ĐẠI HỌC ĐÀ NẴNG
PHÂN HIỆU TẠI KON TUM

BÁO CÁO MÔN THỊ GIÁC MÁY TÍNH

Đề tài: Xây dựng mô hình ứng dụng nhận diện cảm xúc khuôn mặt

GVHD : TS. HUỲNH HỮU HƯNG
SVTH : CHU HỮU MẠNH
: BÀNH VĂN KỲ
: TRẦN ĐÌNH KHANH
MSSV : 1817480201017
: 1817480201010
: 1817480201009
LỚP : K12TT

Kon Tum, tháng 01 năm 2022

MỤC LỤC

DANH MỤC CÁC HÌNH

CHƯƠNG 1
LỜI MỞ ĐẦU

Thị giác máy tính là thuật ngữ mô tả một tập hợp các công nghệ cho phép các thiết bị máy tính, phần mềm, robot hoặc bất kỳ thiết bị nào; thu nhận, phân tích và xử lý hình ảnh.

Thị giác máy tính đã và đang là một lĩnh vực phát triển mạnh mẽ ở Việt Nam và trên toàn thế giới. Trong nhiều năm qua đã có rất nhiều các công trình nghiên cứu về thị giác máy tính nói chung và nhận diện khuôn mặt nói riêng. Các công trình nghiên cứu đa dạng các loại nhận diện khuôn mặt từ ảnh đen trắng đến ảnh màu và từ các góc độ thẳng, nghiên khác nhau. Không những vậy, bài toán nhận diện khuôn mặt còn được mở rộng và phát triển ở nhiều khía cạnh khác nhau.

Báo cáo dưới đây là quá trình làm việc, cơ sở lý thuyết và các giải pháp đặt ra để giải quyết bài toán điểm danh ứng dụng nhận diện cảm xúc khuôn mặt. Mô hình được thiết kế dựa trên kỹ thuật nhận diện hình ảnh với ngôn ngữ Python.

Phương án tiếp cận: áp dụng các phương pháp nhận diện khuôn mặt như: Haar Cascade, MTCNN … kết hợp với thư viện Tensorflow để huấn luyện (training) một mô hình (model) mới dựa trên mô hình đã được huấn luyện trước (pre-trained model) để nhận diện khuôn mặt và phân loại nó. Kết quả cuối cùng được hiển thị ra màn hình.

CHƯƠNG 2
TỔNG QUAN ĐỀ TÀI

2.1. GIỚI THIỆU ĐỀ TÀI

Khuôn mặt của con người biểu hiện nhiều cảm xúc mà không cần phải nói ra. Đó là một trong những phương tiện mạnh mẽ và tự nhiên nhất để con người truyền đạt thể hiện cảm xúc. Vấn đề nhận diện cảm xúc đang là một đề tài nghiên cứu có ảnh hưởng to lớn đến cuộc sống.

Hệ thống nhận diện cảm xúc khuôn mặt được sử dụng nhiều trong cuộc sống: điều trị y tế, giao tiếp song ngôn ngữ, đánh giá đau của bệnh nhân, phát hiện nói dối, giám sát trạng thái của người lái xe phát hiện trạng thái buồn ngủ dựa vào cảm xúc trên khuôn mặt được phát triển để cảnh báo cho người lái xe khi thấy dấu hiệu buồn ngủ, mệt mỏi. Cho đến nay, hầu hết các nghiên cứu đã được tiến hành về việc tự đô png hóa nhận dạng nét mặt từ video, biểu cảm giọng nói từ âm thanh, biểu cảm viết từ văn bản và sinh lý học được đo bằng thiết bị đeo được.

Trong báo cáo này, nhóm xin trình bày về việc phân tích và nhận diện cảm xúc khuôn mặt từ video.

2.2. BẢNG PHÂN CÔNG CÔNG VIỆC

Bảng 2.2.1.a.1.1: Bảng phân công công việc

STT	Mô tả công việc	Thành viên thực hiện
1	Bàn luận và lựa chọn đề tài	Cả nhóm
2	Tìm hiểu CNN và các phiên bản mở rộng (ResNet50)	Bành Văn Kỳ
3	Đọc & tìm hiểu các thư viện hỗ trợ	Chu Hữu Mạnh
4	Thu thập hình ảnh làm dataset để phục vụ cho việc huấn luyện (training) và kiểm tra (test)	Trần Đình Khanh
5	Cắt khung hình chứa mặt người và đánh nhãn	Chu Hữu Mạnh
6	Thực việc xử lý ảnh để tạo ra bộ Data để test và training	Trần Đình Khanh Chu Hữu Mạnh
7	Huấn luyện model Keras để nhận diện và phân loại cảm xúc	Bành Văn Kỳ
8	Kiểm tra độ chính xác của các Model	Cả nhóm
9	Ghép model vào dự án và demo	Cả nhóm

CHƯƠNG 3
PHÂN TÍCH BÀI TOÁN

3.1. YÊU CẦU CHUNG

Bài toán nhận diện bằng khuôn mặt có thể chia thành 2 giai đoạn chính, bao gồm:

- Vẽ bounding box và sử dụng webcam để đưa khuôn mặt và đúng khu vực cần nhận diện

- Sau khi đưa khuôn mặt vào đúng khu vực cần nhận diện thì thực hiện nhận diện cảm xúc dựa vào các features bắt được

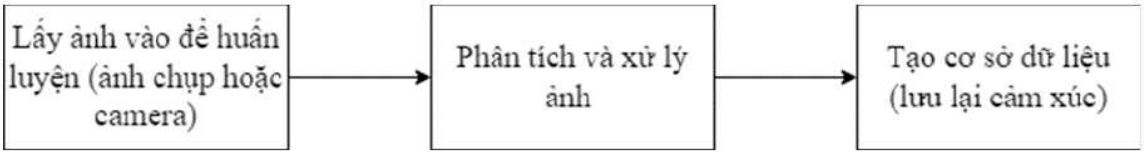
- Cuối cùng, hiển thị tên của cảm xúc nhận diện lên bounding box

3.2. KIẾN TRÚC TỔNG QUAN CỦA HỆ THỐNG

3.2.1. Sơ đồ khối

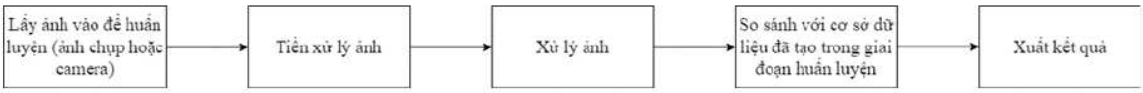
Hệ thống được chia thành thành 2 giai đoạn chính: Huấn luyện và nhận dạng

- Huấn luyện gồm các quá trình: Lấy ảnh đầu vào, phân tích và xử lý ảnh đầu vào, tạo cơ sở dữ liệu.



Hình 3.2.1.a.1: Huấn luyện

- Nhận diện gồm các quá trình: Lấy ảnh đầu vào, xử lý đầu vào, so sánh với cơ sở dữ liệu đã tạo ra trong giai đoạn huấn luyện, đưa ra kết quả



Hình 3.2.1.a.2: Nhận dạng

3.2.2. Tổng quan về hệ thống nhận diện cảm xúc khuôn mặt

Nhận dạng cảm xúc qua khuôn mặt, sử dụng 2 phương pháp chính là SVM và CNN. Mục tiêu của bài viết:

- So sánh phương pháp SVM và CNN trong nhận dạng cảm xúc qua khuôn mặt.

- So sánh phương pháp CNN cơ bản và CNN cơ bản kết hợp các đặc trưng truyền thống nhóm có tìm hiểu 1 số bài báo và 1 số các phần mềm ứng dụng thực tế hiện nay về nhận diện khuôn mặt thì thấy có 2 vấn đề chính và quan trọng nhất cần giải quyết là: Thiếu dữ liệu traning và các biến thể không liên quan đến biểu hiện cảm xúc: ánh sáng, tư thế đầu và sai lệch nhận dạng. Để giải quyết vấn đề trên thì phương pháp nhận dạng cảm xúc qua khuôn mặt được chia thành nhiều hướng theo các tiêu chí khác nhau, chia thành hai loại chính: phương pháp truyền thống và phương pháp hiện đại:

+ Phương pháp truyền thống: Hệ thống nhận dạng cảm xúc qua khuôn mặt với phương pháp truyền thống thì xử lý bài qua các giai đoạn: tiền xử lý hình ảnh khuôn mặt, trích xuất đặc trưng và phân loại.

+ Phương pháp hiện đại: Trong phần này, sẽ mô tả các bước chính phổ biến trong hệ thống nhận dạng cảm xúc qua khuôn mặt thực hiện qua các giai đoạn: tiền xử lý, phân lớp sử dụng học sâu. Những năm gần đây, học sâu có độ chính xác hơn

phương pháp truyền thống vì nó không phải qua bước trích xuất các đặc trưng một cách tường minh, nó sẽ thực hiện đi kèm với phương pháp phân loại.

3.3. CƠ SỞ LÝ THUYẾT VÀ CÔNG NGHỆ SỬ DỤNG

3.3.1. Các công nghệ và chức năng

Bảng 3.3.1.a.1.1: Công nghệ và chức năng sử dụng

Công nghệ	Phiên bản	Chức năng
Python	3.7.9	Là ngôn ngữ được sử dụng để thực hiện đề tài
Opencv	4.2	Thư viện trong việc xử lý ảnh và tạo data
Tensorflow	1.15	Là thư viện chính sử dụng để huấn luyện và nhận diện cảm xúc
Cuda	11.5.1_496.13	Tính toán các thông tin, dữ liệu đồ họa cần được kết xuất.
Tkinter		Thư viện
Keras		Thư viện

3.3.2. Python

Python là ngôn ngữ lập trình hướng đối tượng, cấp cao, mạnh mẽ, được tạo ra bởi Guido van Rossum. Nó dễ dàng để tìm hiểu và đang nổi lên như một trong những ngôn ngữ lập trình nhập môn tốt nhất cho người lần đầu tiếp xúc với ngôn ngữ lập trình. Python hoàn toàn tạo kiểu động và sử dụng cơ chế cấp phát bộ nhớ tự động. Python có cấu trúc dữ liệu cấp cao mạnh mẽ và cách tiếp cận đơn giản nhưng hiệu quả đối với lập trình hướng đối tượng. Cú pháp lệnh của Python là điểm cộng vô cùng lớn vì sự rõ ràng, dễ hiểu và cách gõ linh động làm cho nó nhanh chóng trở thành một ngôn ngữ lý tưởng để viết script và phát triển ứng dụng trong nhiều lĩnh vực, ở hầu hết các nền tảng.

3.3.3. Opencv

OpenCV là tên viết tắt của open source computer vision library – hoàn toàn có thể được hiểu là một thư viện nguồn mở cho máy tính. Cụ thể hơn OpenCV là kho tàng trữ những mã nguồn mở được dùng để giải quyết và xử lý hình ảnh, tăng trưởng những ứng dụng đồ họa trong thời hạn thực. OpenCV được cho phép cải tổ vận tốc của CPU khi triển khai những hoạt động giải trí real time. Nó còn phân phối một số lượng lớn những mã giải quyết và xử lý Giao hàng cho quá trình của thị giác máy tính hay những learning machine khác.

Thư viện OpenCV được phát hành với giấy phép BDS. Do đó các dịch vụ nó cung cấp là hoàn toàn miễn phí và được hạn chế tối đa các rào cản thông thường. Cụ thể, bạn được phép sử dụng phần mềm này cho cả hoạt động thương mại lẫn phi thương mại. OpenCV sở hữu giao diện thiên thiện với mọi loại ngôn ngữ lập trình, ví dụ như C++, C, Python hay Java… Ngoài ra, nó cũng dễ dàng tương thích với các hệ điều hành khác nhau, bao gồm từ Windows, Linux, Mac OS, iOS cho đến cả Android.

Theo tính năng và ứng dụng của OpenCV, có thể chia thư viện này thánh các nhóm tính năng và module tương ứng như sau:

- Xử lý và hiển thị Hình ảnh/ Video/ I/O (core, imgproc, highgui)
- Phát hiện các vật thể (objdetect, features2d, nonfree)
- Geometry-based monocular hoặc stereo computer vision (calib3d, stitching, videostab)
- Computational photography (photo, video, superres)
- Machine learning & clustering (ml, flann)
- CUDA acceleration (gpu)

3.3.4. Tensorflow

Tensorflow là một thư viện có mã nguồn mở, được dùng để tính toán machine learning với quy mô lớn. TensorFlow kết hợp một loạt các mô hình và thuật toán machine learning cùng deep learning, từ đó làm cho chúng trở nên hữu ích bằng những phép toán. TensorFlow sử dụng Python để cung cấp một API front-end thuận tiện cho việc xây dựng các ứng dụng với framework, đồng thời thực thi các ứng dụng đó bằng ngôn ngữ C++ để đạt hiệu suất cao hơn.

Kiến trúc TensorFlow hoạt động được chia thành 3 phần:

- Tiền xử lý dữ liệu
- Dựng model
- Train và ước tính model

Cách thức hoạt động:

- TensorFlow cho phép bạn xây dựng biểu đồ và cấu trúc luồng dữ liệu để mô tả cách dữ liệu di chuyển qua biểu đồ hoặc di chuyển qua một seri mà các node đang xử lý. Mỗi một node trong đồ thị đại diện cho một operation toán học, có thể gọi đây là mảng dữ liệu đa chiều hay tensor.

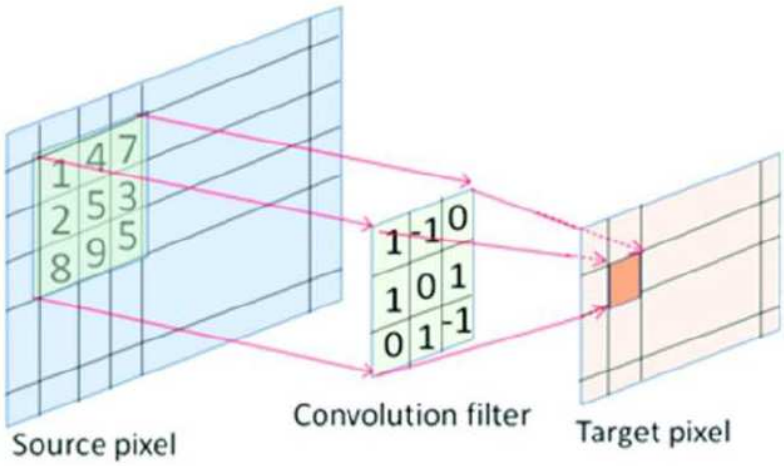
- TensorFlow sẽ cung cấp tất cả thông tin cho lập trình viên bằng ngôn ngữ lập trình Python. Python có nhiệm vụ điều phối các luồng công việc và kết nối chúng lại với nhau. Các node và tensor có trong TensorFlow cũng là những đối tượng của Python.

3.3.5. Convolutional Neural Network

Convolutional Neural Network (CNN hoặc ConvNet) được tạm dịch là: Mạng nơ ron tích tụ. Đây được xem là một trong những mô hình của Deep Learning – tập hợp các thuật toán để có mô hình dữ liệu trừu tượng hóa ở mức cao bằng cách sử dụng nhiều lớp xử lý cấu trúc phức tạp. Hiểu đơn giản, CNN là một lớp của mạng nơ-ron sâu, được áp dụng phổ biến nhất để phân tích hình ảnh trực quan.

a. Convolutional Layer

Trong ba lớp của Convolutional Neural Network, Convolutional Layer được xem là lớp có vai trò quan trọng nhất. Bởi vì Convolutional Layer sẽ đại diện CNN thực hiện mọi phép toán.



Hình 3.3.5.a.1: Convolutional Layer

Khi nhắc đến lớp Convolutional Layer, chúng ta cần làm rõ một số khái niệm đó là: Filter Map, Stride, Padding, Feature Map.

- Filter Map: Nếu như ANN kết nối với từng Pixel của hình ảnh đầu vào thì CNN được sử dụng những Filter để áp vào các vùng của hình ảnh. Những Filter Map này có thể xem là một ma trận 3 chiều, bao gồm những con số và các con số chính là Parameter.

- Stride: Trong Convolutional Neural Network, Stride được hiểu là khi chúng ta dịch chuyển Filter Map theo Pixel và dựa vào giá trị từ trái sang phải. Stride đơn giản là biểu thị sự dịch chuyển này.

- Padding: Padding chính là những giá trị 0 được thêm vào lớp Input.

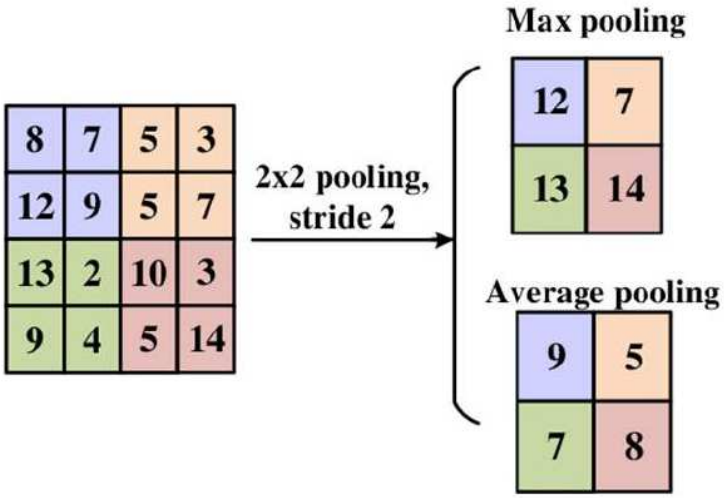
- Feature Map: Đây là kết quả hiển thị sau mỗi lần Filter Map quét qua Input. Cứ mỗi lần quét như vậy, bạn sẽ thấy sự xuất hiện của quá trình tính toán được xảy ra.

b. Pooling Layer

Khi đầu vào quá lớn, các lớp Pooling Layer sẽ được dịch chuyển vào giữa những lớp Convolutional Layer nhằm giảm các Parameter.

Pooling Layer được biết đến với hai loại phổ biến là: Max Pooling và Average Pooling.

Tại Pooling Layer, khi bạn sử dụng lớp Max Pooling thì số lượng Parameter có thể sẽ giảm đi. Vì vậy, Convolutional Neural Network sẽ xuất hiện nhiều lớp Filter Map, mỗi Filter Map đó sẽ cho ra một Max Pooling khác nhau.



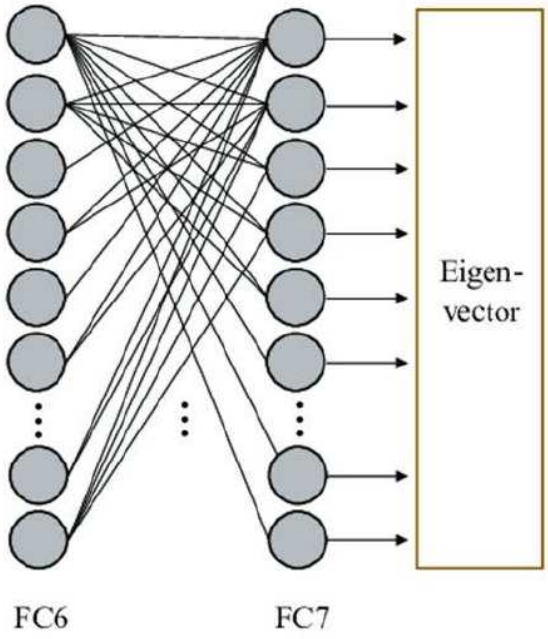
Hình 3.3.5.b.1: Pooling Layer

c. Relu Layer

Đây chính là một hàm kích hoạt trong Neural Network. Chúng ta có thể biết đến hàm kích hoạt này với một tên gọi khác là Activation Function. Nhiệm vụ chính của hàm kích hoạt là mô phỏng lại các Neuron có tỷ lệ truyền xung qua Axon. Trong đó, hàm kích hoạt sẽ bao gồm các hàm cơ bản như: Sigmoid, Tanh, Relu, Leaky Relu, Maxout.

d. Fully Connected Layer

Fully Connected Layer thường sử dụng để đưa ra các kết quả.



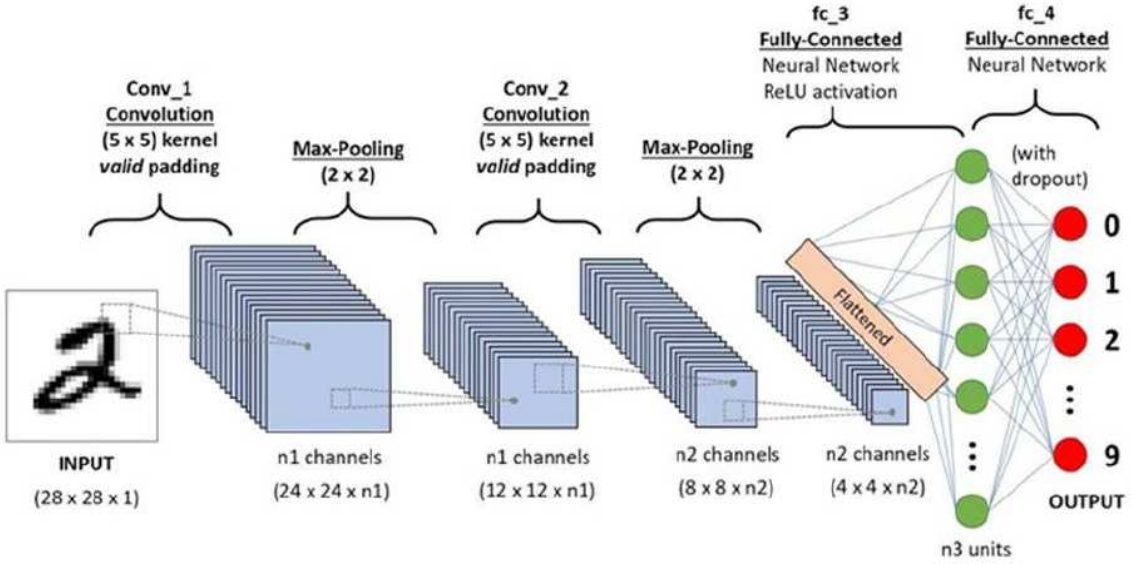
Hình 3.3.5.d.1: Fully Connected Layer

e. Cấu trúc của Convolutional Neural Network

Mạng Convolutional Neural Network là tập hợp nhiều lớp Convolutional chồng lên nhau, sử dụng các hàm Nonlinear Activation và tanh để kích hoạt các trọng số

trong các node. Ở mỗi lớp CNN, sau khi được các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho những lớp tiếp theo. Mỗi Layer kết tiếp sẽ là kết quả Convolution từ Layer trước đó nên chúng ta có được các kết nối cục bộ.

Thông qua quá trình huấn luyện mạng, các lớp Layer CNN tự động học các giá trị được thể hiện qua các lớp Filter.



Hình 3.3.5.e.1: Cấu trúc của Convolutional Neural Network

3.3.6. Cuda

CUDA là từ viết tắt của Compute Unified Device Architecture - Kiến trúc hợp nhất tính toán của các thiết bị điện tử được phát triển độc quyền bởi hãng công nghệ NVIDIA. Còn về CUDA core hay nhân CUDA thì chúng ta có thể hiểu đây là một nhân xử lý trong GPU của card đồ họa - đơn vị chịu trách nhiệm tính toán các thông tin, dữ liệu đồ họa cần được kết xuất. Nhân CUDA tích hợp trong GPU của card đồ họa rời, càng nhiều nhân CUDA thì khả năng tính toán đồng thời nhiều thông tin càng nhanh và chính xác

3.3.7. Tkinter

Tkinter là một gói trong Python có chứa module Tk hỗ trợ cho việc lập trình GUI. Tk ban đầu được viết cho ngôn ngữ Tcl. Sau đó Tkinter được viết ra để sử dụng Tk bằng trình thông dịch Tcl trên nền Python. Ngoài Tkinter ra còn có một số công cụ khác giúp tạo một ứng dụng GUI viết bằng Python như wxPython, PyQt, và PyGTK

3.3.8. Keras

Keras là một open source cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 205 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano.

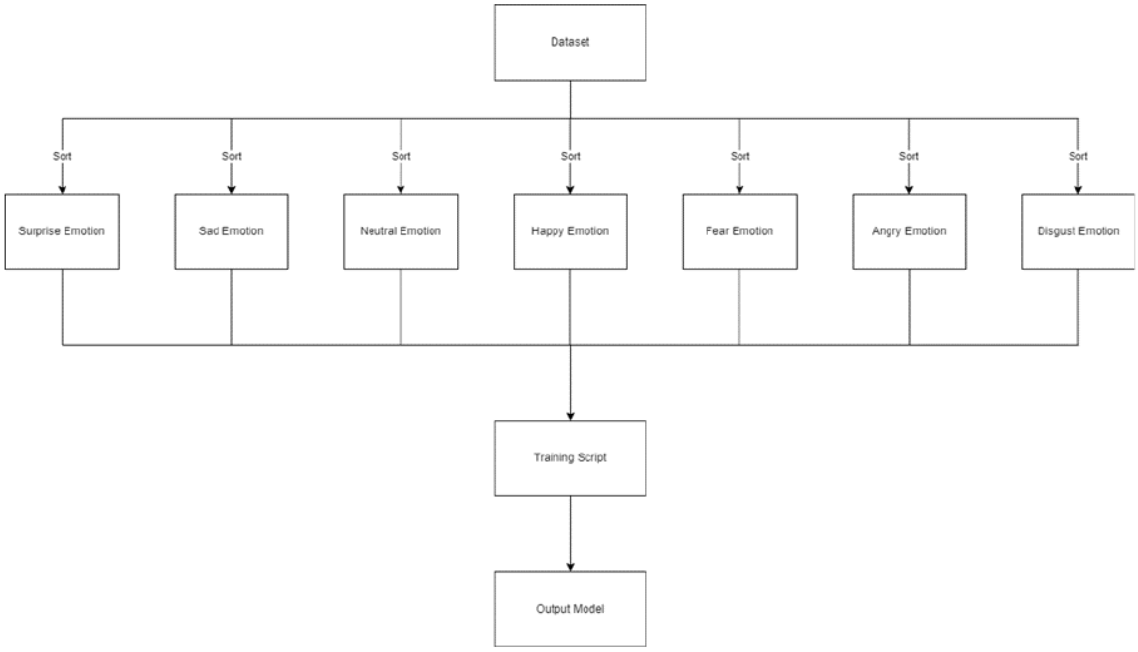
3.3.9.

CHƯƠNG 4
TRIỂN KHAI

4.1. XÂY DỰNG SEQUENTIAL MODEL NHẬN DIỆN CẢM XÚC

4.1.1. Tổng quan

Mục đích của model này dùng để nhận diện được cảm xúc của khuôn mặt trước Camera.

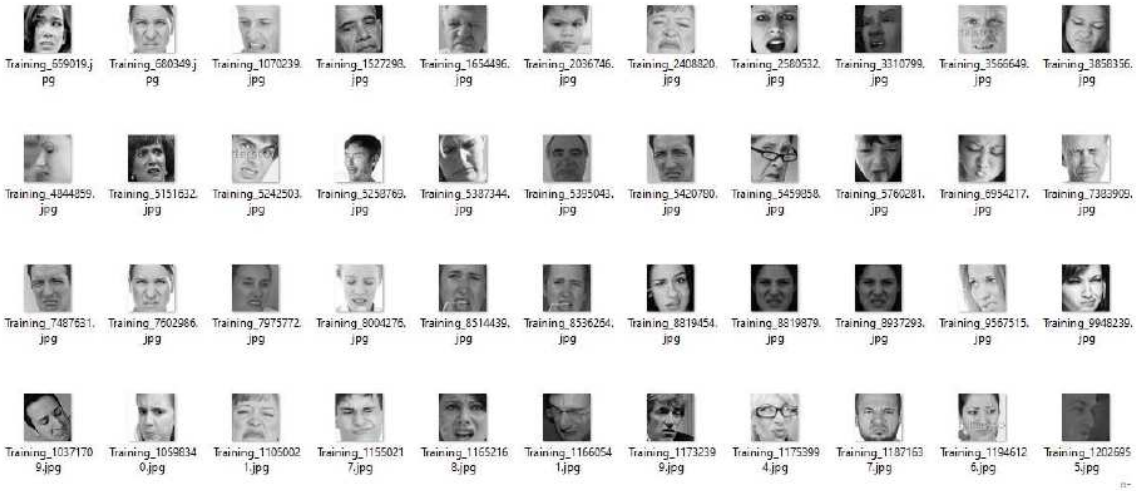


Hình 4.1.1.a.1: Tổng quan

Tạo tập Dataset và đánh nhãn để làm dữ liệu đầu vào cho việc huấn luyện Model. Dataset được chia thành 6 thư mục là surprise (dùng để nhận biết cảm xúc ngạc nhiên), sad (dùng để nhận biết cảm xúc buồn), neutral (dùng để nhận biết cảm xúc không xác định), happy (dùng để nhận biết cảm xúc vui), fear (dùng để nhận biết cảm xúc sợ hãi) và angry (dùng để nhận biết cảm xúc giận dữ). Các bước tiến hành cụ thể như sau:

- Sử dụng Camera của máy tính sử dụng trong hệ thống (hoặc Camera của máy tính nhúng), chế độ Selfie để quay phim.
- Quay một đoạn phim ngắn để thu thập dữ liệu về khuôn mặt.
- Chuyển từ ảnh bình thường sau khi thu thập được sang thành ảnh xám.
- Chuyển ảnh xám về ảnh nhị phân.

Hình 4.1.1.a.2: Tức giận



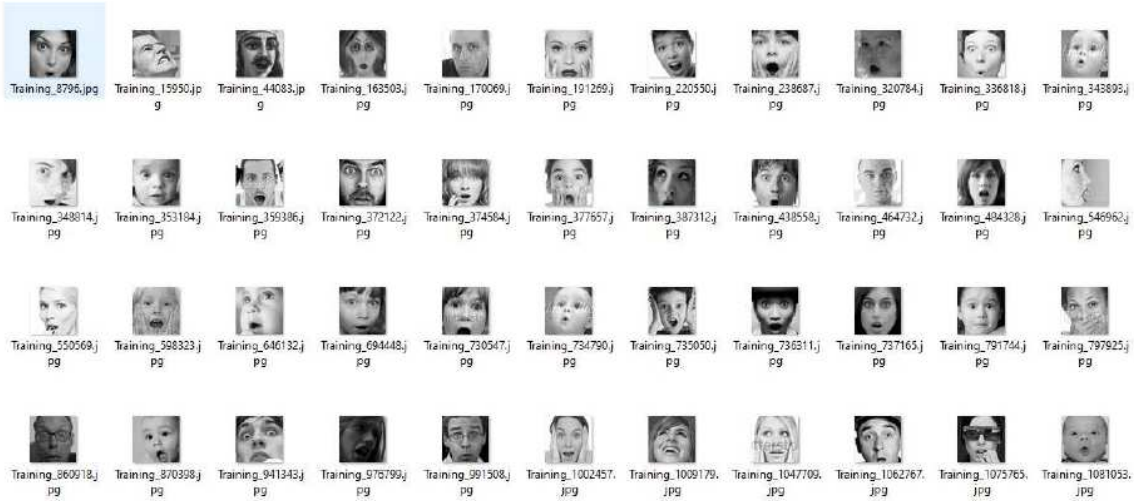
Hình 4.1.1.a.3: Ghê tởm

Hình 4.1.1.a.4: Lo sợ

Hình 4.1.1.a.5: Vui mừng

Hình 4.1.1.a.6: Bình thường

Hình 4.1.1.a.7: Buồn



Hình 4.1.1.a.8: Ngạc nhiên

4.1.2. Áp dụng TensorFlow vào bài toán nhận diện cảm xúc

Sau khi đã có được Dataset và đánh được nhãn, tiến hành cài đặt và chạy Script để xây dựng Sequential Model, với sự hỗ trợ của thư viện Keras. Các bước thực hiện như sau:

- Đầu tiên, import các thư viện có liên quan.

```
import numpy as np
import cv2
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.preprocessing.image import ImageDataGenerator
```

Hình 4.1.2.a.1: Import thư viện

- Khởi tạo mô hình Training và Validation

```
train_dir = 'train'
val_dir = 'test'
train_datagen = ImageDataGenerator(rescale=1./255)
val_datagen = ImageDataGenerator(rescale=1./255)
train_generator = train_datagen.flow_from_directory(
        train_dir,
        target_size=(48,48),
        batch_size=64,
        color_mode="grayscale",
        class_mode='categorical')
validation_generator = val_datagen.flow_from_directory(
        val_dir,
        target_size=(48,48),
        batch_size=64,
        color_mode="grayscale",
        class_mode='categorical')
```

Hình 4.1.2.a.2: Khởi tạo mô hình Training và Validation

- Train_generator: Trình đào tạo CNN
- Validation_generator: Trình xác thực CNN:
  + Taget_size: Là kích thước của hình ảnh đầu vào.
  + Batch_size: Số lượng hình ảnh được tạo ra từ trình tạo mỗi lô.
  + Color_mode: Vì hình ảnh có màu xám nên đặt thành "grayscale".
  + Class_mode: Đặt 'categorical' vì phát triển hệ thống mã tự động, cả đầu vào và đầu ra có thể sẽ là cùng một hình ảnh.
- Xây dựng kiến trúc mạng tích chập CNN

```
emotion_model = Sequential()
emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48,48,1)))
emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Flatten())
emotion_model.add(Dense(1024, activation='relu'))
emotion_model.add(Dropout(0.5))
emotion_model.add(Dense(7, activation='softmax'))
```

Hình 4.1.2.a.3: Xây dựng kiến trúc mạng tích chập CNN

  + Kernel_size: Chỉ định chiều cao vài chiều rộng của cửa sổ tích chập 2D.
  + Activation='relu': Kích hoạt làm relu để lọc các giá trị nhỏ hơn 0.
  + Input_shape: kích thước của dữ liệu đầu vào.
- Training mô hình

```
emotion_model.compile(loss='categorical_crossentropy',optimizer=Adam(lr=0.0001, decay=1e-6),metrics=['accuracy'])
emotion_model_info = emotion_model.fit_generator(
        train_generator,
        steps_per_epoch=28709 // 64,
        epochs=50,
        validation_data=validation_generator,
        validation_steps=7178 // 64)
```

Hình 4.1.2.a.4: Training mô hình

- Lưu lại các trọng số của mô hình

```
emotion_model.save_weights('model.h5')
```

Hình 4.1.2.a.5: Lưu lại các trọng số của mô hình

4.2. SỬ DỤNG OPENCV HAARCASCADE XML PHÁT HIỆN CÁC KHUÔN MẶT TRONG WEBCAM VÀ DỰ ĐOÁN CẢM XÚC.

```
cv2.ocl.setUseOpenCL(False)
emotion_dict = {0: "   Angry   ", 2: "  Fearful  ", 3: "   Happy   ", 4: "  Neutral  ",
                5: "    Sad    ", 6: "Surprised"}
emoji_dist = {0: "./emojis/angry.png",  2: "./emojis/fearful.png", 3: "./emojis/happy.png",
              4: "./emojis/neutral.png", 5: "./emojis/sad.png", 6: "./emojis/surpriced.png"}
cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    if not ret:
        break
    bounding_box = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    num_faces = bounding_box.detectMultiScale(gray_frame,scaleFactor=1.3, minNeighbors=5)
    for (x, y, w, h) in num_faces:
        cv2.rectangle(frame, (x, y-50), (x+w, y+h+10), (255, 0, 0), 2)
        roi_gray_frame = gray_frame[y:y + h, x:x + w]
        cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray_frame, (48, 48)), -1), 0)
        emotion_prediction = emotion_model.predict(cropped_img)
        maxindex = int(np.argmax(emotion_prediction))
        cv2.putText(frame, emotion_dict[maxindex], (x+20, y-60), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE_AA)
    cv2.imshow('Video', cv2.resize(frame,(1200,860),interpolation = cv2.INTER_CUBIC))
    if cv2.waitKey(1) & 0xFF == ord('q'):
        exit(0)
cap.release()
cv2.destroyAllWindows()
```

Hình 4.2.1.a.1: Sử dụng openCV haarcascade xml phát hiện các khuôn mặt trong webcam và dự đoán cảm xúc

4.3. CODE GIAO DIỆN NGƯỜI DÙNG VÀ MAPPING VỚI CÁC EMOJI

Tạo một file main.py để tạo giao diện.

```
def show_vid():
    cap1 = cv2.VideoCapture(0)
    if not cap1.isOpened():
        print("cant open the camera1")
    flag1, frame1 = cap1.read()
    frame1 = cv2.resize(frame1, (600, 500))

    bounding_box = cv2.CascadeClassifier(
        'haarcascade_frontalface_default.xml')
    gray_frame = cv2.cvtColor(frame1, cv2.COLOR_BGR2GRAY)
    num_faces = bounding_box.detectMultiScale(gray_frame, scaleFactor=1.3, minNeighbors=5)

    for (x, y, w, h) in num_faces:
        cv2.rectangle(frame1, (x, y - 50), (x + w, y + h + 10), (255, 0, 0), 2)
        roi_gray_frame = gray_frame[y:y + h, x:x + w]
        cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray_frame, (48, 48)), -1), 0)
        prediction = emotion_model.predict(cropped_img)

        maxindex = int(np.argmax(prediction))
        # cv2.putText(frame1, emotion_dict[maxindex], (x+20, y-60), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE_AA)
        show_text[0] = maxindex
    if flag1 is None:
        print("Major error!")
    elif flag1:
        global last_frame1
        last_frame1 = frame1.copy()
        pic = cv2.cvtColor(last_frame1, cv2.COLOR_BGR2RGB)
        img = Image.fromarray(pic)
        imgtk = ImageTk.PhotoImage(image=img)
        lmain.imgtk = imgtk
        lmain.configure(image=imgtk)
        lmain.after(10, show_vid)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        exit()
```

Hình 4.3.1.a.1: Tạo một file main.py để tạo giao diện

```
def show_vid2():
    frame2 = cv2.imread(emoji_dist[show_text[0]])
    pic2 = cv2.cvtColor(frame2, cv2.COLOR_BGR2RGB)
    img2 = Image.fromarray(frame2)
    imgtk2 = ImageTk.PhotoImage(image=img2)
    lmain2.imgtk2 = imgtk2
    lmain3.configure(text=emotion_dict[show_text[0]], font=('arial', 45, 'bold'))

    lmain2.configure(image=imgtk2)
    lmain2.after(10, show_vid2)


if __name__ == '__main__':
    root = tk.Tk()
    img = ImageTk.PhotoImage(Image.open("logo.png"))
    heading = Label(root, image=img, bg='black')

    heading.pack()
    heading2 = Label(root, text="Photo to Emoji", pady=20, font=('arial', 45, 'bold'), bg='black', fg='#CDCDCD')

    heading2.pack()
    lmain = tk.Label(master=root, padx=50, bd=10)
    lmain2 = tk.Label(master=root, bd=10)

    lmain3 = tk.Label(master=root, bd=10, fg="#CDCDCD", bg='black')
    lmain.pack(side=LEFT)
    lmain.place(x=50, y=250)
    lmain3.pack()
    lmain3.place(x=960, y=250)
    lmain2.pack(side=RIGHT)
    lmain2.place(x=900, y=350)
```

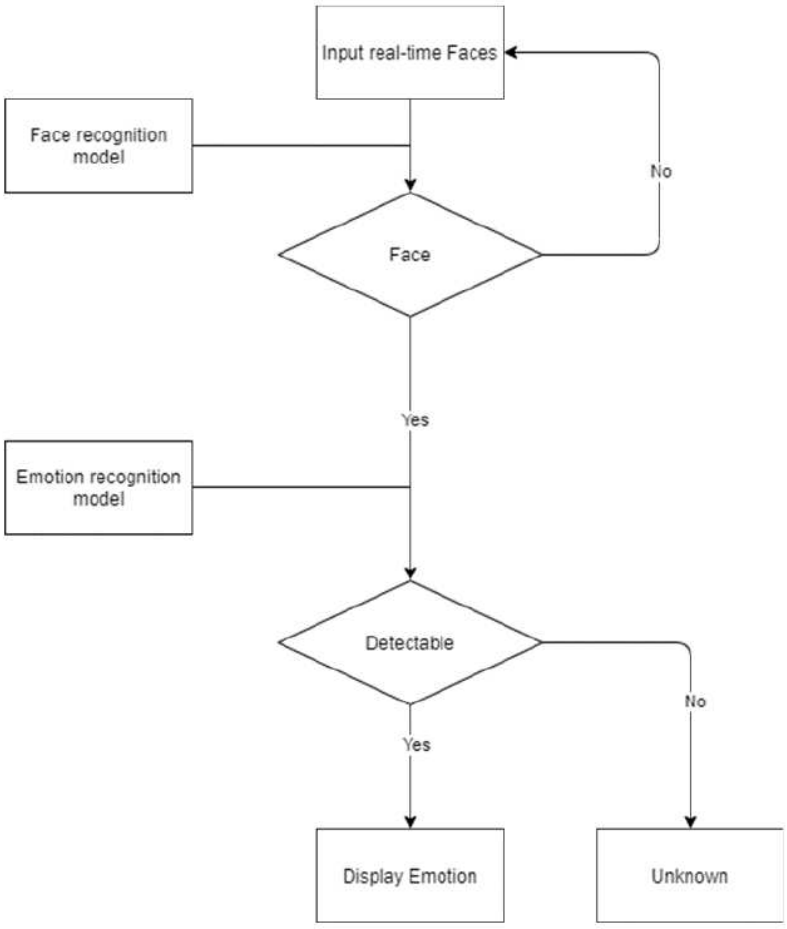
Hình 4.3.1.a.2: Tạo một file main.py để tạo giao diện

```
root.title("Photo To Emoji")
root.geometry("1400x900+100+10")
root['bg'] = 'black'
exitbutton = Button(root, text='Quit', fg="red", command=root.destroy, font=('arial', 25, 'bold')).pack(side=BOTTOM)
show_vid()
show_vid2()
root.mainloop()
```

Hình 4.3.1.a.3: Tạo một file main.py để tạo giao diện

4.4. TÍCH HỢP VÀ HOÀN THIỆN HỆ THỐNG

4.4.1. Tổng quan



Hình 4.4.1.a.1: Sơ đồ tổng quan

4.4.2. Load modes

```
emotion_model = Sequential()
emotion_model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(48, 48, 1)))
emotion_model.add(Conv2D(64, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Conv2D(128, kernel_size=(3, 3), activation='relu'))
emotion_model.add(MaxPooling2D(pool_size=(2, 2)))
emotion_model.add(Dropout(0.25))
emotion_model.add(Flatten())D
emotion_model.add(Dense(1024, activation='relu'))
emotion_model.add(Dropout(0.5))
emotion_model.add(Dense(7, activation='softmax'))
emotion_model.load_weights('model.h5')
cv2.ocl.setUseOpenCL(False)
emotion_dict = {0: "   Angry   ", 2: "  Fearful  ", 3: "   Happy   ", 4: "  Neutral  ",
|   |   |   |   5: "    Sad    ", 6: "Surprised"}
emoji_dist = {0: "./emojis/angry.png",  2: "./emojis/fearful.png", 3: "./emojis/happy.png",
|   |   |   | 4: "./emojis/neutral.png", 5: "./emojis/sad.png", 6: "./emojis/surpriced.png"}
```

Hình 4.4.2.a.1: Load modes

Thực hiện việc load Face detection model và Emotion recognition model vào từ các file train đã thực hiện trước đó và gán nhãn cho emotion.

4.4.3. Mở camera và nhận diện

Sau khi load được các model vào ta tiến hành kết hợp nhận diện với thời gian thực như sau.

```
cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    if not ret:
        break
    bounding_box = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    num_faces = bounding_box.detectMultiScale(gray_frame,scaleFactor=1.3, minNeighbors=5)
    for (x, y, w, h) in num_faces:
        cv2.rectangle(frame, (x, y-50), (x+w, y+h+10), (255, 0, 0), 2)
        roi_gray_frame = gray_frame[y:y + h, x:x + w]
        cropped_img = np.expand_dims(np.expand_dims(cv2.resize(roi_gray_frame, (48, 48)), -1), 0)
        emotion_prediction = emotion_model.predict(cropped_img)
        maxindex = int(np.argmax(emotion_prediction))
        cv2.putText(frame, emotion_dict[maxindex], (x+20, y-60), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 255, 255), 2, cv2.LINE_AA)
    cv2.imshow('Video', cv2.resize(frame,(1200,860),interpolation = cv2.INTER_CUBIC))
    if cv2.waitKey(1) & 0xFF == ord('q'):
        exit(0)
cap.release()
cv2.destroyAllWindows()
```

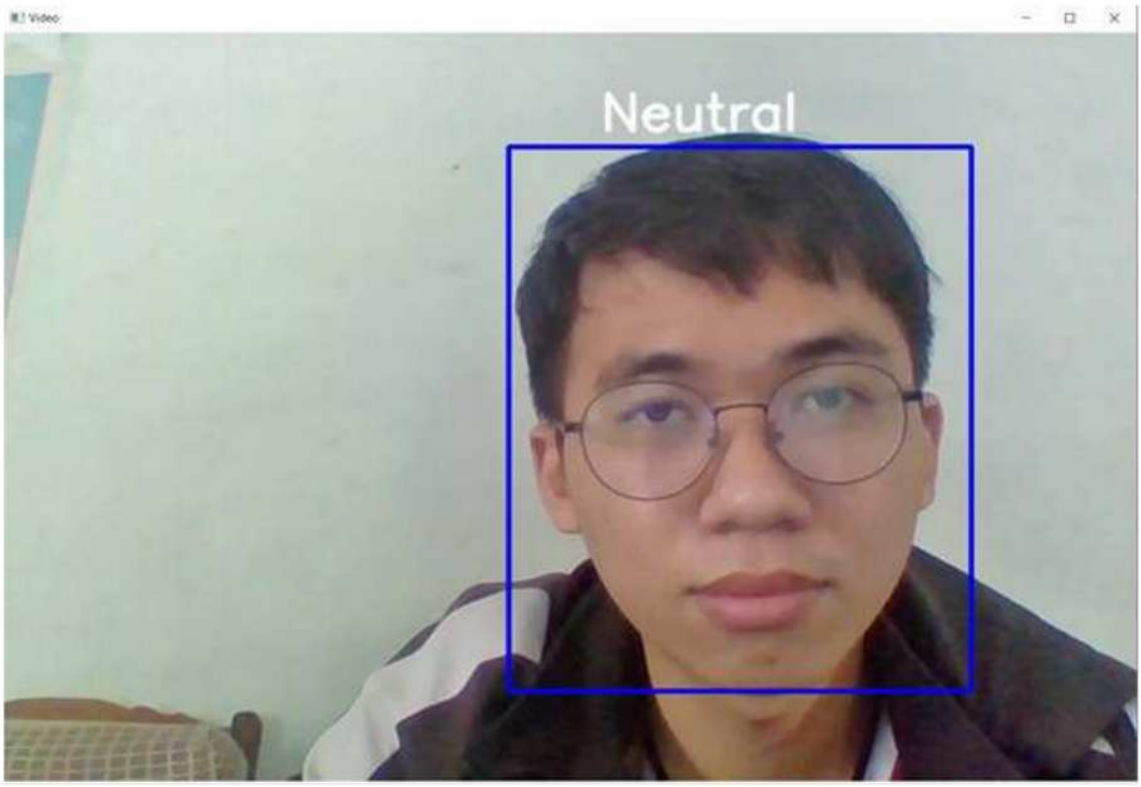
Hình 4.4.3.a.1: Mở camera và nhận diện

+ Đầu tiên ta tiến hành mở camera bằng hàm ViddeoCaptrue(0)
+ Chuyển đổi ảnh vào thành ảnh xám
+ Thực hiện nhận diện khuôn mặt bằng model đã load vào trước đó
+ Vẽ rectangle bao quanh khuôn mặt đã nhận diện được và resize với đúng kích thước bằng hàm cv2.resize()
+ Sau khi nhận diện được khuôn mặt ta tiến hành nhận diện cảm xúc bằng Emotion detection model dã load vào
+ So sánh ảnh khuôn mặt đã detect được với kết quả mà model mà load vào và đưa ra kết quả.

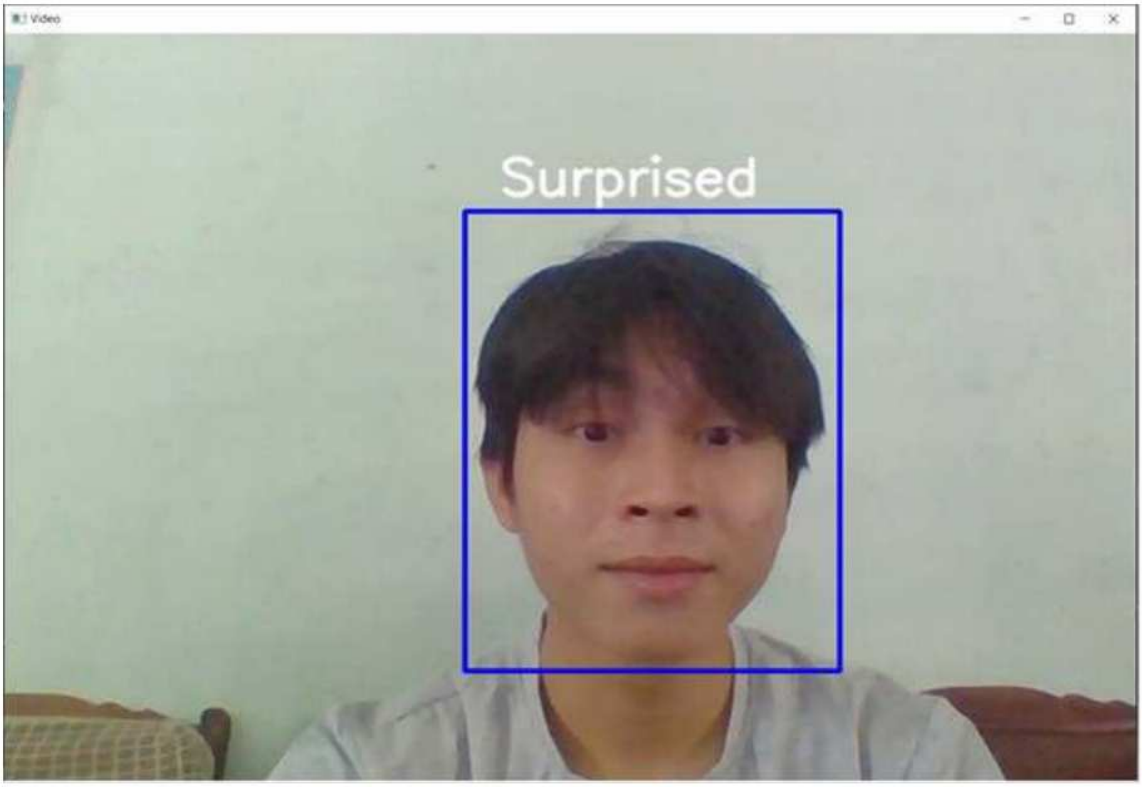
CHƯƠNG 5
KẾT QUẢ THU ĐƯỢC VÀ CÁC VẤN ĐỀ GẶP PHẢI

5.1. KẾT QUẢ THU ĐƯỢC

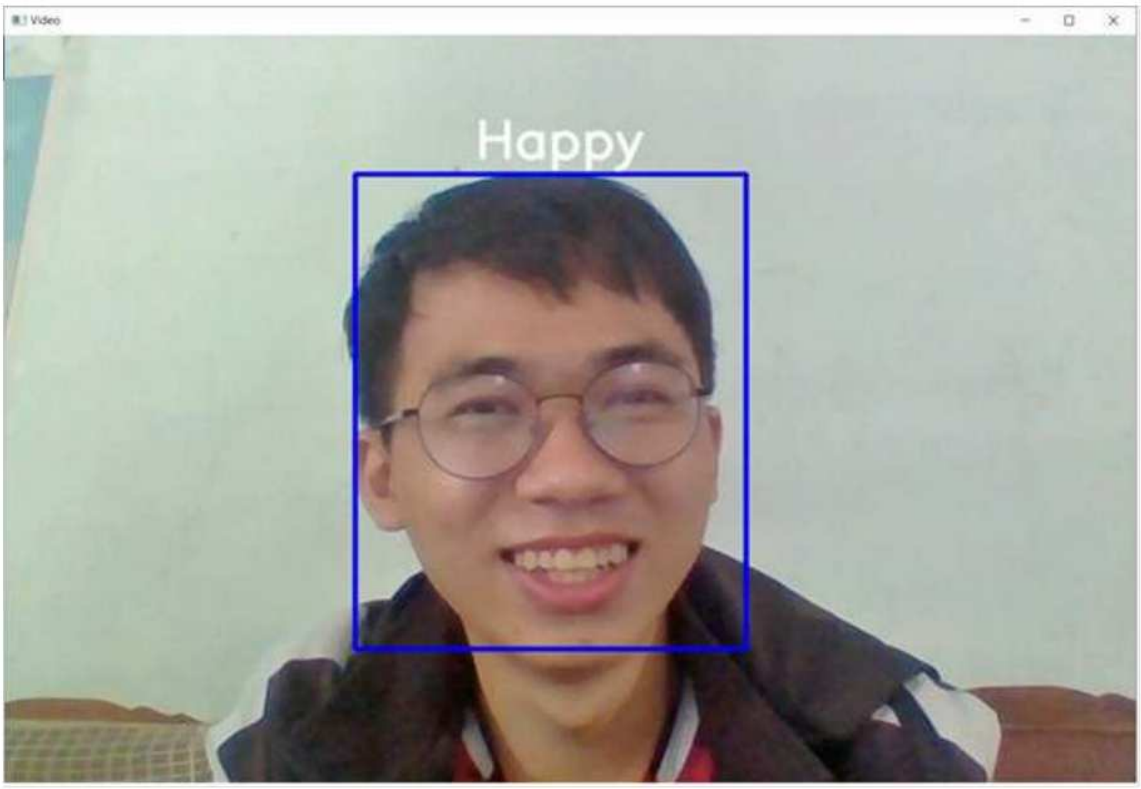
5.1.1. Nhận diện thông thường



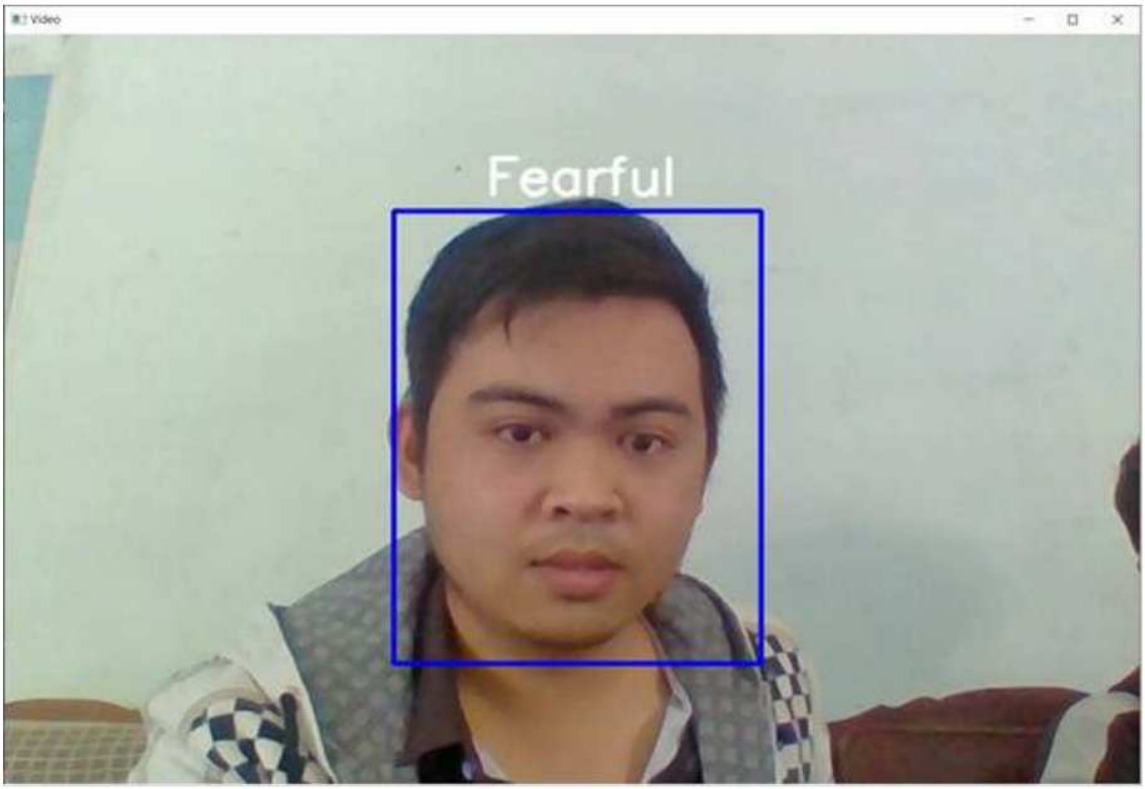
Hình 5.1.1.a.1: Nhận diện cảm xúc bình thường



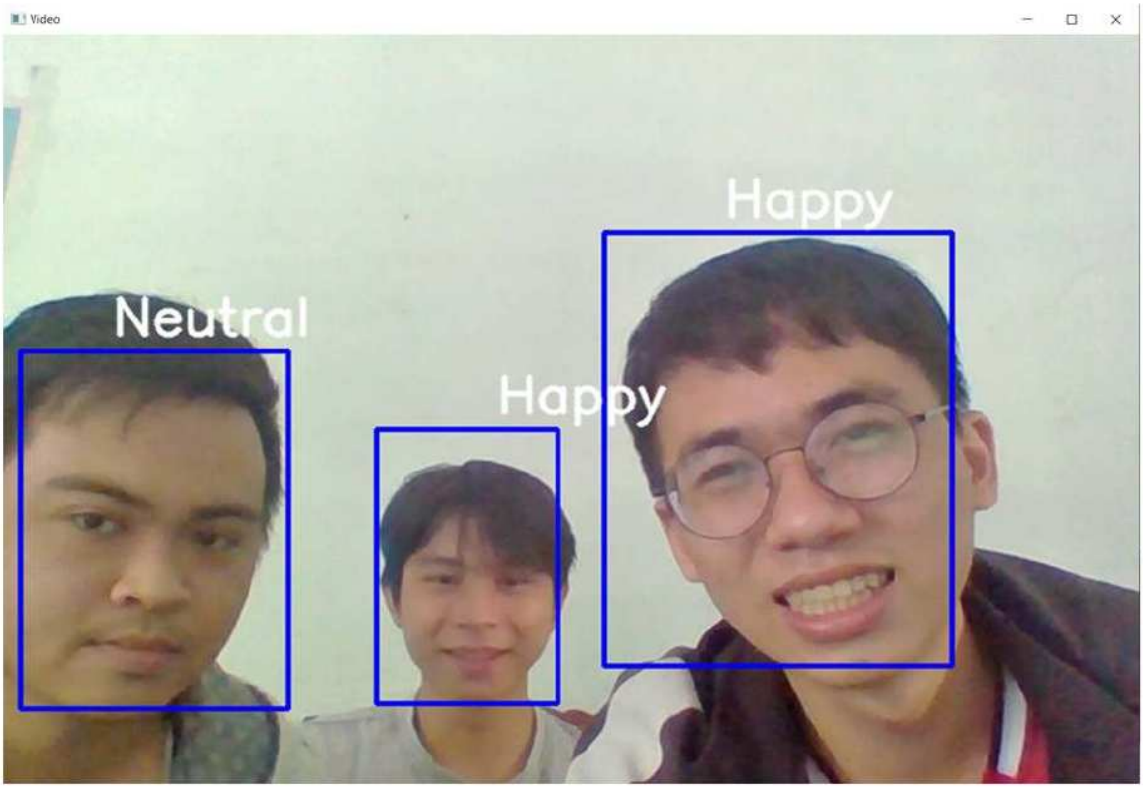
Hình 5.1.1.a.2: Nhận diện cảm xúc ngạc nhiên



Hình 5.1.1.a.3: Nhận diện cảm xúc vui mừng

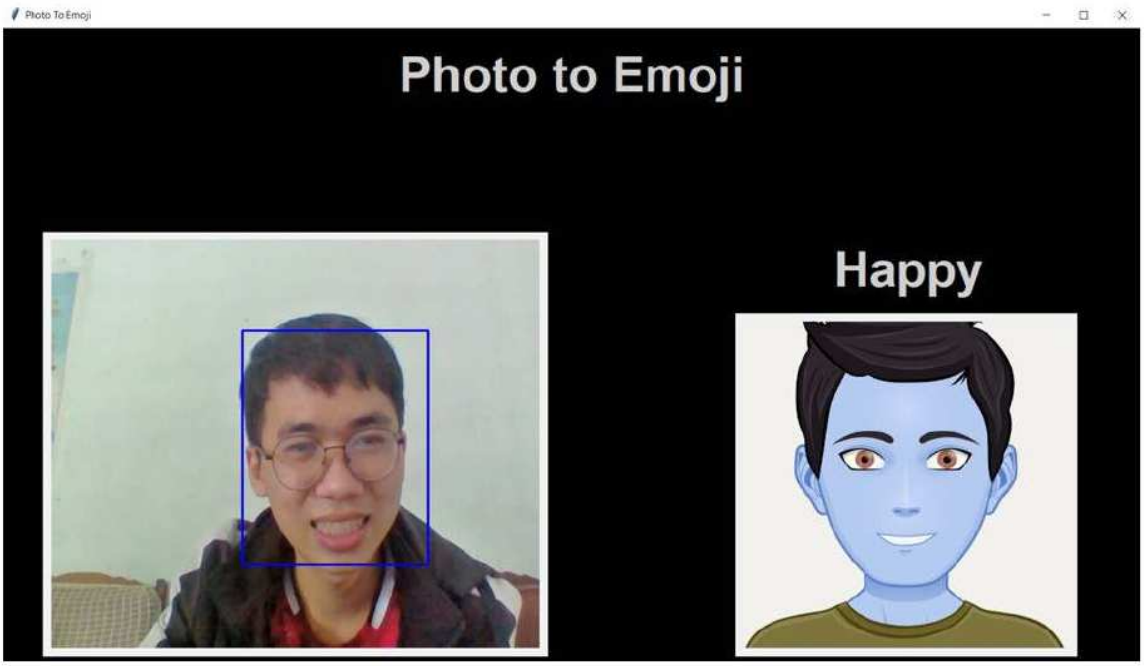


Hình 5.1.1.a.4: Nhận diện cảm xúc sợ hãi

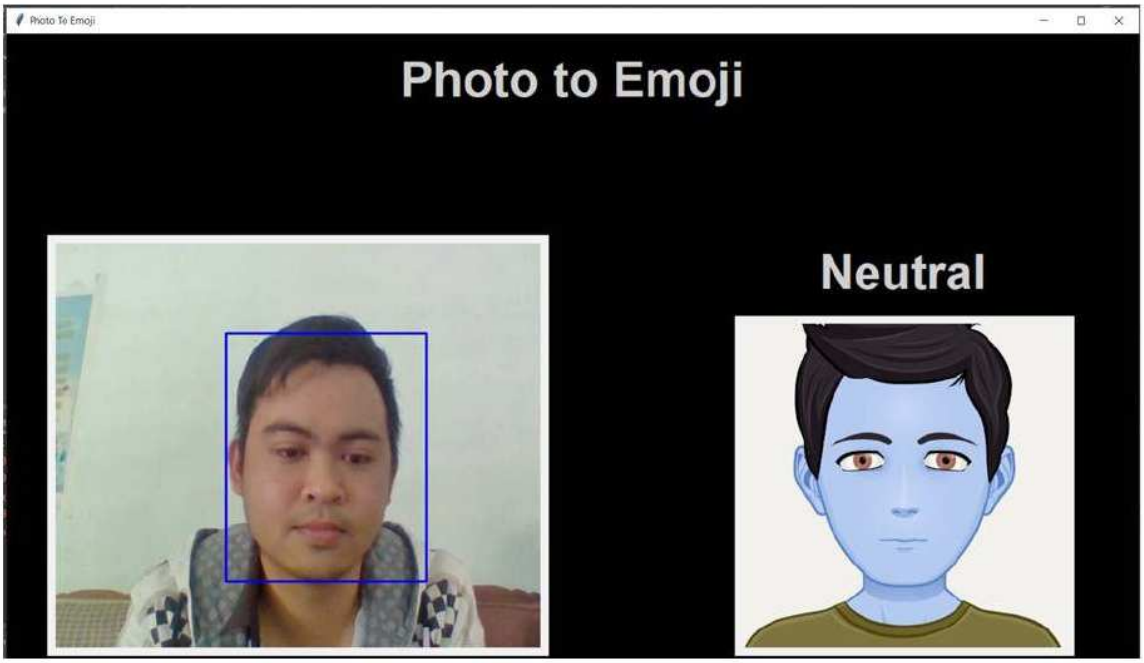


Hình 5.1.1.a.5: Nhận diện cảm xúc của 3 thành viên

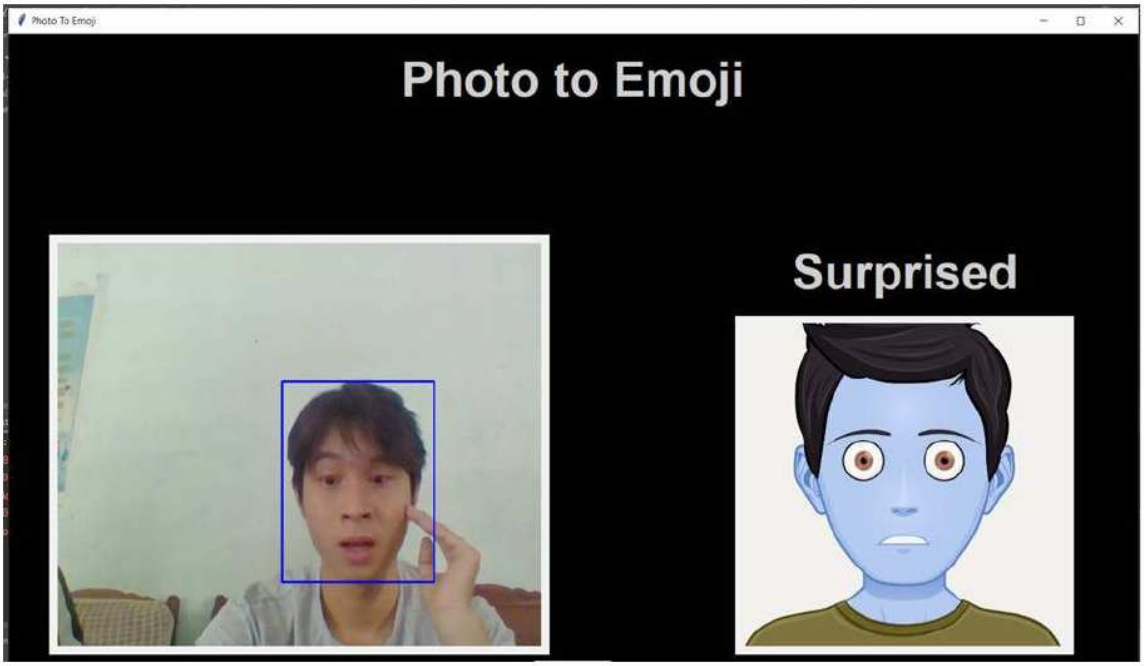
5.1.2. Nhận diện có biểu tượng cảm xúc



Hình 5.1.2.a.1: Cảm xúc vui mừng



Hình 5.1.2.a.2: Cảm xúc bình thường



Hình 5.1.2.a.3: Cảm xúc sợ hãi

5.2. CÁC VẤN ĐỀ ĐÃ THU ĐƯỢC

- Xây dựng được một ứng dụng cơ bản để nhận diện được cảm xúc thời gian thực
- Giải quyết được các vấn đề đã đề ra ban đầu.
- Vận dụng được các kiến thức đã học ở môn Thị giác máy tính và Trí tuệ nhân tạo.
- Phát triển và nâng cao khả năng làm việc nhóm, tư duy giải quyết vấn đề.

5.3. CÁC VẤN ĐỀ CẦN CẢI THIỆN

- Nhận diện khuôn mặt còn khá hạn chế và phụ thuộc nhiều vào điều kiện lý tưởng (ánh sáng, góc độ của khuôn mặt, …).

- Cảm xúc nhận diện được vẫn chưa ổn định và đạt đô p chính xác cao.
- Vẫn còn sự nhầm lẫn trong việc nhận diện.

CHƯƠNG 6
TÀI LIỆU THAM KHẢO

[1] Thắng, N.C (2019) Nhận diện khuôn mặt trong Video bằng Facenet và MTCNN – Link: https://www.miai.vn/2019/09/11/face-recog-2-0-nhan-dien-khuon-mat-trong-video-bang-mtcnn-va-facenet/

[2] Adrian Rosebrock (March 11,2019) Liveness Detection with OpenCV – Link: https://www.pyimagesearch.com/2019/03/11/liveness-detection-with-opencv/

[3] Https://www.tensorflow.org/api_docs

[4] Https://forum.machinelearningcoban.com/t/kien-truc-cac-mang-cnn-noi-tieng-phan-1-alex-lenet-inception-vgg/2582/