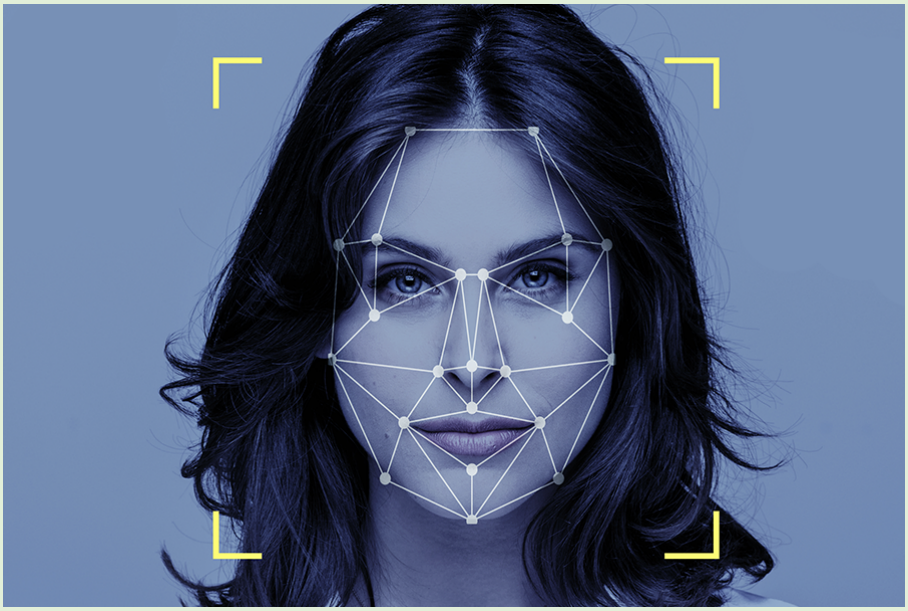
Project Report

Facial Recognition for Class Attendance



Amit Asok

201EC207

Sriram Gangadharan

201EC262

Table of Contents:-

APPROVAL FROM GUIDE

Mini project in Image Processing Inbox ×

SRIRAM PONNAMBALAM G .
Good Evening Sir, We intend on doing a project on (class attendance) using facial recognition. We will create a model database

Dr. Shyam Lal
You can do but creat data base of 100 people minimum

SRIRAM PONNAMBALAM G . <srirampg.201ec262@nitk.edu.in>
to Shyam, AMIT

Good Evening,
Thank You sir for your response. We will create our data base according to your instructions.

Thank You
Sriram Ponnambalam Gangadharan

ABSTRACT

A face analyzer is software that identifies or confirms a person's identity using their face. It works by identifying and measuring facial features in an image. Facial recognition can identify human faces in images or videos, determine if the face in two images belongs to the same person, or search for a face among a large collection of existing images. Biometric security systems use facial recognition to uniquely identify individuals during user onboarding or logins as well as strengthen user authentication activity. Mobile and personal devices also commonly use face analyzer technology for device security.

MOTIVATION

Face recognition technology can help avoid identity fraud in today's environment, when identity theft is common.

According to a 2019 estimate, 3.2 million fraud instances were reported to the FTC (Federal Trade Commission), with 20.33% of cases involving identity theft.

Modern AI-enabled facial recognition technology is highly accurate and can match even the most distinct features of a human face. Businesses and organizations of all sizes can use this technology to significantly reduce the danger of identity theft.

Facial recognition can be used in attendance to minimize proxies. It is a reliable and an efficient system that can operate with minimal to no error.

WORK DONE

We created a model that works with the created dataset. The dataset consists of 50 students. We used the OpenCV library to take the input through the web camera. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, and classify human actions in videos. The Haar Cascade algorithm is used in the detection of faces. It is able to track faces real time.

DATASET


aman.21.46.jpg
aman.21.47.jpg
aman.21.48.jpg
aman.21.49.jpg
aman.21.50.jpg
aman.21.51.jpg
aman.21.52.jpg
aman.21.53.jpg
aman.21.54.jpg
aman.21.55.jpg
aman.21.56.jpg
aman.21.57.jpg
aman.21.58.jpg
aman.21.59.jpg
aman.21.60.jpg
aman.21.61.jpg
aman.21.62.jpg
aman.21.63.jpg
aman.21.64.jpg
aman.21.65.jpg
aman.21.66.jpg
aman.21.67.jpg
aman.21.68.jpg
aman.21.69.jpg
aman.21.70.jpg
aman.21.71.jpg
aman.21.72.jpg
amit.7.1.jpg
amit.7.2.jpg
amit.7.3.jpg
amit.7.4.jpg
amit.7.5.jpg
amit.7.6.jpg
amit.7.7.jpg
amit.7.8.jpg
amit.7.9.jpg
amit.7.10.jpg
amit.7.11.jpg
amit.7.12.jpg
amit.7.13.jpg
amit.7.14.jpg
amit.7.15.jpg
amit.7.16.jpg
amit.7.17.jpg
amit.7.18.jpg
amit.7.19.jpg
amit.7.20.jpg
amit.7.21.jpg
amit.7.22.jpg
amit.7.23.jpg
amit.7.24.jpg
amit.7.25.jpg
amit.7.26.jpg
amit.7.27.jpg
amit.7.28.jpg
amit.7.29.jpg
amit.7.30.jpg
amit.7.31.jpg
amit.7.32.jpg
amit.7.33.jpg
amit.7.34.jpg
amit.7.35.jpg
amit.7.36.jpg
amit.7.37.jpg
amit.7.38.jpg
amit.7.39.jpg
amit.7.40.jpg
amit.7.41.jpg
amit.7.42.jpg
amit.7.43.jpg
amit.7.44.jpg
amit.7.45.jpg
amit.7.46.jpg
amit.7.47.jpg
amit.7.48.jpg
amit.7.49.jpg
amit.7.50.jpg
amit.7.51.jpg
amit.7.52.jpg
amit.7.53.jpg
amit.7.54.jpg
amit.7.55.jpg
amit.7.56.jpg
amit.7.57.jpg
amit.7.58.jpg
amit.7.59.jpg
amit.7.60.jpg
amit.7.61.jpg
amit.7.62.jpg
amit.7.63.jpg
amit.7.64.jpg
amit.7.65.jpg
amit.7.66.jpg
amit.7.67.jpg
amit.7.68.jpg
amit.7.69.jpg
amit.7.70.jpg
amit.7.71.jpg
amit.7.72.jpg
amit.7.73.jpg
amit.7.74.jpg
amit.7.75.jpg
amit.7.76.jpg
amit.7.77.jpg
amit.7.78.jpg
amit.7.79.jpg
amit.7.80.jpg
amit.7.81.jpg
amit.7.82.jpg
amit.7.83.jpg
amit.7.84.jpg
amit.7.85.jpg
amit.7.86.jpg
amit.7.87.jpg
amit.7.88.jpg
amit.7.89.jpg
amit.7.90.jpg
amit.7.91.jpg
amit.7.92.jpg
amit.7.93.jpg
amit.7.94.jpg
amit.7.95.jpg
amit.7.96.jpg
amit.7.97.jpg
amit.7.98.jpg
amit.7.99.jpg
amit.7.100.jpg
amit.7.101.jpg
amit.7.102.jpg
anush.11.1.jpg
anush.11.2.jpg
anush.11.3.jpg
anush.11.4.jpg
anush.11.5.jpg
anush.11.6.jpg
anush.11.7.jpg

CODE

1. main.py

```
import os  # access ing the os functions
import check_camera
import Capture_Image
import Train_Image
import Recognize

# creating the title bar function24

def title_bar():
    os.system('cls')  # for windows

    # title of the program

    print("\t*********************************************")
    print("\t***** Face Recognition Attendance System *****")

    print("\t***************2*****************************")


# creating the user main menu function

def mainMenu():
    title_bar()
    print()
    print(10 * "*", "WELCOME MENU", 10 * "*")
    print("[1] Check Camera")
    print("[2] Capture Faces")
    print("[3] Train Images")
    print("[4] Recognize & Attendance")
    print("[5] Quit")
```

```python
while True:
    try:
        choice = int(input("Enter Choice: "))

        if choice == 1:
            checkCamera()
            break
        elif choice == 2:
            CaptureFaces()
            break
        elif choice == 3:
            Trainimages()
            break
        elif choice == 4:
            RecognizeFaces()
            break
            mainMenu()
        elif choice == 5:
            print("Thank You")
            break
        else:
            print("Invalid Choice. Enter 1-4")
            mainMenu()
    except ValueError:
        print("Invalid Choice. Enter 1-4\n Try Again")
exit
```

```python
# calling the camera test function from check camera.py file

def checkCamera():
    check_camera.camer()
    key = input("Enter any key to return main menu")
    mainMenu()


# ------------------------------------------------------------
# calling the take image function form capture image.py file

def CaptureFaces():
    Capture_Image.takeImages()
    key = input("Enter any key to return main menu")
    mainMenu()


# ---------------------------------------------------------------
# calling the train images from train_images.py file

def Trainimages():
    Train_Image.TrainImages()
    key = input("Enter any key to return main menu")
    mainMenu()


# ------------------------------------------------------------------
# calling the recognize_attendance from recognize.py file
```

```python
# ---------------------------------------------------------------------
# calling the recognize_attendance from recognize.py file

def RecognizeFaces():
    Recognize.recognize_attendence()
    key = input("Enter any key to return main menu")
    mainMenu()


# ---------------main driver ------------------
mainMenu()
```

2. check_camera.py

```python
def camer():
    import cv2

    # Load the cascade
    face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    face_cascade.load('haarcascade_frontalface_default.xml')

    # To capture video from webcam.
    cap = cv2.VideoCapture(0)

    while True:
        # Read the frame
        _, img = cap.read()

        # Convert to grayscale
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

        # Detect the faces
        faces = face_cascade.detectMultiScale(gray, 1.3, 5, minSize=(30, 30), flags = cv2.CASCADE_SCALE_IMAGE)

        # Draw the rectangle around each face
        for (x, y, w, h) in faces:
            cv2.rectangle(img, (x, y), (x + w, y + h), (10,159,255), 2)


        # Display
        cv2.imshow('Webcam Check', img)
```

```python
        # Stop if escape key is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # Release the VideoCapture object
    cap.release()
    cv2.destroyAllWindows()
```

3. capture.py

```python
import csv
import cv2
import os

dirname = os.path.dirname(__file__)
filename = os.path.join(dirname, 'StudentsDetails\StudentDetails.csv')

# counting the numbers

def is_number(s):
    try:
        float(s)
        return True
    except ValueError:
        pass

    try:
        import unicodedata
        unicodedata.numeric(s)
        return True
    except (TypeError, ValueError):
        pass

    return False
```

```python
def takeImages():

    Id = input("Enter Your Id: ")
    name = input("Enter Your Name: ")

    if(is_number(Id) and name.isalpha()):
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector = cv2.CascadeClassifier(cv2.data.haarcascades + harcascadePath)
        sampleNum = 0

        while(True):
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, 1.3, 5, minSize=(30,30),flags = cv2.CASCADE_SCALE_IMAGE)
            for(x,y,w,h) in faces:
                cv2.rectangle(img, (x, y), (x+w, y+h), (10, 159, 255), 2)
                #incrementing sample number
                sampleNum = sampleNum+1
                #saving the captured face in the dataset folder TrainingImage
                cv2.imwrite("TrainingImage" + os.sep + name + "."+Id + '.' +
                            str(sampleNum) + ".jpg", gray[y:y+h, x:x+w])
                #display the frame
                cv2.imshow('frame', img)
            #wait for 100 miliseconds
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
            # break if the sample number is more than 100
            elif sampleNum > 100:
                break
```

```
            sampleNum = sampleNum+1
            #saving the captured face in the dataset folder TrainingImage
            cv2.imwrite("TrainingImage" + os.sep + name + "."+Id + '.' +
                        str(sampleNum) + ".jpg", gray[y:y+h, x:x+w])
            #display the frame
            cv2.imshow('frame', img)
        #wait for 100 miliseconds
        if cv2.waitKey(100) & 0xFF == ord('q'):
            break
        # break if the sample number is more than 100
        elif sampleNum > 100:
            break
    cam.release()
    cv2.destroyAllWindows()
    res = "Images Saved for ID : " + Id + " Name : " + name
    row = [Id, name]
    csvFile = open(filename, 'a+')
    writer = csv.writer(csvFile)
    writer.writerow(row)
    csvFile.close()
else:
    if(is_number(Id)):
        print("Enter Alphabetical Name")
    if(name.isalpha()):
        print("Enter Numeric ID")
```

4. recognize.py

```
import datetime
import os
import time

import cv2
import pandas as pd

dirname = os.path.dirname(__file__)
filename = os.path.join(dirname, 'StudentsDetails\StudentDetails.csv')

#-------------------------
def recognize_attendence():
    recognizer = cv2.face.LBPHFaceRecognizer_create()  # cv2.createLBPHFaceRecognizer()
    recognizer.read("TrainingImageLabel"+os.sep+"Trainner.yml")
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath)
    df = pd.read_csv(filename)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names = ['Id', 'Name', 'Date', 'Time']
    attendance = pd.DataFrame(columns=col_names)

    # Initialize and start realtime video capture
    cam = cv2.VideoCapture(0, cv2.CAP_DSHOW)
    cam.set(3, 640)  # set video width
    cam.set(4, 480)  # set video height
    # Define min window size to be recognized as a face
    minW = 0.1 * cam.get(3)
    minH = 0.1 * cam.get(4)
```

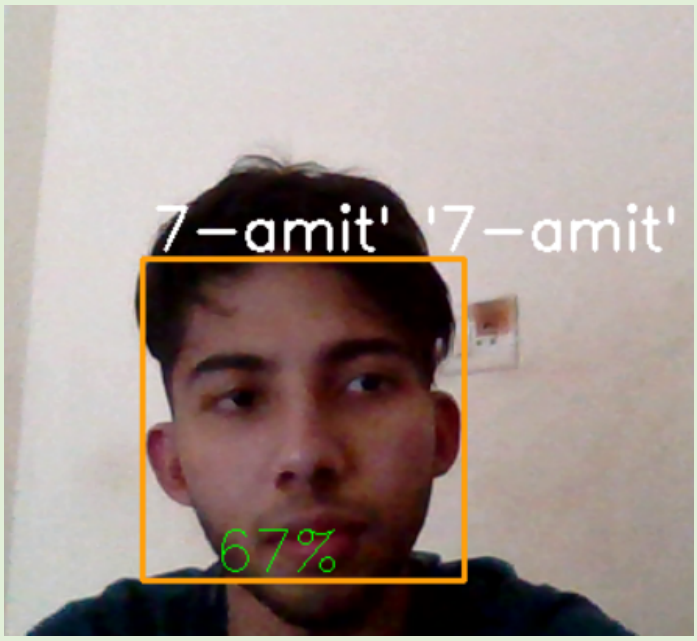
```python
while True:
    ret, im = cam.read()
    gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    faces = faceCascade.detectMultiScale(gray, 1.2, 5,minSize = (int(minW), int(minH)),flags = cv2.CASCADE_SCALE_IMAGE)
    for(x, y, w, h) in faces:
        cv2.rectangle(im, (x, y), (x+w, y+h), (10, 159, 255), 2)
        Id, conf = recognizer.predict(gray[y:y+h, x:x+w])

        if conf < 100:

            aa = df.loc[df['Id'] == Id]['Name'].values
            confstr = "  {0}%".format(round(100 - conf))
            tt = str(Id)+"-"+aa


        else:
            Id = '  Unknown  '
            tt = str(Id)
            confstr = "  {0}%".format(round(100 - conf))

        if (100-conf) > 55\
                :
            ts = time.time()
            date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
            timeStamp = datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
            aa = str(aa)[2:-2]
            attendance.loc[len(attendance)] = [Id, tt, date, timeStamp]
```

5. train.py

```python
import os
import time
import cv2
import numpy as np
from PIL import Image
from threading import Thread



# -------------- image labesl ------------------------

def getImagesAndLabels(path):
    # get the path of all the files in the folder
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
    # print(imagePaths)

    # create empth face list
    faces = []
    # create empty ID list
    Ids = []
    # now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
        # loading the image and converting it to gray scale
        pilImage = Image.open(imagePath).convert('L')
        # Now we are converting the PIL image into numpy array
        imageNp = np.array(pilImage, 'uint8')
        # getting the Id from the image
        Id = int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
```

```python
    for imagePath in imagePaths:
        # loading the image and converting it to gray scale
        pilImage = Image.open(imagePath).convert('L')
        # Now we are converting the PIL image into numpy array
        imageNp = np.array(pilImage, 'uint8')
        # getting the Id from the image
        Id = int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(Id)
    return faces, Ids
# ----------- train images function ---------------
def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, Id = getImagesAndLabels("TrainingImage")
    Thread(target = recognizer.train(faces, np.array(Id))).start()
    # Below line is optional for a visual counter effect
    Thread(target = counter_img("TrainingImage")).start()
    recognizer.save("TrainingImageLabel"+os.sep+"Trainner.yml")
    print("All Images")

# Optional, adds a counter for images trained (You can remove it)
def counter_img(path):
    imgcounter = 1
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
    for imagePath in imagePaths:
        print(str(imgcounter) + " Images Trained", end="\r")
        time.sleep(0.008)
```

OUTPUTS



C4

	A	B	C	D	E	F	G	H	I	J	K
1	Id	Name	Date	Time							
2	7	['7-amit' '	11/21/2022	10:40:53							
3											
4											
5											
6											
7											
8											
9											
10											

CONCLUSION

We built a data set and related literature to get familiarized with the given data samples. In order to build the facial recognition system, vast amounts of data is needed. For this purpose, we have started creating a data set of 50 entities. Training and testing the model involves Face Detection, Face Alignments, Feature Extraction, Face Recognition. The model is able to identify correctly albeit with a slight chance of error.

The accuracy of the model could be improved by using CNN algorithm which involves deep learning.

REFERENCES

- https://www.analyticsvidhya.com/blog/2021/11/build-face-recognition-attendance-system-using-python/
- https://realpython.com/face-recognition-with-python/