Report for Face Recognition with app lock

Python Programming (INT-213)

Name	: Kunal Tyagi
Reg no.	: 12013504
Name	: Sumit Kumar
Reg no.	:12012381
Program	: CSE B.tech
Semester	: Third
School	: School of Computer Science
Name of University	: Lovely Professional University
Date of submission	: 20-Nov-2021

Face Recognition with app lock

Abstract :-

The face is one of the easiest ways to distinguish the individual identity of each other. Face recognition is a personal identification system that uses personal characteristics of a person to identify the person's identity. Human face recognition procedure basically consists of two phases, namely face detection, where this process takes place very rapidly in humans, except under conditions where the object is located at a short distance away, the next is the introduction, which recognize a face as individuals. Stage is then replicated and developed as a model for facial image recognition (face recognition) is one of the much-studied biometrics technology and developed by experts. There are two kinds of methods that are currently popular in developed face recognition pattern namely, Eigenface method and Fisherface method.

Facial image recognition Eigenface method is based on the reduction of facedimensional space using Principal Component Analysis (PCA) for facial features. The main purpose of the use of PCA on face recognition using Eigen faces was formed (face space) by finding the eigenvector corresponding to the largest eigenvalue of the face image. The area of this project face detection system with face recognition is Image processing. The software requirements for this project is PyCharm.

Table of Contents

Introduction :-

1.1 *Context* -

This project has been done as a part of my course for the CSE at Lovely Professionally University. Supervised by Ankita Wadhawan, I have to fulfill the requirements in order to succeed the module.

1.2 *Motivations* -

Because we are particularly interested in anything related to Machine Learning, the group project provided us with an excellent opportunity to study and reinforce our enthusiasm for this topic. In terms of application possibilities, the capacity to create estimates, forecasts, and offer machines the ability to learn on their own is both powerful and endless. Machine Learning may be used in finance, medicine, and practically any other field. As a result, I choose to focus my effort on Machine Learning.

1.3 *Idea* -

We intended to make project as didactic as possible as a first experience, so we approached each step of the machine learning process and tried to comprehend them thoroughly. We chose Face recognition with app lock as our strategy, which is known as a "toy problem," which are challenges that are not of immediate scientific relevance but are good to show and practice. The goal was to

Team Members :-

Team Leader :

Sumit Kumar:-

1. Contributions:-
2. Coding (joined)
3. machine learning(joined)
4. Reports(joined)
5. Datasets

Kunal Tyagi:-

1. Contributions:-
2. Coding (joined)
3. machine learning(joined)
4. Reports(joined)
5. Datasets

Libraries :-

Cv2:-

Cv2 is the module import name for opencv-python, "Unofficial pre-built CPU-only OpenCV packages for Python". The traditional OpenCV has many complicated steps involving building the module from scratch, which is unnecessary. I would recommend remaining with the opencv-python library.

Numpy:-

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.It is the fundamental package for scientific computing with Python.
As the whole project is based on whole complex stats ,we will use this fast calculations and provide results.

PIL (Pillow):-

Python Imaging Library (expansion of PIL) is the de facto image processing package for Python language. It incorporates lightweight image processing tools that aids in editing, creating and saving images. Support for Python Imaging Library got discontinued in 2011, but a project named pillow forked the original PIL project and added Python3.x support to it. Pillow was announced as a replacement for PIL for future usage. Pillow supports a large number of image file formats including BMP, PNG, JPEG, and TIFF.

The library encourages adding support for newer formats in the library by creating new file decoders.

OS :-

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. Python OS module provides the facility to establish the interaction between the user and the operating system. It offers many useful OS functions that are used to perform OS-based tasks and get related information about operating system.

Coding and ScreenShot of Project

Step 1 - Create Training Data

```python
import cv2

cam = cv2.VideoCapture(0, cv2.CAP_DSHOW) #create a video capture object which is helpful to capture videos through webcam
cam.set(3, 640) # set video FrameWidth
cam.set(4, 480) # set video FrameHeight


detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
#Haar Cascade classifier is an effective object detection approach

face_id = input("Enter a Numeric user ID  here:  ")
#Use integer ID for every new face (0,1,2,3,4,5,6,7,8,9.....)

print("Taking samples, look at camera ..... ")
count = 0 # Initializing sampling face count

while True:

    ret, img = cam.read() #read the frames using the above created object
    converted_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY) #The function converts an input image from one color space to another
    faces = detector.detectMultiScale(converted_image, 1.3, 5)

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

        cv2.rectangle(img, (x,y), (x+w,y+h), (255,0,0), 2) #used to draw a rectangle on any image
        count += 1


        cv2.imwrite("samples/face." + str(face_id) + '.' + str(count) + ".jpg", converted_image[y:y+h,x:x+w])
        # To capture & Save images into the datasets folder
```

```
        cv2.imshow('image', img) #Used to display an image in a window

    k = cv2.waitKey(100) & 0xff_# Waits for a pressed key
    if k == 27:_# Press 'ESC' to stop
        break
    elif count >= 100:_# Take 50 sample (More sample --> More accuracy)
        break

print("Samples taken now closing the program....")
cam.release()
cv2.destroyAllWindows()
```

Output of training Data :-

```
C:\Users\hp\anaconda3\python.exe "D:/Face-Recognition-main/Sample generator.py"
Enter a Numeric user ID  here:  1
Taking samples, look at camera .......
Samples taken now closing the program....

Process finished with exit code 0
```

Step 2 - Train Model

```python
import cv2
import numpy as np
from PIL import Image #pillow package
import os

path = 'samples' # Path for samples already taken

recognizer = cv2.face.LBPHFaceRecognizer_create() # Local Binary Patterns Histograms
detector = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
#Haar Cascade classifier is an effective object detection approach


def Images_And_Labels(path): # function to fetch the images and labels

    imagePaths = [os.path.join(path,f) for f in os.listdir(path)]
    faceSamples=[]
    ids = []

    for imagePath in imagePaths: # to iterate particular image path

        gray_img = Image.open(imagePath).convert('L') # convert it to grayscale
        img_arr = np.array(gray_img,'uint8') #creating an array

        id = int(os.path.split(imagePath)[-1].split(".")[1])
        faces = detector.detectMultiScale(img_arr)

        for (x,y,w,h) in faces:
            faceSamples.append(img_arr[y:y+h,x:x+w])
            ids.append(id)

    return faceSamples,ids

print ("Training faces. It will take a few seconds. Wait ...")

faces,ids = Images_And_Labels(path)
recognizer.train(faces, np.array(ids))

recognizer.write('trainer/trainer.yml')  # Save the trained model as trainer.yml

print("Model trained, Now we can recognize your face.")
```

Output of the Model trained :-

```
C:\Users\hp\anaconda3\python.exe "D:/Face-Recognition-main/Model Trainer.py"
Training faces. It will take a few seconds. Wait ...
Model trained, Now we can recognize your face.

Process finished with exit code 0
```

Step 3 - Run Our Facial Recognition

```
import cv2

recognizer = cv2.face.LBPHFaceRecognizer_create() # Local Binary Patterns Histograms
recognizer.read('trainer/trainer.yml')   #load trained model
cascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath); #initializing haar cascade for object detection approach

font = cv2.FONT_HERSHEY_SIMPLEX #denotes the font type


id = 2 #number of persons you want to Recognize


names = ['', 'Kunal']  #names, leave first empty bcz counter starts from 0


cam = cv2.VideoCapture(0, cv2.CAP_DSHOW) #cv2.CAP_DSHOW to remove warning
cam.set(3, 640) # set video FrameWidht
cam.set(4, 480) # set video FrameHeight

# Define min window size to be recognized as a face
minW = 0.1*cam.get(3)
minH = 0.1*cam.get(4)

# flag = True

while True:

    ret, img =cam.read() #read the frames using the above created object
```

```
    converted_image = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)  #The function converts an input image from one color space to another

    faces = faceCascade.detectMultiScale(
        converted_image,
        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) #used to draw a rectangle on any image

        id, accuracy = recognizer.predict(converted_image[y:y+h,x:x+w]) #to predict on every single image

        # Check if accuracy is less them 100 ==> "0" is perfect match
        if (accuracy < 100):
            id = names[id]
            accuracy = "  {0}%".format(round(100 - accuracy))

        else:
            id = "unknown"
            accuracy = "  {0}%".format(round(100 - accuracy))

        cv2.putText(img, str(id), (x+5,y-5), font, 1, (255,255,255), 2)
        cv2.putText(img, str(accuracy), (x+5,y+h-5), font, 1, (255,255,0), 1)

    cv2.imshow('camera',img)
```

```
    k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video
    if k == 27:
        break

# Do a bit of cleanup
print("Thanks for using this program, have a good day.")
cam.release()
cv2.destroyAllWindows()
```

Output of Facial Recognition :-

```
C:\Users\hp\anaconda3\python.exe "D:/Face-Recognition-main/Face recognition.py"
Thanks for using this program, have a good day.

Process finished with exit code 0
```