Design Overview for Face Recognition Program

Name: Luong Trac Duc Anh
Student ID: 103488117

Summary of Program

As machine learning is becoming an integrated part in our lives, its applications are no longer news to us, from the content that appears when you scroll through your social media applications to the banking domains that prevents frauds and hackers. Another use of machine learning is face recognition, which can be integrated into device-login, human-detection, etc.

My program - Face Recognition works simply by executing the application file, and the app will access the camera, take a screenshot, and make a prediction on who is presented in the taken photo.

My program is written in Python, and utilise libraries such as:
- OpenCV for face detection.
- Keras and Tensorflow to load the model from the dataset.
- playsound and gtts (Google text to speech) to integrate the play sound feature in the app.
- webdriver_manager to import ChromeDriverManager and open Chrome (or any other browser that you prefer like Firefox or Opera) to open Wikipedia.
- Wikipedia to set the preferred language.
- os to manipulate file paths.
- numpy to store the image array.
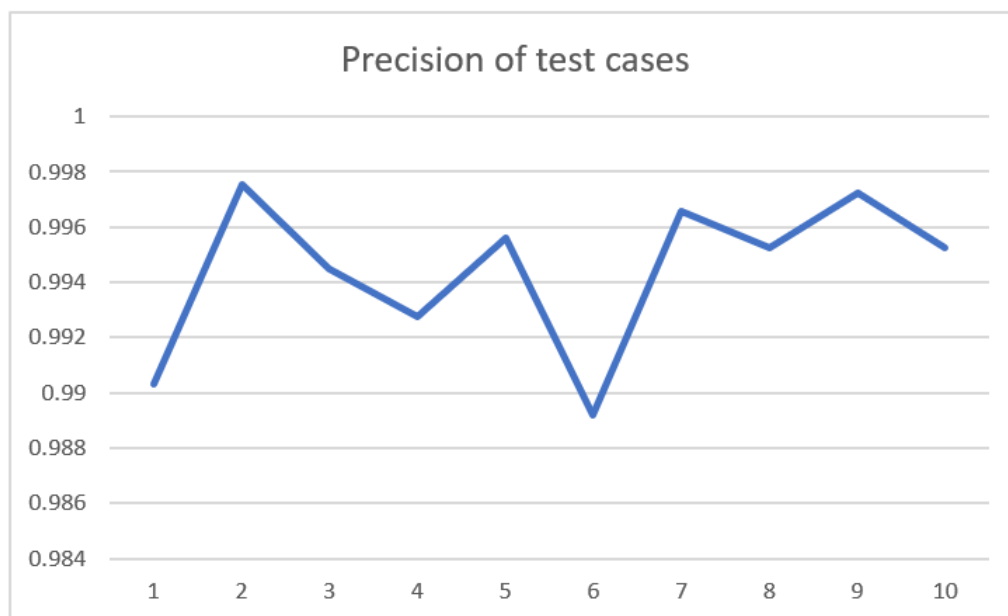- pillow to open, save, and manipulate file formats.

Figure 1: Precision of test cases

The figure above demonstrates the percentage recorded after 10 cases. As we can see from the chart, the number only fluctuates between 0.988 and 0.998, proving the high precision of the program in face recognition.
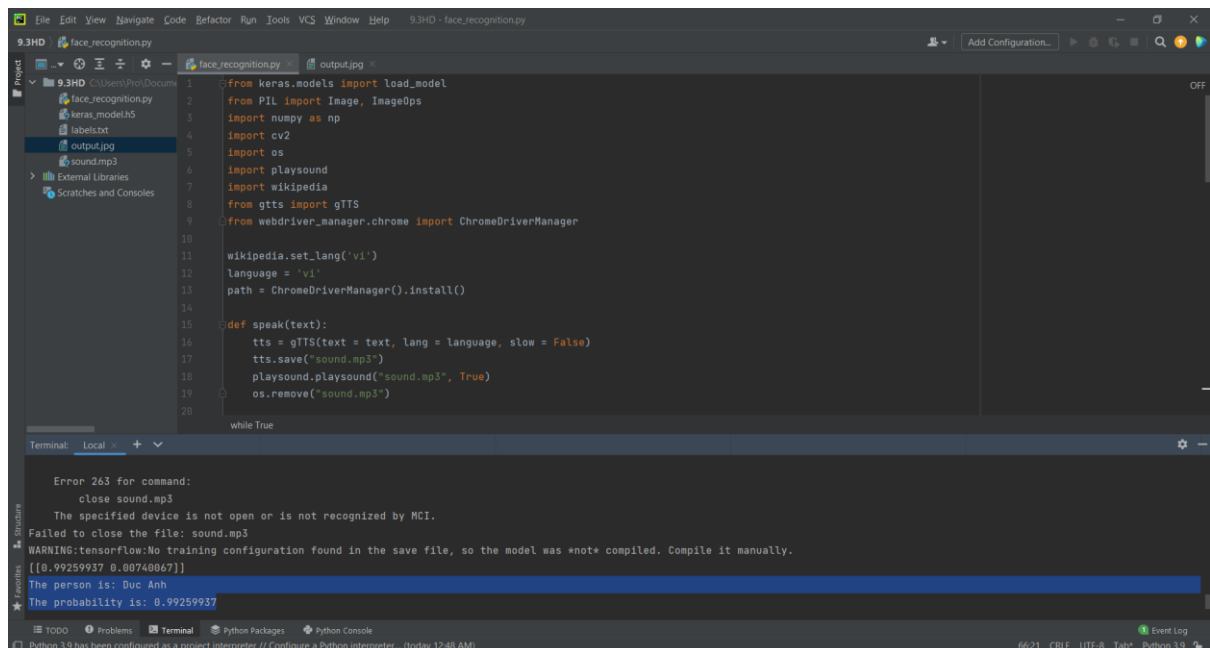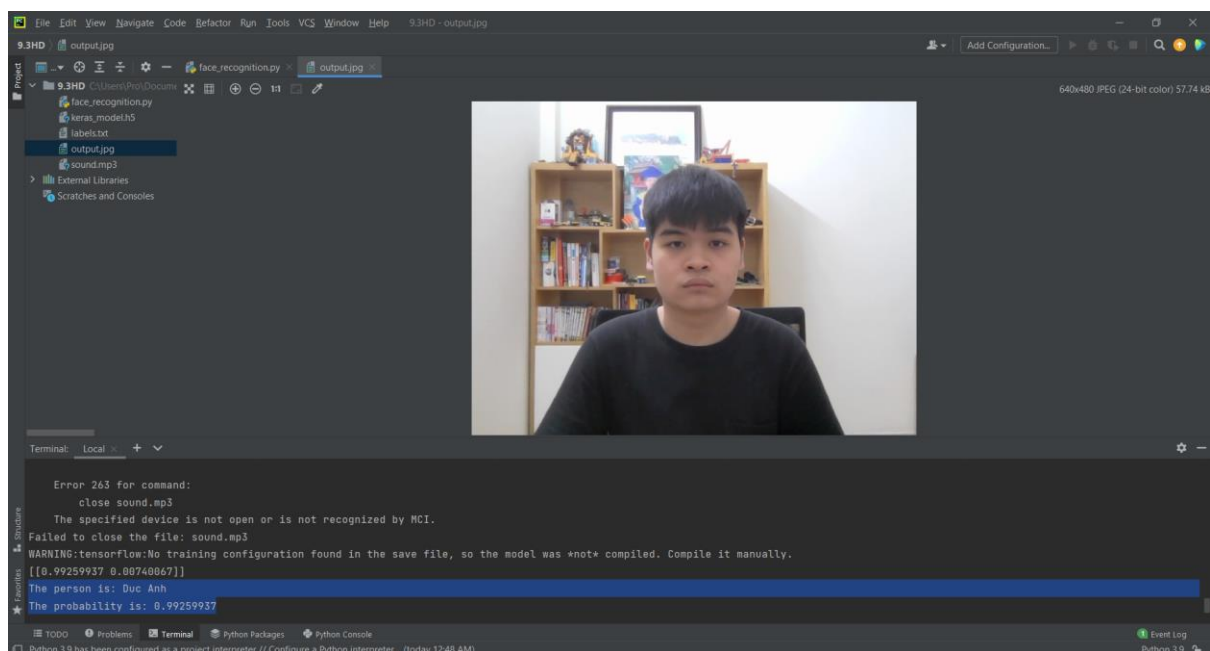

Figure 2: Workspace and code

Figure 3: Program output

For the sake of testing, I only created 2 records, which are Duc Anh (me) and Ca Chua (my niece). You can create your own records by following these steps:
1. Go to Teachable Machine.
2. Create an Image Project.
3. Add your classes.

4. Train the model.
5. Download your model with Tensorflow.
6. Extract the zip file to get keras_model.h5 and labels.txt.
7. Place both files to the folder containing face_recognition.py.
8. Change the value in the name array to your new classes.
9. Run the program face_recognition.py through the terminal.

A link to the program demonstration will be provided in task 11.4D/HD Custom code video.

Required Data Types

Describe each of the records and enumerations you will create using the following table (one per record).

Field Name	Type	Notes
:name	array	Hold the name values (classes).
:index	integer	Position of the array.
:max_index	integer	Max position of the array.

Overview of Program Structure

Functions:

speak(text):
- tts: set the language and text for the webdriver_manager library.
- tts.save: save the sound.mp3 file generated from Chrome.
- playsound: play the sound.mp3 file that reads the user's name.
- os.remove: remove the sound.mp3 file if there is already one existing in the same directory.

capture_image():
- ret, frame: open the camera.
- cv2.imwrite: capture a photo and save it as output.jpg in the same directory.

face_detection():
- model: load the model in the keras_model.h5 file.
- data: create the array of the right shape to feed into the keras model.
- size: resized width and height of the image.
- image: open the image file output.jpg that was just saved.
- image_array: turn the image into a numpy array.
- image.show(): display the resized image.
- normalized_image_array: normalize the image
- data[0]: load the image into the array.
- prediction: run the inference and print out the result.
- speak: record a sentence for the sound.mp3 file.

Structure chart

```
┌─────────────────┐      ┌──────────────────┐      ┌──────────────────┐      ┌──────────────────┐
│ User run the    │ ───► │ The program open │ ───► │ The model is     │ ───► │ The image is     │
│ program         │      │ the camera and   │      │ loaded from the  │      │ resized to a     │
│                 │      │ capture an image │      │ dataset          │      │ size of 224x224  │
└─────────────────┘      └──────────────────┘      └──────────────────┘      └──────────────────┘
```

- User run the program
- The program open the camera and capture an image
- The model is loaded from the dataset
- The image is resized to a size of 224x224
- The image is run through the dataset
- The model makes a prediction
  - If the picture contains a person from the dataset
    - Print out the prediction (person's name) on the terminal
    - Use the speak function to read out loud the person's name
    - The program ends
  - If the picture does not contain a person from the dataset
    - An error occurs
    - The program ends