**Thực hiện lập trình phát hiện khuôn mặt bằng phương pháp Haar\_cascade và thư viện Media pipe**

README:

Bước 1: Tải folder trên github và giải nén

Bước 3: Mở folder trong phần mềm lập trình Visual Code hoặc PyCharm

Bước 4: Tạo môi trường ảo: “py -3 -m venv venv”

Bước 4: Ở khung Terminal: vào môi trường ảo bằng lệnh “venv\scripts\activate”

Bước 5: Tải thư viện cần thiết là openCV gõ lệnh “pip install opencv-python”, thư viện mediapipe gõ lệnh “pip install mediapipe”.

*\*Với phương pháp Haar\_cascade:*

Bước 6: Chạy chương trình bằng lệnh “python face.py”

*\*Với phương pháp Media pipe*

Bước 6: Chạy chương trình bằng lệnh “python face2.py”

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Vậy thế nào là phương pháp Haar\_cascade và Media pipe ?

Nhận diện bằng phương pháp Haar\_cascade:

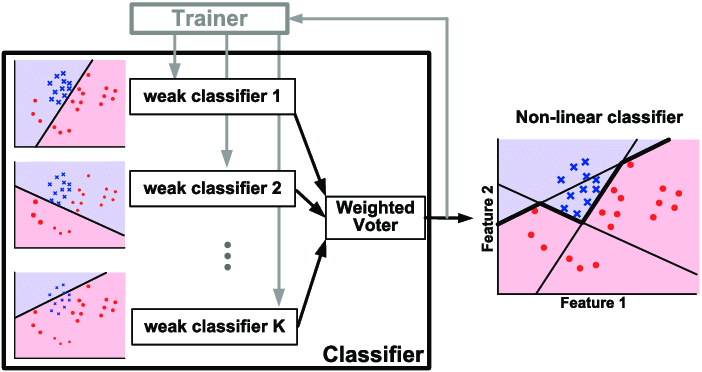
Haar\_cascade là một mô module phát hiện đối tượng do OpenCV trong một dự án viết ra, trong đó có “**haarcascade\_frontalface\_default.xml**” là một model được xây dựng và đóng gói thành một file .xml, model này được sử để hổ trợ để nhận diện khuôn mặt con người.

Về cách hoạt động của model Haar\_cascade:

Nói một cách đơn giản là model này sẽ sử dụng các bộ lọc Haar, các bộ lọc này chỉ trích xuất các đặc trưng Haar (các đặc trưng của khuôn mặt), ta sẽ trượt (cascade) bộ lọc này trên một bức ảnh, việc này giống với các trượt của sổ bộ lọc trong mô hình nơ tron tích chập CNN. Qua đó ta có thể trích xuất ta được các đặc điểm trên khuôn mặt từ đó phát hiện được khuôn mặt.

Model sẽ sử dụng một cửa sổ trượt có kích thước 20x20 pixel, trong cửa sổ trước sẽ chứa các bộ lọc Haar khác nhau, bộ lọc Haar cũng không khác bộ lọc bình thường, chỉ là nó sẽ trích suất ra các đặt trưng quan trong của khuôn mặt mà thôi. Ví dụ như bộ lọc cạnh ngang sẽ được đặt ở vị trí giữa con mắt với lông mày vì ở đâu có sự khác này về màu sắc hay bộ lọc cạnh thẳng sẽ đặt ở sống mũi vì nó khác với hai bên má về góc độ ánh sáng hay bộ lọc lấy phần trung tâm đặt ở con mắt, … các bộ lọc như thế sẽ được đặt trong cửa sổ trượt, và cứ mỗi lần di chuyển của sổ trước các bộ lọc sẽ thực thi nhiệm vụ của nó. Nếu tất cả bộ lọc để bắt được hình ảnh có chứa khuôn mặt thì nó sẽ ngừng khung cửa sổ trượt lại và lưu các đầu ra của từng bộ lọc lại, sau đó mới tiếp tục trượt, nếu không thì sẽ bỏ qua và bỏ kết quả các bộ lọc. Sau khi trượt quả một lần và thu lại kết quả, cửa sổ trượt và bộ lọc có thể sẽ tiến hành nhỏ đi để trượt thêm một lần nữa trên kết quả vừa rồi, đến khi bộ lọc nhỏ vừa đủ với kích thước khuôn mặt trên hình thì ngừng lại và cho ra kết quả.

Trên thực tế có rất nhiều loại bộ lọc khoản 160k bộ lọc khác nhau nhưng để chọn một lại bộ lọc tốt thì không dể. Vì lẽ đó thay vì chọn bộ lọc ta sẽ tiến hành cải tiến nó để cho ra kết quả tốt hơn và phương pháp cải tiến là Adaboost. Phương pháp này sẽ thực hiện cải tiến các bộ lọc bằng cách gộp các bộ lọc lại với nhau (đa phần là các bộ lọc tuyến tính) để tạo thành bộ lọc phi tuyến tính có tác dụng lọc tốt hơn.



*Hình 1. Hình ảnh minh họa phương pháp Adaboost*

Để rõ ràng hơn về phương pháp lọc, phát hiện khuôn mặt bằng haar\_cascade, ta có thể xem quá trình thực hiện của nó thông qua clip sau:

https://www.youtube.com/watch?v=L0JkjIwz2II

Nhận diện bằng thư viện Media pipe:

Media pipe là một gói thư viện của python, gói thư viện này chứa nhiều công cụ hỗ trợ cho người lập trình xây dựng các bài toán nhận diện khuôn mặt, nhận diện thân hình, nhận diện đối tượng, … chủ yếu là dùng để giải quyết các bài toán về computer vision.

“Chi tiết về code xin mời xem video demo và giải thích trong cùng thư mục”