Face Recognition with Digital Image Processing

(Using OpenCV & Python)

A COURSE PROJECT REPORT

By

Saurabh Pandey (RA2011003010207)
Aakash Chaudhary (RA2011003010159)

Under the guidance of

Mr.Saminathan S
Assistant Professor
In partial fulfillment for the Course
of

18CSE353T - Digital Image Processing

in

Computer Science & Engineering

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Chengalpattu District

APRIL 2023

COLLEGE OF ENGINEERING &amp; TECHNOLOGY
SRM INSTITUTE OF SCIENCE &amp; TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this project report "**Face Recognition with Digital Image Processing**" is the bonafide work of "**Aakash Chaudhary(RA2011003010159)** and **Saurabh Pandey(RA2011003010207)**" of III Year/VI Sem B.tech(CSE) who carried out the mini project work under my supervision for the course **18CSE353T-Digital Image Processing** in SRM Institute of Science and Technology during the academic year 2022-2023(Even sem).

SIGNATURE
Mr.Saminathan S
Assistant Professor
Department of CTECH

ABSTRACT

A face recognition system is an application of digital image processing that uses algorithms to detect and identify human faces in images or videos. The system extracts relevant facial features, such as the eyes, nose, and mouth, and uses them to create a unique biometric signature for each individual. This signature is then compared to a database of known faces to identify the person.

Digital image processing techniques, such as filtering, segmentation, and feature extraction, are used to enhance the quality of facial images and to isolate key facial features for recognition. The system is also capable of detecting variations in lighting, pose, and facial expressions to improve the accuracy of identification.

Face recognition systems have a wide range of applications, from security and surveillance to human-computer interaction and personalized marketing. The development of such systems is a rapidly evolving field, with ongoing research focused on improving accuracy, scalability, and robustness in various real-world scenarios

TABLE OF CONTENTS

INTRODUCTION TO FACE RECOGNITION SYSTEM

Face recognition is a technology that enables computers to detect and recognize human faces. This technology is based on digital image processing techniques that extract facial features and patterns to create a unique biometric signature for each individual. Face recognition systems can be used for a wide range of applications, including security and surveillance, access control, human-computer interaction, and personalization.

The concept of face recognition has been around for decades, but recent advances in digital image processing, machine learning, and artificial intelligence have significantly improved its accuracy and reliability. The development of deep learning algorithms, such as convolutional neural networks (CNNs), has enabled computers to learn from large datasets of facial images and to identify patterns and features that are unique to each individual.

Face recognition systems typically consist of two main components: face detection and face recognition. Face detection is the process of locating and isolating faces in images or videos, while face recognition is the process of comparing the detected faces to a database of known faces to identify the person.

One of the key challenges in face recognition is dealing with variations in lighting, pose, and facial expressions. To overcome this challenge, digital image processing techniques are used to enhance the quality of facial images and to normalize them to a standard pose and lighting condition. This enables the system to extract consistent and accurate features for recognition.

In recent years, face recognition systems have become more prevalent in our daily lives. They are used in smartphones, social media platforms, and online banking applications to provide secure and convenient authentication. However, the widespread adoption of face recognition has also raised concerns about privacy and security. The potential for misuse and abuse of this technology has led to calls for greater regulation and oversight to ensure its ethical and responsible use.

In conclusion, face recognition is a powerful technology that has the potential to revolutionize many aspects of our lives. However, its development and deployment must be guided by ethical and legal frameworks to ensure its responsible use and to protect the privacy.

FUNDAMENTAL OF DIGITAL IMAGE PROCESSING

Fundamentals of Digital Image Processing involve techniques used to manipulate images to enhance their quality, extract useful information and identify objects in images. It is a subfield of signal processing that deals with images or multi-dimensional signals. Digital Image Processing involves a wide range of algorithms and techniques used to process images to improve their quality, manipulate them to achieve specific effects, and extract useful information for various applications.

One of the fundamental concepts of digital image processing is image representation, where an image is represented as a two-dimensional array of pixel values. Each pixel corresponds to a single color or grayscale value and forms the basic unit of an image. Another important concept is the use of mathematical operations, such as convolution and Fourier transform, to manipulate image data.

Filters are another essential concept of digital image processing. They are used to remove unwanted noise, blur, sharpen or enhance features in images. Filtering techniques, such as median filtering and Gaussian filtering, are widely used to enhance the quality of images and remove noise from them.

Segmentation is another important technique used in digital image processing to identify objects or regions of interest in an image. Segmentation techniques, such as thresholding, edge detection, and region growing, are widely used to segment images for object recognition and tracking.

Feature extraction is another key concept in digital image processing. It involves identifying relevant features in an image, such as edges, corners, and texture, to create a compact representation of an image. These features are then used in tasks such as object recognition and classification.

FACIAL FEATURE EXTRACTION

Facial feature extraction is a critical step in face recognition systems that involves the identification and extraction of relevant features from facial images. These features serve as distinctive characteristics that can be used to differentiate one face from another. The extracted features are used to create a unique biometric template that can be compared to a database of known faces to identify an individual.

Facial feature extraction typically involves several steps, including pre-processing, detection, normalization, and encoding. Pre-processing techniques, such as filtering and noise reduction, are used to enhance the quality of facial images and remove any artifacts that may interfere with feature detection. Detection algorithms are then used to locate key facial features, such as the eyes, nose, and mouth, and to identify the facial landmarks that define their positions.

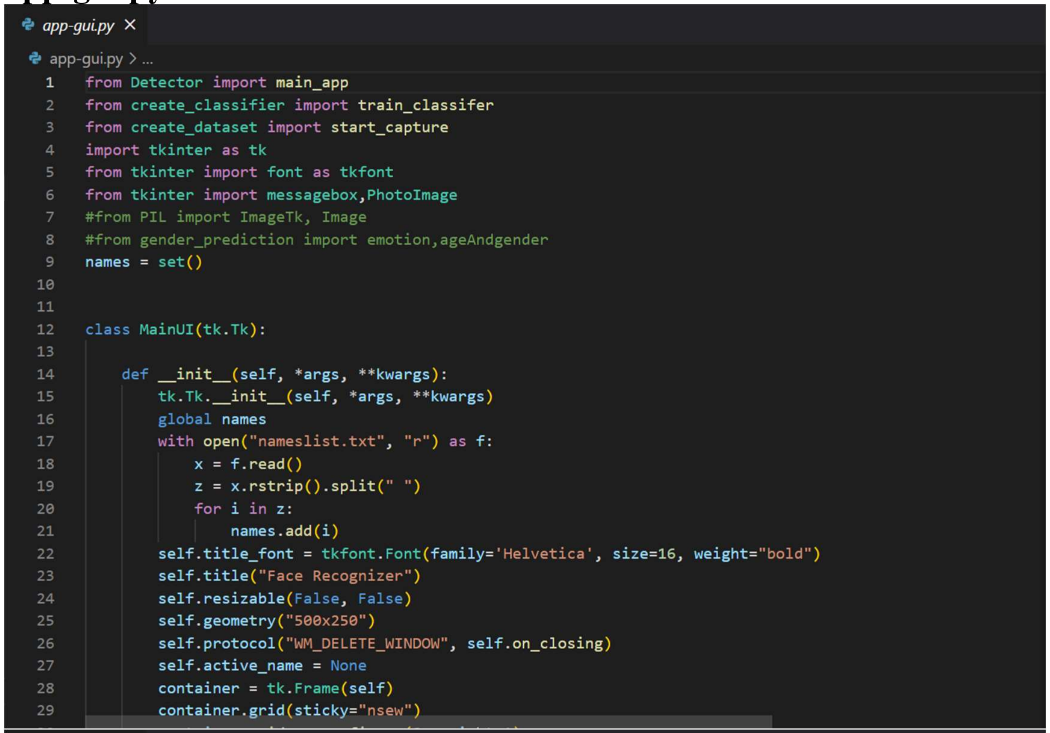
Normalization is the process of standardizing the size and orientation of facial images, so that they can be accurately compared to other images. This involves aligning the detected features with a reference coordinate system and scaling the image to a standard size. Encoding is the final step in facial feature extraction and involves transforming the detected features into a set of numerical values that can be used for recognition.

There are various techniques and algorithms used for facial feature extraction, including Principal Component Analysis (PCA), Local Binary Patterns (LBP), Histograms of Oriented Gradients (HOG), and Convolutional Neural Networks (CNNs). Each of these methods has its own strengths and weaknesses, and the choice of algorithm depends on factors such as the complexity of the application, the size of the dataset, and the computational resources available.

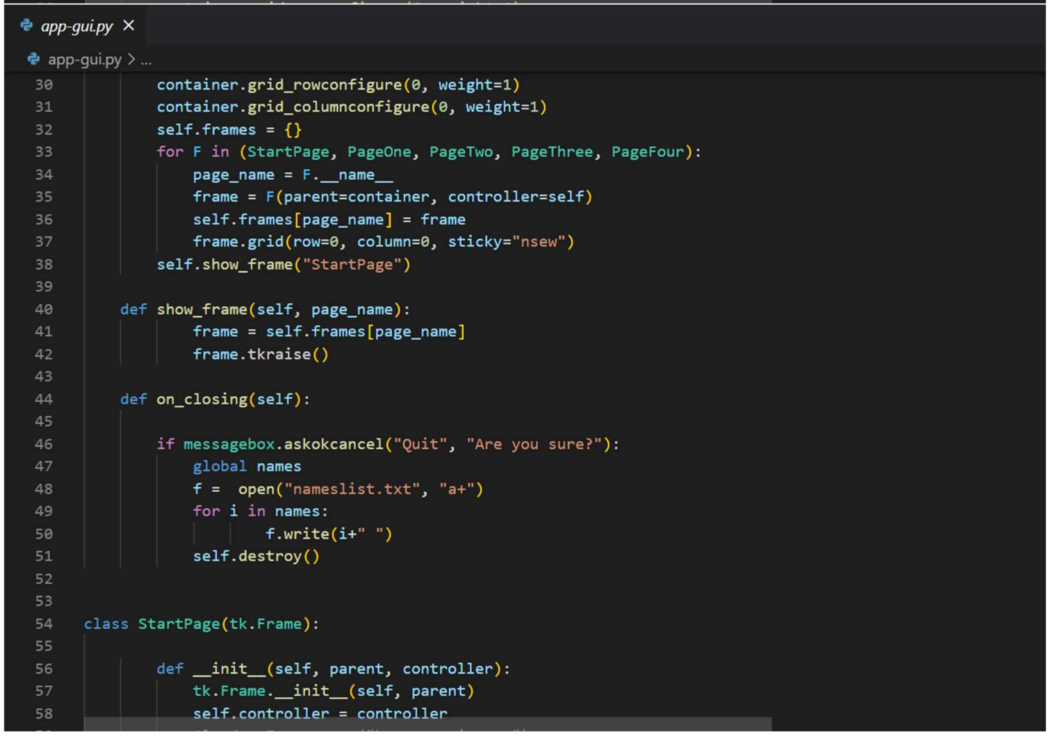
Facial feature extraction is a challenging task due to variations in lighting, pose, expression, and occlusion. Research in this field is ongoing, with the development of new algorithms and techniques aimed at improving the accuracy and robustness of face recognition systems in various real-world scenarios.

PYTHON CODE

app-gui.py

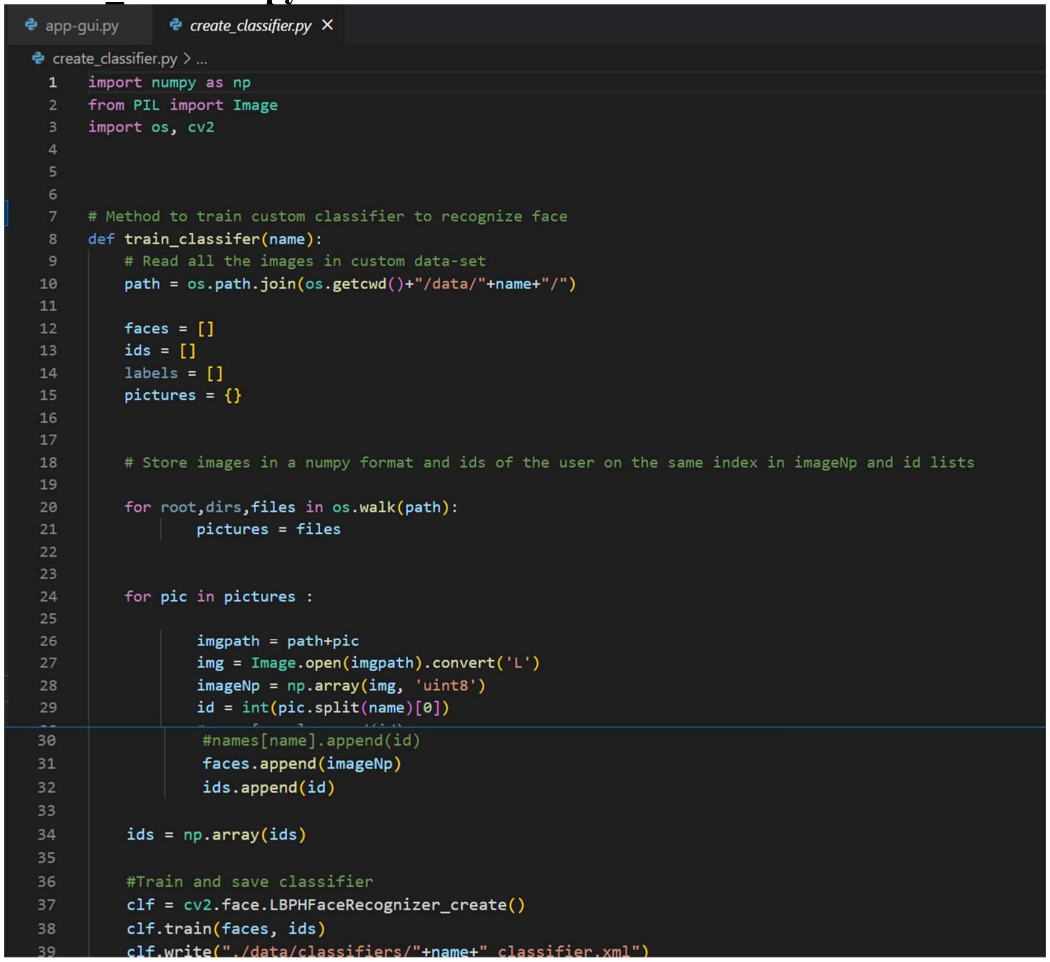


```python
from Detector import main_app
from create_classifier import train_classifer
from create_dataset import start_capture
import tkinter as tk
from tkinter import font as tkfont
from tkinter import messagebox,PhotoImage
#from PIL import ImageTk, Image
#from gender_prediction import emotion,ageAndgender
names = set()


class MainUI(tk.Tk):

    def __init__(self, *args, **kwargs):
        tk.Tk.__init__(self, *args, **kwargs)
        global names
        with open("nameslist.txt", "r") as f:
            x = f.read()
            z = x.rstrip().split(" ")
            for i in z:
                names.add(i)
        self.title_font = tkfont.Font(family='Helvetica', size=16, weight="bold")
        self.title("Face Recognizer")
        self.resizable(False, False)
        self.geometry("500x250")
        self.protocol("WM_DELETE_WINDOW", self.on_closing)
        self.active_name = None
        container = tk.Frame(self)
        container.grid(sticky="nsew")
```



```python
        container.grid_rowconfigure(0, weight=1)
        container.grid_columnconfigure(0, weight=1)
        self.frames = {}
        for F in (StartPage, PageOne, PageTwo, PageThree, PageFour):
            page_name = F.__name__
            frame = F(parent=container, controller=self)
            self.frames[page_name] = frame
            frame.grid(row=0, column=0, sticky="nsew")
        self.show_frame("StartPage")

    def show_frame(self, page_name):
            frame = self.frames[page_name]
            frame.tkraise()

    def on_closing(self):

        if messagebox.askokcancel("Quit", "Are you sure?"):
            global names
            f =  open("nameslist.txt", "a+")
            for i in names:
                    f.write(i+" ")
            self.destroy()


class StartPage(tk.Frame):

        def __init__(self, parent, controller):
            tk.Frame.__init__(self, parent)
            self.controller = controller
```

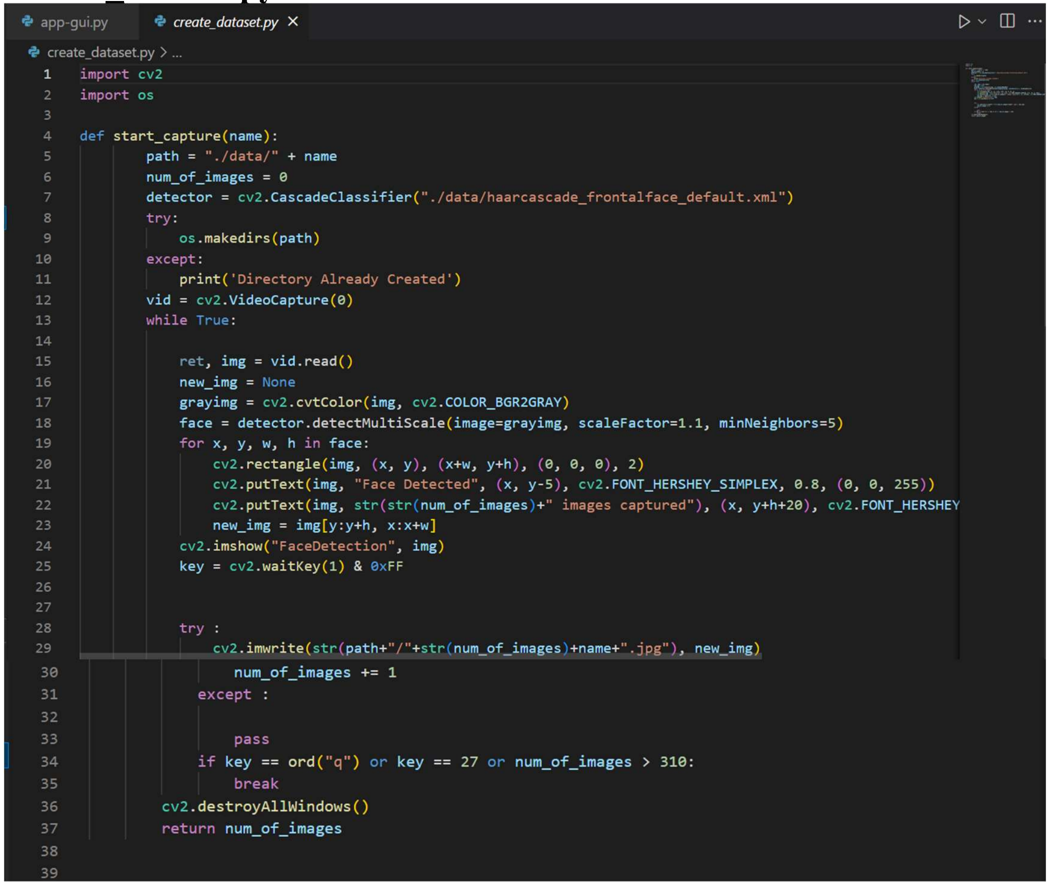
Create_classifier.py


app-gui.py   create_classifier.py

create_classifier.py > ...

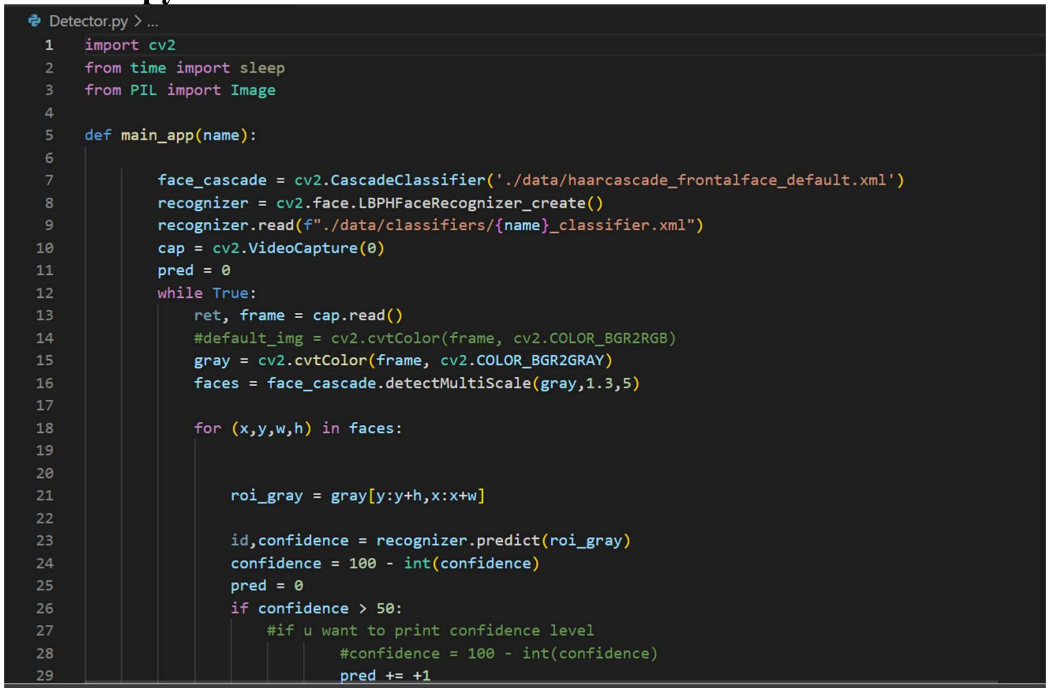
```python
import numpy as np
from PIL import Image
import os, cv2



# Method to train custom classifier to recognize face
def train_classifer(name):
    # Read all the images in custom data-set
    path = os.path.join(os.getcwd()+"/data/"+name+"/")

    faces = []
    ids = []
    labels = []
    pictures = {}


    # Store images in a numpy format and ids of the user on the same index in imageNp and id lists

    for root,dirs,files in os.walk(path):
            pictures = files


    for pic in pictures :

            imgpath = path+pic
            img = Image.open(imgpath).convert('L')
            imageNp = np.array(img, 'uint8')
            id = int(pic.split(name)[0])
            #names[name].append(id)
            faces.append(imageNp)
            ids.append(id)

    ids = np.array(ids)

    #Train and save classifier
    clf = cv2.face.LBPHFaceRecognizer_create()
    clf.train(faces, ids)
    clf.write("./data/classifiers/"+name+"_classifier.xml")
```

Create_dataset.py

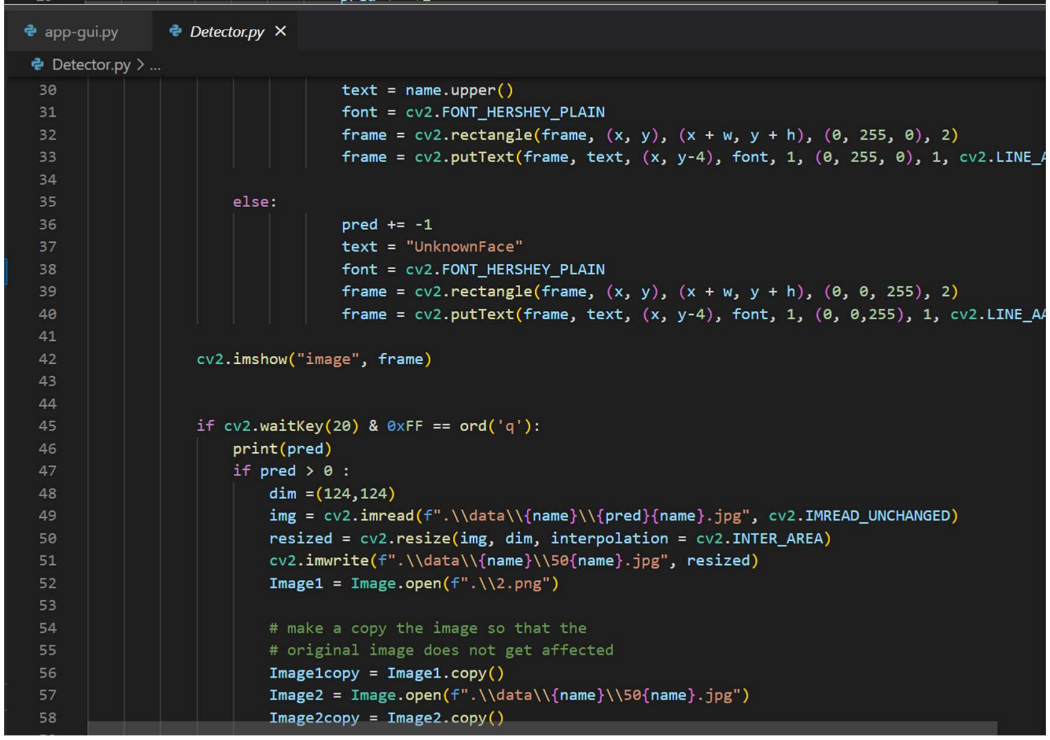


```python
import cv2
import os

def start_capture(name):
        path = "./data/" + name
        num_of_images = 0
        detector = cv2.CascadeClassifier("./data/haarcascade_frontalface_default.xml")
        try:
            os.makedirs(path)
        except:
            print('Directory Already Created')
        vid = cv2.VideoCapture(0)
        while True:

            ret, img = vid.read()
            new_img = None
            grayimg = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            face = detector.detectMultiScale(image=grayimg, scaleFactor=1.1, minNeighbors=5)
            for x, y, w, h in face:
                cv2.rectangle(img, (x, y), (x+w, y+h), (0, 0, 0), 2)
                cv2.putText(img, "Face Detected", (x, y-5), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0, 0, 255))
                cv2.putText(img, str(str(num_of_images)+" images captured"), (x, y+h+20), cv2.FONT_HERSHEY
                new_img = img[y:y+h, x:x+w]
            cv2.imshow("FaceDetection", img)
            key = cv2.waitKey(1) & 0xFF


            try :
                cv2.imwrite(str(path+"/"+str(num_of_images)+name+".jpg"), new_img)
                num_of_images += 1
            except :

                pass
            if key == ord("q") or key == 27 or num_of_images > 310:
                break
        cv2.destroyAllWindows()
        return num_of_images
```

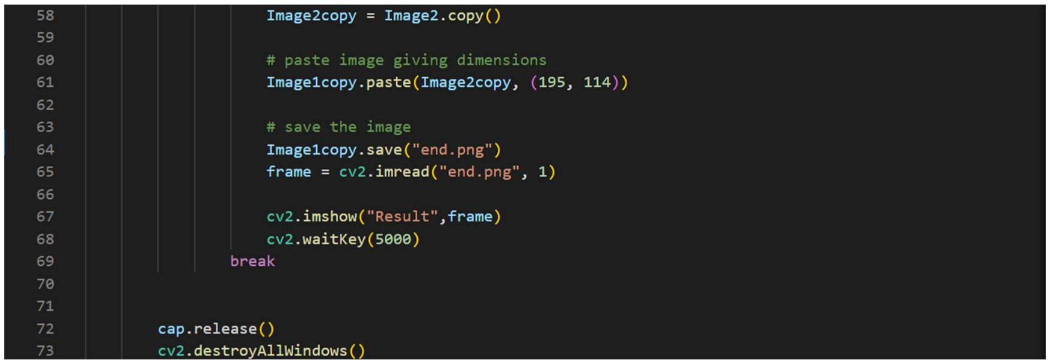
Detector.py



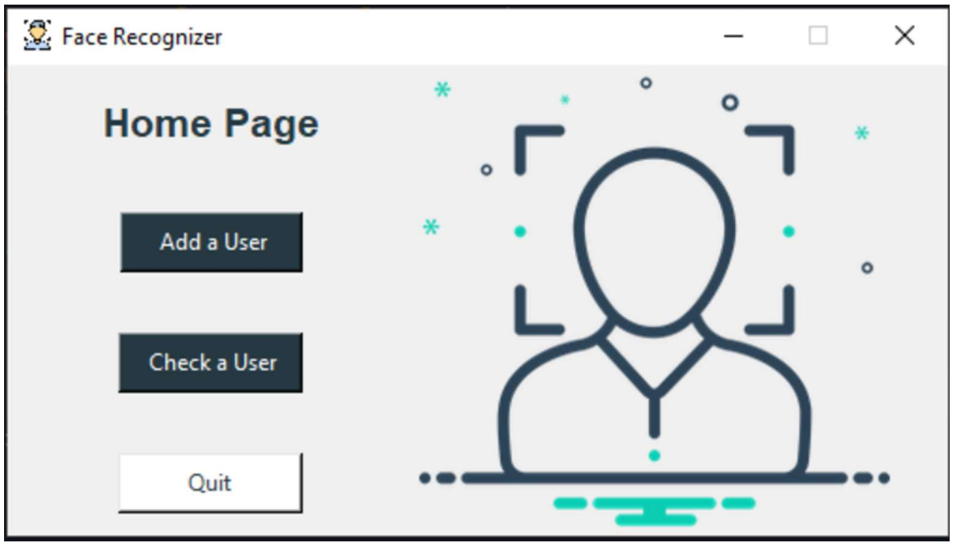
```python
import cv2
from time import sleep
from PIL import Image

def main_app(name):

        face_cascade = cv2.CascadeClassifier('./data/haarcascade_frontalface_default.xml')
        recognizer = cv2.face.LBPHFaceRecognizer_create()
        recognizer.read(f"./data/classifiers/{name}_classifier.xml")
        cap = cv2.VideoCapture(0)
        pred = 0
        while True:
            ret, frame = cap.read()
            #default_img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            faces = face_cascade.detectMultiScale(gray,1.3,5)

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


                roi_gray = gray[y:y+h,x:x+w]

                id,confidence = recognizer.predict(roi_gray)
                confidence = 100 - int(confidence)
                pred = 0
                if confidence > 50:
                    #if u want to print confidence level
                            #confidence = 100 - int(confidence)
                            pred += +1
```

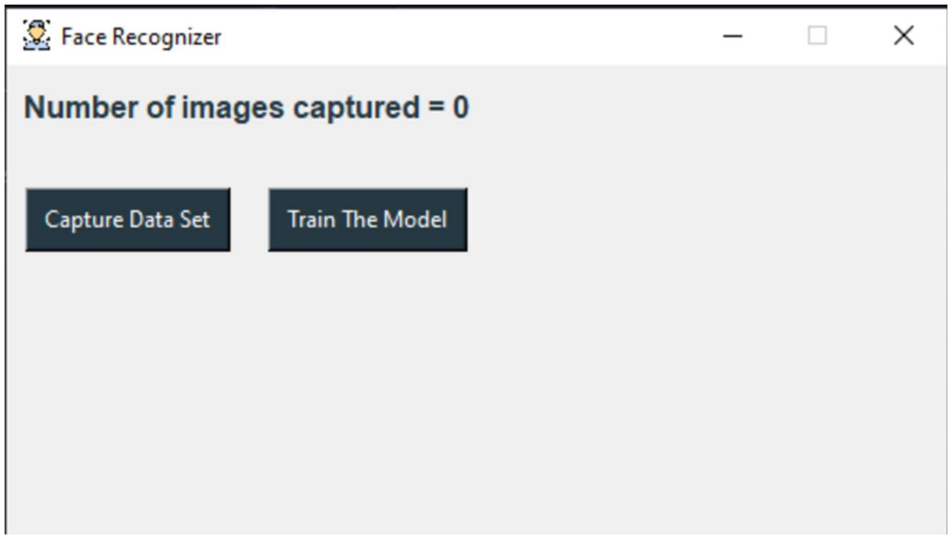


```python
                            text = name.upper()
                            font = cv2.FONT_HERSHEY_PLAIN
                            frame = cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
                            frame = cv2.putText(frame, text, (x, y-4), font, 1, (0, 255, 0), 1, cv2.LINE_A

                else:
                            pred += -1
                            text = "UnknownFace"
                            font = cv2.FONT_HERSHEY_PLAIN
                            frame = cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 2)
                            frame = cv2.putText(frame, text, (x, y-4), font, 1, (0, 0,255), 1, cv2.LINE_AA

            cv2.imshow("image", frame)


            if cv2.waitKey(20) & 0xFF == ord('q'):
                print(pred)
                if pred > 0 :
                    dim =(124,124)
                    img = cv2.imread(f".\\data\\{name}\\{pred}{name}.jpg", cv2.IMREAD_UNCHANGED)
                    resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)
                    cv2.imwrite(f".\\data\\{name}\\50{name}.jpg", resized)
                    Image1 = Image.open(f".\\2.png")

                    # make a copy the image so that the
                    # original image does not get affected
                    Image1copy = Image1.copy()
                    Image2 = Image.open(f".\\data\\{name}\\50{name}.jpg")
                    Image2copy = Image2.copy()
```



```python
                    Image2copy = Image2.copy()

                    # paste image giving dimensions
                    Image1copy.paste(Image2copy, (195, 114))

                    # save the image
                    Image1copy.save("end.png")
                    frame = cv2.imread("end.png", 1)

                    cv2.imshow("Result",frame)
                    cv2.waitKey(5000)
                break


        cap.release()
        cv2.destroyAllWindows()
```


Face Recognizer
Home Page
Add a User
Check a User
Quit


Face Recognizer
Number of images captured = 0
Capture Data Set
Train The Model

CONCLUSION

Facial feature extraction is a crucial step in face recognition systems, and it plays a key role in identifying individuals from facial images. The extraction process involves pre-processing, detection, normalization, and encoding of facial features, which serve as unique biometric signatures for each individual.

The use of digital image processing techniques, such as PCA, LBP, HOG, and CNNs, have greatly improved the accuracy and robustness of facial feature extraction, making it possible to detect features in challenging environments with varying lighting, pose, expression, and occlusion.

Despite the advances made in this field, there are still challenges that need to be addressed, such as the development of algorithms that can accurately detect facial features in real-time and the need for improved privacy and security measures to protect personal biometric data.

Overall, facial feature extraction is an exciting field that has the potential to revolutionize the way we identify and authenticate individuals in various applications, from security and surveillance to personalized marketing and human-computer interaction.

REFERENCES

- Turk, M. and Pentland, A. (1991). Eigenfaces for Recognition. Journal of Cognitive Neuroscience, 3(1), 71-86. DOI: 10.1162/jocn.1991.3.1.71

- Ahonen, T., Hadid, A. and Pietikäinen, M. (2006). Face Recognition with Local Binary Patterns. Proceedings of the European Conference on Computer Vision, Graz, Austria, 469-481. DOI: 10.1007/11744085_36

- Dalal, N. and Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Diego, CA, USA, 886-893. DOI: 10.1109/CVPR.2005.177