i,c] = tmp
    return img

#使用dlib自带的frontal_face_detec
detector = dlib.get_frontal_face
# 打开摄像头 参数为输入流，可以为摄像
camera = cv2.VideoCapture(0)
```

```
index = 1
while True:
    if (index <= 10000):
        print('Being processed p
        # 从摄像头读取照片
        success, img = camera.re
        # 转为灰度图片
        gray_img = cv2.cvtColor
        # 使用detector进行人脸检测
        dets = detector(gray_img

        for i, d in enumerate(de
            x1 = d.top() if d.to
            y1 = d.bottom() if d
            x2 = d.left() if d.l
            y2 = d.right() if d

            face = img[x1:y1,x2
            # 调整图片的对比度与亮度
            face = relight(face,

            face = cv2.resize(fa

            cv2.imshow('image',

            cv2.imwrite(output_d

            index += 1
        key = cv2.waitKey(30) &
        if key == 27:
            break
    else:
        print('Finished!')
        break
```

在这里我也给出一个opencv来识别人脸的代码示例：

```
import cv2
import os
```

```
import sys
import random

out_dir = './my_faces'
if not os.path.exists(out_dir):
    os.makedirs(out_dir)


# 改变亮度与对比度
def relight(img, alpha=1, bias=0
    w = img.shape[1]
    h = img.shape[0]
    #image = []
    for i in range(0,w):
        for j in range(0,h):
            for c in range(3):
                tmp = int(img[j
                if tmp > 255:
                    tmp = 255
                elif tmp < 0:
                    tmp = 0
                img[j,i,c] = tmp
    return img


# 获取分类器
haar = cv2.CascadeClassifier('ha

# 打开摄像头 参数为输入流，可以为摄像
camera = cv2.VideoCapture(0)

n = 1
while 1:
    if (n <= 10000):
        print('It`s processing %
        # 读帧
        success, img = camera.re

        gray_img = cv2.cvtColor
        faces = haar.detectMult
```

```
        for f_x, f_y, f_w, f_h
            face = img[f_y:f_y+
            face = cv2.resize(fa
            '''
            if n % 3 == 1:
                face = relight(
            elif n % 3 == 2:
                face = relight(
            '''
            face = relight(face
            cv2.imshow('img', fa
            cv2.imwrite(out_dir+
            n+=1
        key = cv2.waitKey(30) &
        if key == 27:
            break
    else:
        break
```

获取其他人脸图片集

需要收集一个其他人脸的图片集，只要不是自己的人脸都可以，可以在网上找到，这里我给出一个我用到的图片集：

网站地址:

http://vis-www.cs.umass.edu/lfw/

图片集下载:

http://vis-www.cs.umass.edu/lfw/lfw.tgz

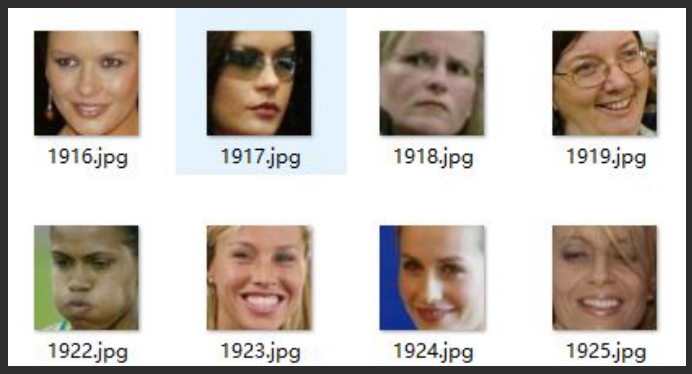
先将下载的图片集，解压到项目目录下的input_img目录下，也可以自己指定目录(修改代码中的input_dir变量)

接下来使用dlib来批量识别图片中的人脸部分，并保存到指定目录下

set_other_people.py

```
# -*- codeing: utf-8 -*-
import sys
import os
import cv2
import dlib

input_dir = './input_img'
output_dir = './other_faces'
size = 64

if not os.path.exists(output_di
    os.makedirs(output_dir)

#使用dlib自带的frontal_face_detec
detector = dlib.get_frontal_face

index = 1
for (path, dirnames, filenames)
    for filename in filenames:
        if filename.endswith('.j
         print('Being processed
            img_path = path+'/'+
            # 从文件读取图片
            img = cv2.imread(img
            # 转为灰度图片
            gray_img = cv2.cvtCo
            # 使用detector进行人脸
            dets = detector(gray

            #使用enumerate 函数遍
            #下标i即为人脸序号
            #left：人脸左边距离图片
            #top：人脸上边距离图片上
            for i, d in enumerat
                x1 = d.top() if
                y1 = d.bottom()
                x2 = d.left() if
                y2 = d.right() i
```

```
            # img[y:y+h,x:x+
            face = img[x1:y1
            # 调整图片的尺寸
            face = cv2.resiz
            cv2.imshow('imag
            # 保存图片
            cv2.imwrite(outp
            index += 1

        key = cv2.waitKey(30
        if key == 27:
            sys.exit(0)
```



这个项目用到的图片数是10000张左右，如果是自己下载的图片集，控制一下图片的数量避免数量不足，或图片过多带来的内存不够与运行缓慢。

训练模型

有了训练数据之后，通过cnn来训练数据，就可以让她记住我的人脸特征，学习怎么认识我了。

train_faces.py

```
import tensorflow as tf
import cv2
import numpy as np
import os
import random
import sys
from sklearn.model_selection im
```

```
my_faces_path = './my_faces'
other_faces_path = './other_face
size = 64

imgs = []
labs = []

def getPaddingSize(img):
    h, w, _ = img.shape
    top, bottom, left, right =
    longest = max(h, w)

    if w < longest:
        tmp = longest - w
        # //表示整除符号
        left = tmp // 2
        right = tmp - left
    elif h < longest:
        tmp = longest - h
        top = tmp // 2
        bottom = tmp - top
    else:
        pass
    return top, bottom, left, r

def readData(path , h=size, w=s
    for filename in os.listdir(
        if filename.endswith('.
            filename = path + '

            img = cv2.imread(fi

            top,bottom,left,rig
            # 将图片放大,  扩充图片
            img = cv2.copyMakeB
            img = cv2.resize(im

            imgs.append(img)
            labs.append(path)
```

```
readData(my_faces_path)
readData(other_faces_path)
# 将图片数据与标签转换成数组
imgs = np.array(imgs)
labs = np.array([[0,1] if lab ==
# 随机划分测试集与训练集
train_x,test_x,train_y,test_y =
# 参数：图片数据的总数，图片的高、宽、
train_x = train_x.reshape(train_
test_x = test_x.reshape(test_x.
# 将数据转换成小于1的数
train_x = train_x.astype('float
test_x = test_x.astype('float32

print('train size:%s, test size
# 图片块，每次取100张图片
batch_size = 100
num_batch = len(train_x) // bat

x = tf.placeholder(tf.float32,
y_ = tf.placeholder(tf.float32,

keep_prob_5 = tf.placeholder(tf
keep_prob_75 = tf.placeholder(t

def weightVariable(shape):
    init = tf.random_normal(sha
    return tf.Variable(init)

def biasVariable(shape):
    init = tf.random_normal(sha
    return tf.Variable(init)

def conv2d(x, W):
    return tf.nn.conv2d(x, W, s

def maxPool(x):
    return tf.nn.max_pool(x, ks
```

```
def dropout(x, keep):
    return tf.nn.dropout(x, kee

def cnnLayer():
    # 第一层
    W1 = weightVariable([3,3,3,
    b1 = biasVariable([32])
    # 卷积
    conv1 = tf.nn.relu(conv2d(x
    # 池化
    pool1 = maxPool(conv1)
    # 减少过拟合，随机让某些权重不更
    drop1 = dropout(pool1, keep

    # 第二层
    W2 = weightVariable([3,3,32
    b2 = biasVariable([64])
    conv2 = tf.nn.relu(conv2d(d
    pool2 = maxPool(conv2)
    drop2 = dropout(pool2, keep

    # 第三层
    W3 = weightVariable([3,3,64
    b3 = biasVariable([64])
    conv3 = tf.nn.relu(conv2d(d
    pool3 = maxPool(conv3)
    drop3 = dropout(pool3, keep

    # 全连接层
    Wf = weightVariable([8*16*32
    bf = biasVariable([512])
    drop3_flat = tf.reshape(dro
    dense = tf.nn.relu(tf.matmu
    dropf = dropout(dense, keep

    # 输出层
    Wout = weightVariable([512,2
    bout = weightVariable([2])
    #out = tf.matmul(dropf, Wou
    out = tf.add(tf.matmul(drop
```

```
    return out

def cnnTrain():
    out = cnnLayer()

    cross_entropy = tf.reduce_me

    train_step = tf.train.AdamOp
    # 比较标签是否相等，再求的所有数的
    accuracy = tf.reduce_mean(tf
    # 将loss与accuracy保存以供tens
    tf.summary.scalar('loss', cr
    tf.summary.scalar('accuracy
    merged_summary_op = tf.summa
    # 数据保存器的初始化
    saver = tf.train.Saver()

    with tf.Session() as sess:

        sess.run(tf.global_varia

        summary_writer = tf.summ

        for n in range(10):
             # 每次取128(batch_si
            for i in range(num_b
                batch_x = train_
                batch_y = train_
                # 开始训练数据，同
                _,loss,summary =

                summary_writer.a
                # 打印损失
                print(n*num_batc

                if (n*num_batch+
                    # 获取测试数据
                    acc = accura
                    print(n*num_
                    # 准确率大于0
```

```
                    if acc > 0.9
                        saver.sa
                        sys.exit
        print('accuracy less 0.9

cnnTrain()
```

训练之后的数据会保存在当前目录下。

使用模型进行识别

最后就是让她认识我了，很简单，只要运行程序，让摄像头拍到我的脸，她就可以轻松地识别出是不是我了。

is_my_face.py

```
output = cnnLayer()
predict = tf.argmax(output, 1)

saver = tf.train.Saver()
sess = tf.Session()
saver.restore(sess, tf.train.la

def is_my_face(image):
    res = sess.run(predict, feed
    if res[0] == 1:
        return True
    else:
        return False

#使用dlib自带的frontal_face_detec
detector = dlib.get_frontal_face

cam = cv2.VideoCapture(0)

while True:
    _, img = cam.read()
    gray_image = cv2.cvtColor(im
    dets = detector(gray_image,
```

```
    if not len(dets):
        #print('Can`t get face.
        cv2.imshow('img', img)
        key = cv2.waitKey(30) &
        if key == 27:
            sys.exit(0)

    for i, d in enumerate(dets)
        x1 = d.top() if d.top()
        y1 = d.bottom() if d.bo
        x2 = d.left() if d.left
        y2 = d.right() if d.rig
        face = img[x1:y1,x2:y2]
        # 调整图片的尺寸
        face = cv2.resize(face,
        print('Is this my face?

        cv2.rectangle(img, (x2,
        cv2.imshow('image',img)
        key = cv2.waitKey(30) &
        if key == 27:
            sys.exit(0)

sess.close()
```

Mysql修改表编码格式 >