ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP 2

NHẬN DẠNG BẰNG DEEP LEARNING

Bộ môn: Sinh trắc học
Lớp: 20_21
GVLT: Lê Hoàng Thái
GVTH: Lê Thanh Phong

Sinh viên thực hiện:
Lê Hoài Phong – 20120545

Thành phố Hồ Chí Minh, ngày 10 tháng 12 năm 2023

LỜI CAM ĐOAN

Em xin cam đoan đây là bài làm của bản thân, do chính em nghiên cứu và thực hiện. Em xin cam đoan không sao chép bài làm của người khác hoặc công trình của các tác giả khác. Mọi kết quả thực nghiệm trong báo cáo là hoàn toàn có thật và được thực nghiệm kĩ lưỡng, không khai khống số liệu. Các tài liệu tham khảo được sử dụng đều được trích dẫn cụ thể và có nguồn gốc rõ ràng. Nếu sai, em xin hoàn toàn chịu trách nhiệm.

Người cam đoan

Lê Hoài Phong

Mục lục

A.THÔNG TIN CHUNG

I. Thông tin sinh viên

Họ và tên	Lê Hoài Phong
Mã số sinh viên	20120545
SĐT	0387671963
Email	20120545@student.hcmus.edu.vn

II. Tổng quan bài làm

Xây dựng hệ thống nhận dạng sử dụng Deep learning và đánh giá mô hình.

Đề bài gồm 4 bài tập. Em thực hiện **Bài tập 1:** Nhận dạng khuôn mặt.

Trong bài tập này, em đã thực hiện:

- Xây dựng **4 hệ thống** khác nhau, tổ hợp từ hai bộ detector (MTCNN, Yolov8) và hai bộ extractor (FaceNet, VggFace). Mục tiêu là để so sánh giữa FaceNet và VggFace, cũng như tìm hiểu detector ảnh hưởng đến kết quả như thế nào.
- Xây dựng một hệ thống đơn giản để nhận diện **realtime** thông qua webcam.
- Tìm hiểu và báo cáo tổng quan lý thuyết.

III. Cấu trúc bài nộp

Bài nộp gồm File báo cáo và hai file code (một file để huấn luyện mô hình và một file để nhận diện realtime).

Nội dung báo cáo bao gồm: Phần B trình bày tìm hiểu lý thuyết, Phần C trình bày kiến trúc các mô hình, Phần C trình bày kết quả thực nghiệm và minh chứng.

Bài nộp **không** bao gồm dataset, bộ trọng số, … vì kích thước rất lớn. Nếu thầy muốn thực thi lại code hoặc kiểm tra kết quả thì có thể xem ở link drive này.

IV. Tự đánh giá

Đánh giá chung: hoàn thành tốt bài tập, có thực hiện sâu hơn so với yêu cầu đề bài.

Thuận lợi:

- Có nhiều bài tập, có thể lựa chọn bài tập vừa sức
- Thời gian deadline phù hợp.
- Hướng tiếp cận Deep learning dễ hiểu và dễ làm quen

Khó khăn:

- Huấn luyện mô hình tương đối nặng và mất nhiều thời gian và công sức.
- Gặp nhiều lỗi xuất phát từ thư viện.

B. KHẢO SÁT BÀI TOÁN

I. Bài toán nhận diện khuôn mặt

Nhận diện khuôn mặt là một trong những bài toán quan trọng trong lĩnh vực thị giác máy tính (computer vision). Bài toán này đòi hỏi máy tính có khả năng phát hiện và nhận diện khuôn mặt của con người từ hình ảnh hoặc video. Nhận diện khuôn mặt có nhiều ứng dụng thú vị, bao gồm hệ thống đăng nhập bằng khuôn mặt, theo dõi giám sát, xác minh danh tính và nhiều ứng dụng trong ngành công nghiệp và an ninh.

Một số thuật ngữ liên quan:

- *Face Detection*: xác định có khuôn mặt người trong ảnh hay không và tìm vị trí khuôn mặt trong ảnh.
- *Face Recognition*: tìm tên hoặc nhãn tương ứng với mặt người trong ảnh.
- *Face Verification*: xác định hai ảnh có cùng chứa mặt một người hay không.
- *Face Attendence*: là bài toán thực tế, thực hiện điểm danh/chấm công qua khuôn mặt, phương pháp phổ biến là dùng Face Recognition.

Có nhiều kiến trúc và phương pháp khác nhau để giải quyết bài toán nhận diện khuôn mặt. Dưới đây là các thuật toán/phương pháp truyền thống:

- Local Binary Pattern (LBP - 1996): Một phương pháp đơn giản nhưng hiệu quả, dựa trên các đặc trưng cục bộ của khuôn mặt.
- Principal Component Analysis (PCA): Sử dụng phân tích thành phần chính để trích xuất đặc trưng quan trọng từ hình ảnh khuôn mặt.
- HOG (Histogram of Oriented Gradients): Dựa vào việc tính toán biểu đồ hướng gradient của hình ảnh.
- FisherFace
- EigenFace
- SIFT ...

Các phương pháp mạng học sâu hiện đại sử dụng mạng nơ-ron sâu (deep neural networks), đặc biệt là Convolutional Neural Networks (CNNs), để học các đặc trưng từ dữ liệu: DeepFace, VGGFace, FaceNet ...

II. Một số thuật toán/hướng tiếp cận cho bài toán nhận diện khuôn mặt

1. One-shot learning

One-shot learning là phương pháp học có giám sát, mà mỗi người chỉ cần một vài ảnh duy nhất để nhận diện. Tuy nhiên, điểm yếu của phương pháp này là cần huấn luyện lại thường xuyên khi có người mới xuất hiện, làm thay đổi shape của output.

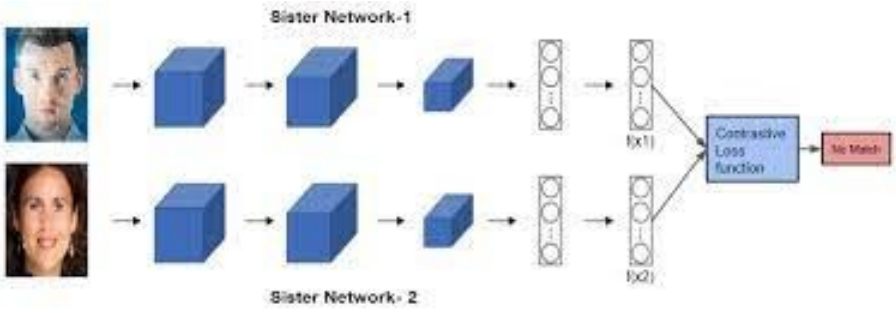
2. Learning similarity

Phương pháp này dựa trên đo khoảng cách giữa các ảnh để xác định xem chúng thuộc về cùng một người hay không. Không cần huấn luyện lại khi có thêm người mới, và không phụ thuộc vào số lượng classes.

$$\begin{cases} d(\text{img1}, \text{img2}) \leq \tau & \rightarrow \quad \text{same} \\ d(\text{img1}, \text{img2}) > \tau & \rightarrow \quad \text{different} \end{cases}$$

3. Siamese network

Siamese network dựa trên base network là Convolutional Neural Network (CNN) đã được loại bỏ output layer, tạo embedding cho hai ảnh và sử dụng hàm loss function để đo sự khác biệt giữa chúng. Mục tiêu chính là tìm biểu diễn của ảnh trong không gian n chiều, giúp tránh vấn đề output shape thay đổi. Siamese network có thể dùng cho bài toán nhận diện vật thể, chữ ký (nhận dạng tổng quan chứ không nhất thiết là nhận dạng mặt người), do đó độ chính xác cũng thấp hơn.



Ưu điểm và Nhược điểm

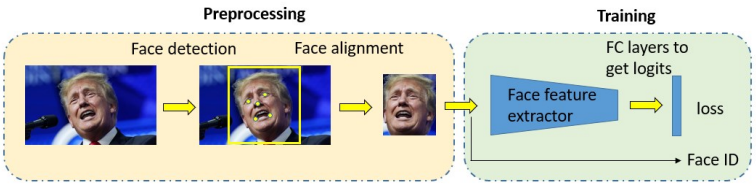
- One-shot learning: Dễ triển khai nhưng cần huấn luyện lại thường xuyên.
- Learning similarity: Không cần huấn luyện lại khi có người mới, không phụ thuộc vào số lượng classes.
- Siam network: Tránh vấn đề output shape thay đổi, tập trung vào việc tìm biểu diễn không gian.

III. Quy trình hoạt động chung cho hệ thống nhận dạng khuôn mặt

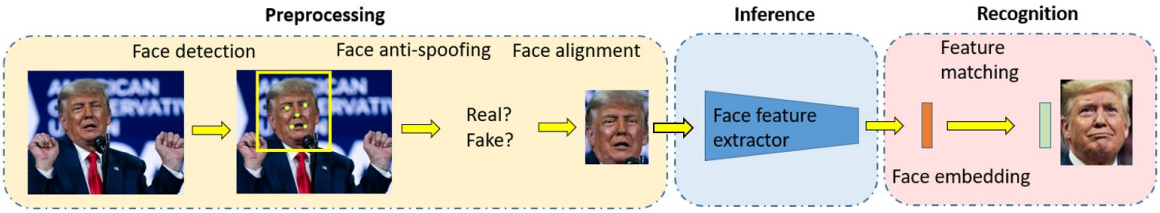
Quy trình hoạt động của một hệ thống nhận diện khuôn mặt thường bao gồm các bước sau:

- Phát hiện khuôn mặt: Xác định vị trí của các khuôn mặt và vẽ khung xung quanh chúng, sau đó lưu trữ tọa độ của khung xung quanh.
- Căn chỉnh khuôn mặt: Cắt các khuôn mặt ra và chuẩn hóa các khuôn mặt để thống nhất với cơ sở dữ liệu huấn luyện (kích thước phù hợp với bộ trích xuất đặc trưng).
- Trích xuất đặc trưng: Trích xuất các đặc trưng của khuôn mặt mà sẽ được sử dụng cho nhiệm vụ huấn luyện và nhận diện.
- Nhận diện khuôn mặt: So khớp khuôn mặt với một hoặc nhiều khuôn mặt đã biết trong cơ sở dữ liệu đã chuẩn bị.

Quy trình Huấn luyện:



Quy trình Chạy suy luận:

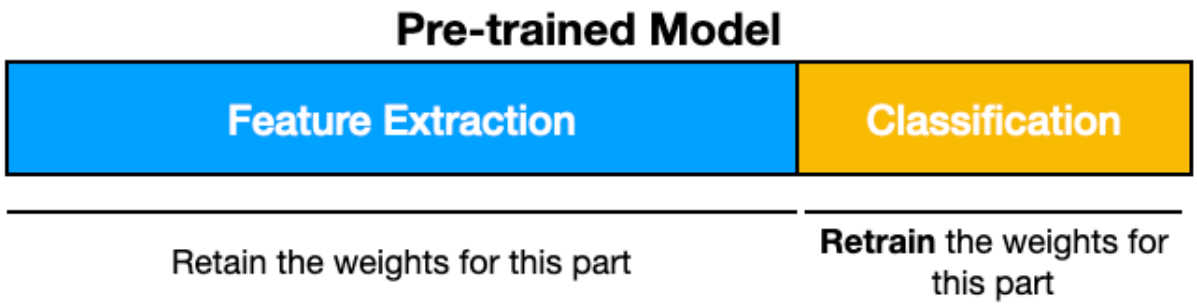


IV. Học chuyển giao (Transfer learning)

Các phương pháp học sâu có ưu điểm là độ chính xác cao, nhưng khuyết điểm đi kèm là đòi hỏi một lượng data để huấn luyện rất lớn, cũng như thời gian, công sức huấn luyện.

Transfer learning là việc sử dụng kiến thức, kỹ năng từ tác vụ nguồn (source task) sang tác vụ khác (target task) có liên quan nhằm cải thiện việc học hàm f cho tác vụ mục tiêu.

Do kiến trúc của các mô hình CNN gồm hai phần chính là Feature extractor và Classification. Phần feature extractor có thể được tận dụng lại cho những bài toán tương tự, ta chỉ cần fine-tuning một số layer cuối.



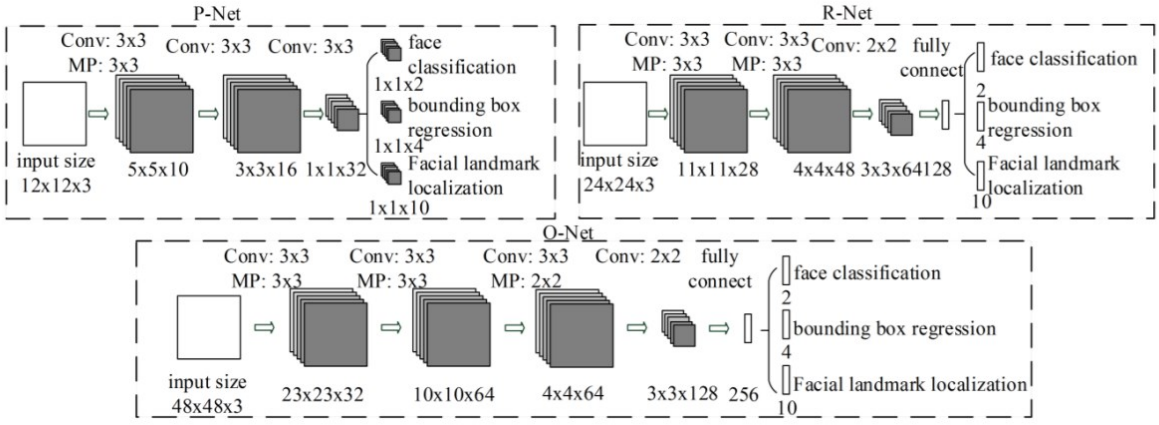
V. Bộ phận Detector

Phần đầu tiên của hệ thống là bộ phận xác định khuôn mặt, ta cần tìm bounding_box của khuôn mặt sau đó thực hiện align face.

1. Yolov8

YoloV8 là một trong những phiên bản mới nhất của mô hình YOLO (You Only Look Once), một trong những mô hình phổ biến trong lĩnh vực nhận diện đối tượng và định vị vật thể trong ảnh và video. Phiên bản YoloV8 nâng cấp từ các phiên bản trước đó, cải thiện độ chính xác và hiệu suất của việc nhận diện vật thể trong ảnh, đặc biệt là trong việc nhận diện khuôn mặt.

2. MTCNN



MTCNN là một mô hình nhận diện khuôn mặt phổ biến được sử dụng trong lĩnh vực thị giác máy tính. MTCNN sử dụng cấu trúc *Cascade* để phát hiện khuôn mặt ở các kích thước khác nhau. Mỗi giai đoạn của Cascade Network chịu trách nhiệm cho một công việc cụ thể, từ việc phát hiện khuôn mặt tổng quát tới việc xác định vị trí cụ thể và xác định các điểm landmarks.

MTCNN sử dụng cơ chế học đa nhiệm (*Multi-task learning*). MTCNN không chỉ tập trung vào việc phát hiện khuôn mặt mà còn thực hiện các nhiệm vụ phụ khác như

xác định vị trí bounding box và landmarks (điểm đặc trưng trên khuôn mặt). Quá trình này tối ưu hóa đồng thời nhiều nhiệm vụ thông qua một mô hình duy nhất, giúp tăng độ chính xác và hiệu suất.

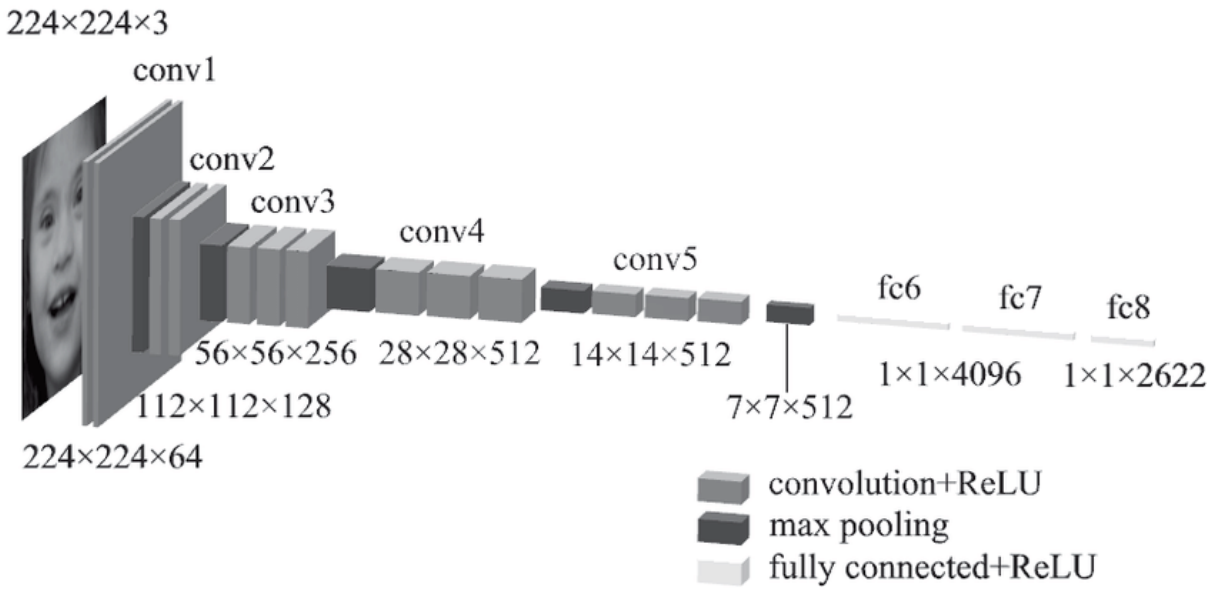
VI. Bộ phận Trích xuất đặc trưng

1. VggFace

VGG được tạo ra bởi các nhà khoa học trong nhóm Visual Geometry Group (VGG) , Oxford. Đến thời điểm hiện tại, nó có 2 phiên bản: VGGFace và VGGFace2.

VGGFace sử dụng dữ liệu lớn để huấn luyện một CNN theo kiến trúc của VGG, tạo ra các vectơ biểu diễn khuôn mặt và sử dụng Triplet Loss để tối ưu hóa các đặc trưng.

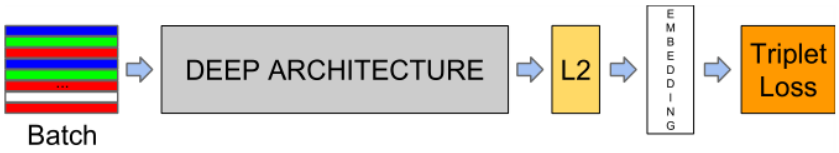
VGGFace2 cũng sử dụng dữ liệu lớn, thu thập từ Google Image Search, nhưng không sử dụng kiến trúc VGG mà thay vào đó là ResNet-50 hoặc SqueezeNet-ResNet-50. Cả hai đều đạt kết quả hàng đầu trong các bài kiểm tra chuẩn.



2. FaceNet

FaceNet là một trong những phương pháp nổi tiếng trong lĩnh vực nhận diện khuôn mặt dựa trên deep learning. Được giới thiệu bởi Google Research vào năm 2015.

- Baseline sử dụng mạng CNN và giảm số chiều của dữ liệu xuống chỉ còn 128 chiều, giúp tăng tốc độ suy luận, đồng thời vẫn đảm bảo độ chính xác.
- Sử dụng hàm loss là triplet loss function, có khả năng học đồng thời đặc điểm giống nhau giữa hai hình ảnh thuộc cùng một nhóm và phân biệt giữa các hình ảnh không cùng nhóm. Giúp mang lại hiệu quả đáng kể hơn so với các phương pháp trước đây.



Triplet Loss so sánh sự tương quan giữa các embeddings của ba hình ảnh:

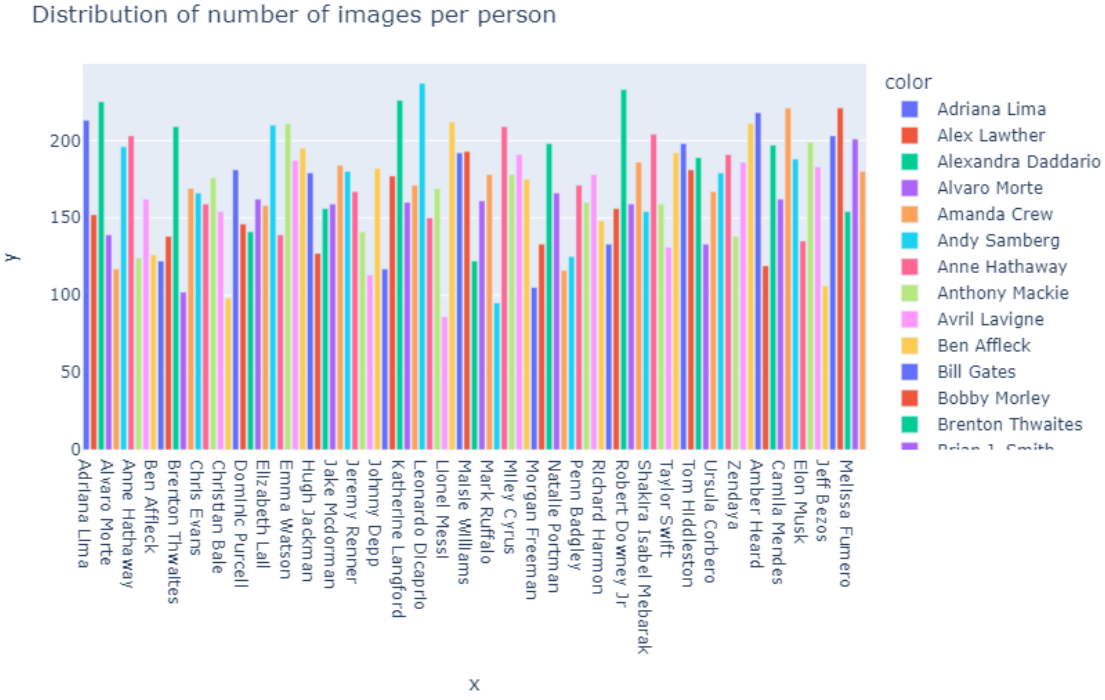
- *Anchor*: Là hình ảnh của một người nào đó.
- *Positive*: Là hình ảnh của cùng một người nhưng ở một góc chụp, ánh sáng hoặc biến thể khác nhau.
- *Negative*: Là hình ảnh của một người khác.

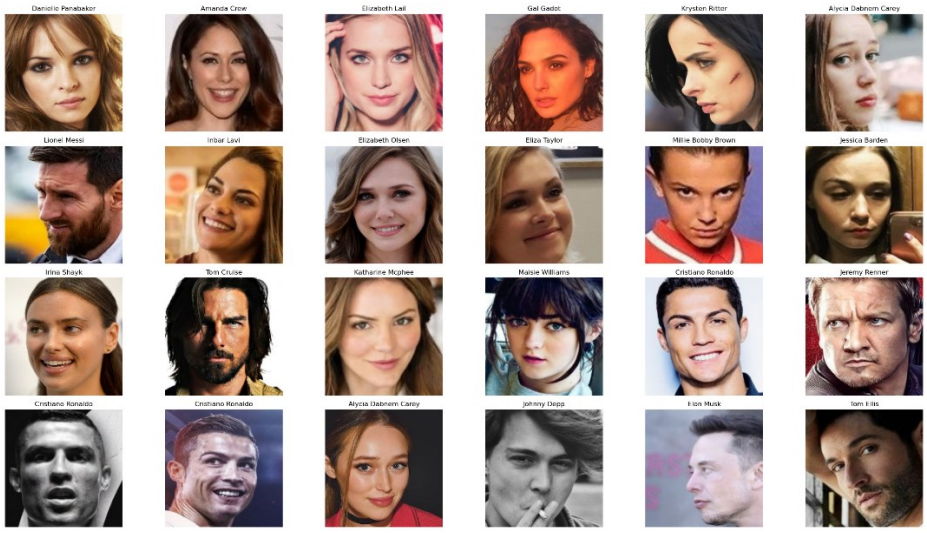
Triplet Loss tối ưu hóa khoảng cách giữa các vectơ embeddings của Anchor và Positive (càng gần nhau càng tốt) đồng thời tối ưu hóa khoảng cách giữa Anchor và Negative (càng xa nhau càng tốt). Mục tiêu là để đảm bảo rằng khoảng cách giữa hai embeddings của cùng một người (Anchor và Positive) nhỏ hơn so với khoảng cách giữa hai embeddings của các người khác nhau (Anchor và Negative).

$$\mathcal{L}(\mathbf{A},\mathbf{P},\mathbf{N}) = \sum_{i=0}^{n} \max(||f(\mathbf{A}_i) - f(\mathbf{P}_i)||_2^2 - ||f(\mathbf{A}_i) - f(\mathbf{N_i})||_2^2 + \alpha, 0)$$

C.TẬP DỮ LIỆU HUẤN LUYỆN

Em sử dụng bộ dữ liệu **Pins Face Recognition**. Bộ dữ liệu này bao gồm **17534** khuôn mặt của 105 người nổi tiếng. Trung bình có 167 ảnh cho mỗi người. Ta kiểm tra sự phân phối ảnh cho từng người và thấy phân phối này đáp ứng yêu cầu (không quá lệch):





Em phân chia tập dữ liệu thành hai phần train và test theo tỉ lệ **80%:20%.** Và chia riêng trong từng class để tránh ảnh của người này chỉ xuất hiện trong test mà `không có trong tập train.

D.MÔ TẢ PHƯƠNG PHÁP

I. YoloV8

Trong bài này em sử dụng pretrain `'yolov8n-face.pt'`, với thư viện `ultralytics`. Việc sử dụng rất đơn giản, chỉ cần gọi lệnh: `face = yolo(path, conf = conf_threshold)`.

II. MTCNN

Em vẫn dùng pretrain MTCNN. Tuy nhiên khi dùng MTCNN thì gặp nhiều khó khăn hơn khi dùng Yolov8 vì tốc độ thực thi khá **chậm** (so với Yolov8). Cách khắc phục là dùng MTCNN của thư viện `facenet_pytorch` (ban đầu em dùng thư viện khác). Một khó khăn khác là cài đặt của thư viện này trả về ảnh có các giá trị pixel kiểu int32, điều này gây **tràn RAM** và không đủ bộ nhớ, do đó em cần phải chuyển về kiểu uint8 cho đồng bộ với Yolo bên trên.

III. FaceNet

Dùng pretrain FaceNet của `keras_facenet` để chuyển aligned face sang embedding. FaceNet yêu cầu ảnh đầu vào có kích thước 160x160. Sau đó huấn luyện SVC như bộ phân lớp: `SVC(kernel='linear', probability=True)`

IV. VggFace

Dùng pretrain (có finetuning) VggFace của `keras_vggface`. Tuy nhiên em gặp một số lỗi (do phía thư viện) và đã tìm được cách khắc phục (trình bày trong source code).

VggFace yêu cầu ảnh đầu vào có kích thước 224x224. Model ban đầu có 26 lớp, trong đó 7 lớp cuối dùng để face recognition. Do đó ta không tải 7 layer cuối (chỉ tải 19 layers).

```
base_model = VGGFace(include_top=False,
    model='vgg16',
    input_shape=(224, 224, 3))
base_model.summary()
```

Sau đó thêm vào 6 layer và huấn luyện lại chúng (20 epoch):

```
# add the custom layers so that the model can recognize
the faces
x = base_model.output
x = GlobalAveragePooling2D()(x)

x = Dense(1024, activation='relu')(x)
```

```
x = Dense(1024, activation='relu')(x)
x = Dense(512, activation='relu')(x)

# final layer with softmax activation
preds = Dense(105, activation='softmax')(x)
```

```
Total params: 16,868,265
Trainable params: 2,153,577
Non-trainable params: 14,714,688
```

V. Realtime với webcam

Em dùng thư viện `face_recognition` để minh họa một hệ thống nhận diện khuôn mặt đơn giản, realtime với webcam. Thư viện này hỗ trợ các hàm cần thiết để làm hệ thống nhận dạng. Đầu tiên ta sẽ detect và encode các khuôn mặt trong database và lưu làm known_face. Sau đó khi mở camera, sẽ duyệt từng khung hình và detect khuôn mặt, so khớp với known_face xem trùng khớp hay không.

E. THỰC NGHIỆM VÀ DEMO

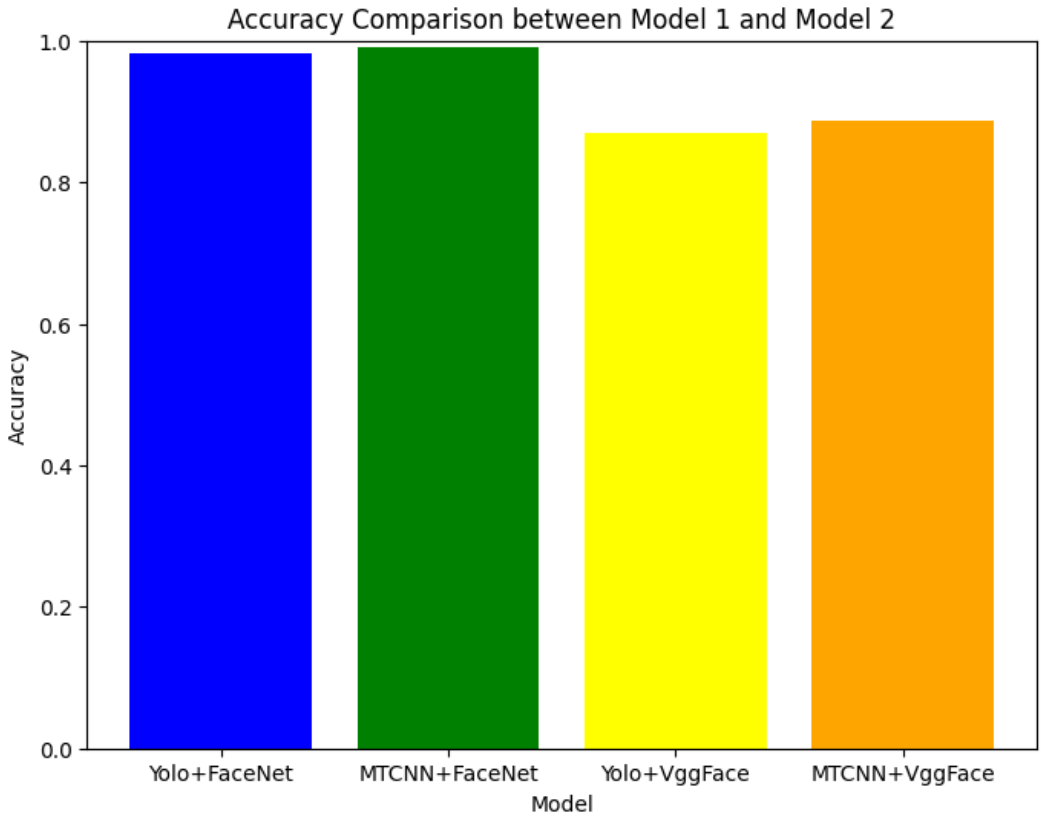
I. Độ đo

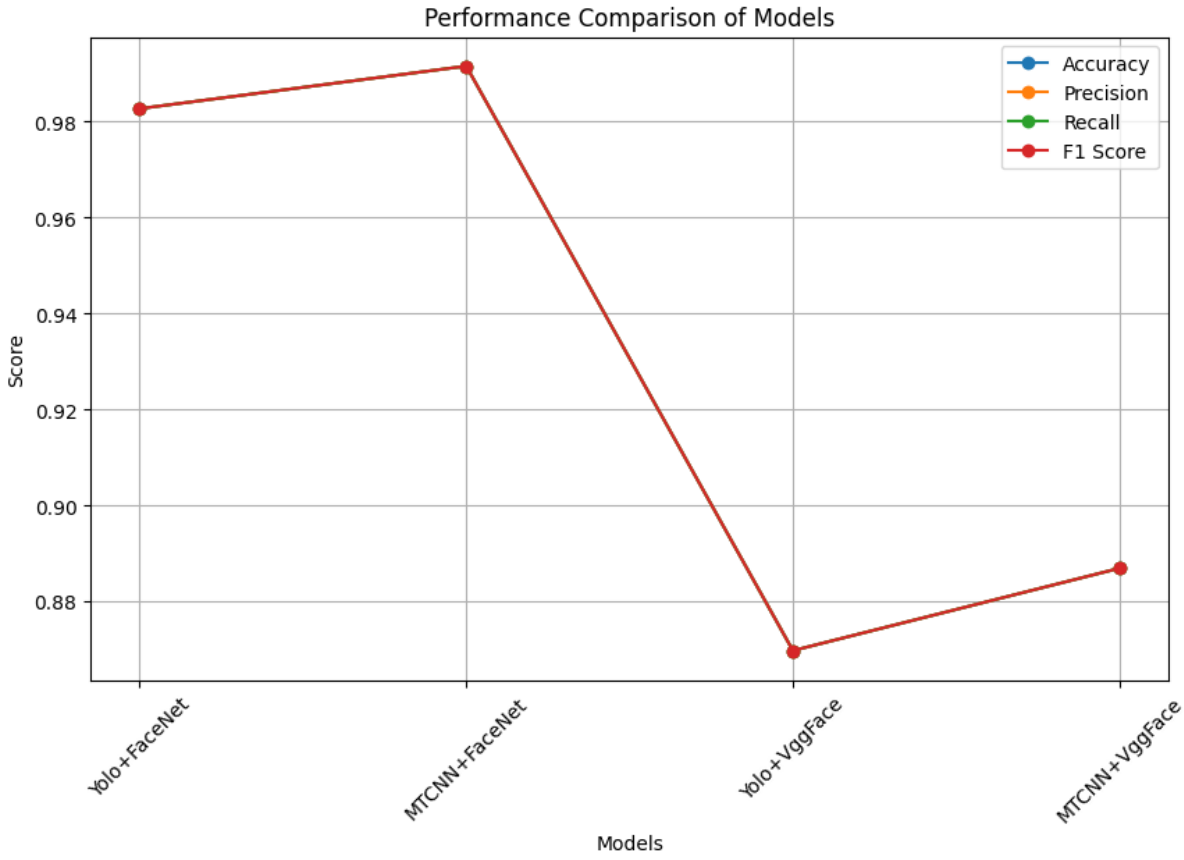
Đánh giá bằng các độ đo đơn giản: Accuracy, Precision, Recall, F1-score.

II. Kết quả thực nghiệm 4 mô hình

	Model	Accuracy	Precision	Recall	F1 Score
0	Yolo+FaceNet	0.982676	0.982676	0.982676	0.982676
1	MTCNN+FaceNet	**0.991540**	0.991540	0.991540	0.991540
2	Yolo+VggFace	0.869747	0.869747	0.869747	0.869747
3	MTCNN+VggFace	0.886915	0.886915	0.886915	0.886915

Đánh giá chung thì mô hình MTCNN+FaceNet cho kết quả rất tốt và cao hơn so với 3 mô hình còn lại. Ta có thể thấy FaceNet trích xuất đặt trưng tốt hơn so với VggFace. Ảnh hưởng của bộ detector lên kết quả thực nghiệm của hệ thống cũng đáng kể, cụ thể MTCNN cho kết quả tốt hơn so với Yolo (FaceNet: `0.982676 -> 0.982676`. VggFace: `0.869747 -> 0.886915`).





Về **thời gian chạy**, do gặp vài trục trặc nên em không thống kê được thời gian chạy chính xác. Tuy nhiên, em nhận xét thời gian chạy của MTCNN lâu hơn đáng kể so với Yolov8 (hơn khoảng 1,8 lần).

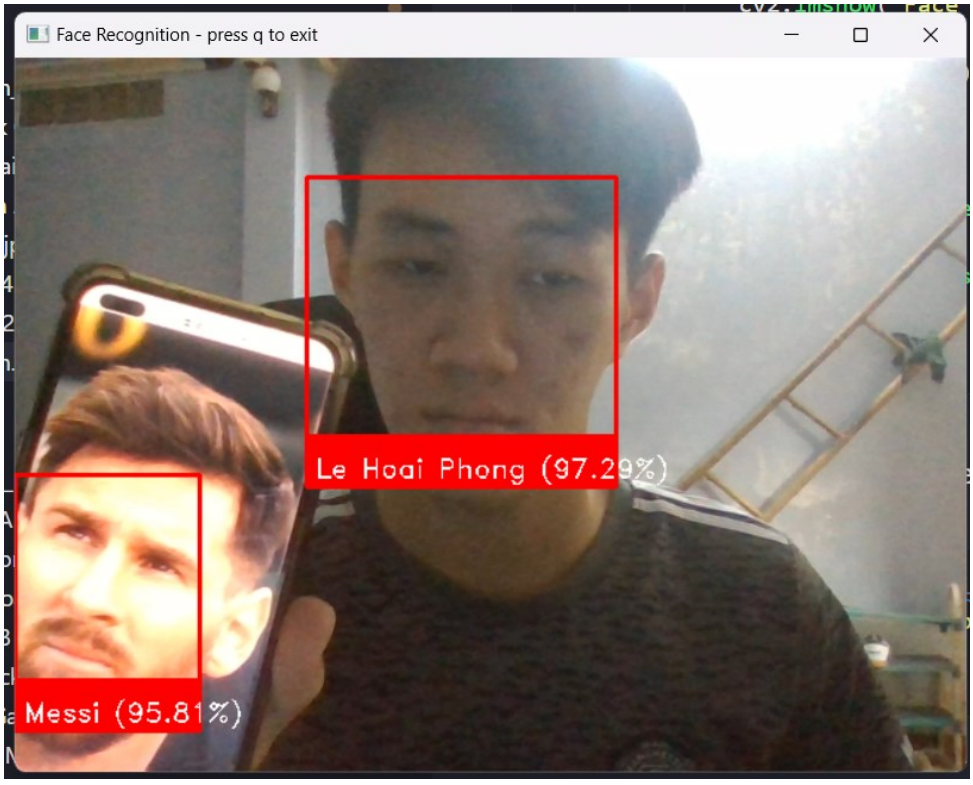
III. Hệ thống nhận diện realtime bằng webcam

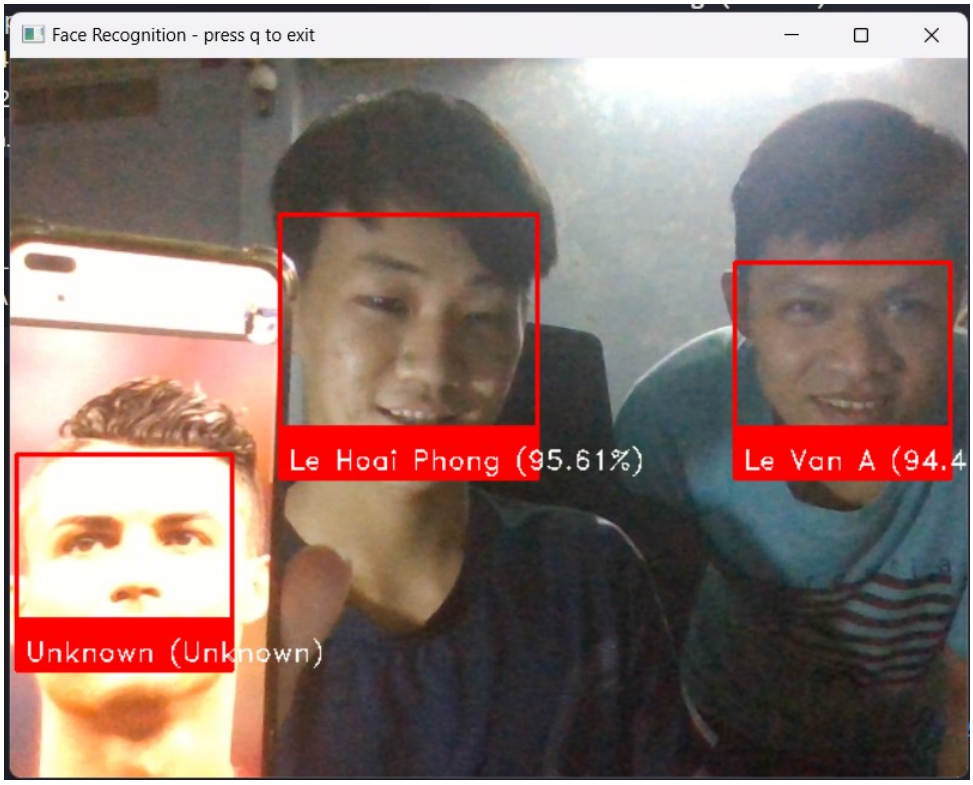
Cách sử dụng: Ta cần một database chứa ảnh của các người cần nhận dạng, mỗi ảnh được đặt tên là tên của người đó (chỉ cần 1 ảnh cho mỗi người).

- `Database:`
  - `Person 1.jpg`
  - `Person 2.jpg`
  - `…`
  - `Person n.jpg`

Mở file `webcam.ipynb` và chạy tất cả các cell, khi đó camera máy sẽ được mở. (Chỉ hoạt động trên local, không hoạt động được trên Colab do không truy cập được camera). Khi tắt thì nhấn phím ‘q’.

Kết quả demo như hình bên dưới (với mặt người không có trong dataset sẽ là unknown):


Face Recognition - press q to exit
Le Hoai Phong (97.29%)
Messi (95.81%)


Face Recognition - press q to exit
Le Hoai Phong (95.61%)
Le Van A (94.4
Unknown (Unknown)

F. TÀI LIỆU THAM KHẢO

[1] Schroff, Florian, Dmitry Kalenichenko, and James Philbin. "Facenet: A unified embedding for face recognition and clustering." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

[2] Wang, Xinyi, et al. "A Survey of Face Recognition." arXiv preprint arXiv:2212.13038 (2022).

[3] Xiang, Jia, and Gengming Zhu. "Joint face detection and facial expression recognition with MTCNN." *2017 4th international conference on information science and control engineering (ICISCE).* IEEE, 2017.

[4] Jiang, Peiyuan, et al. "A Review of Yolo algorithm developments." *Procedia Computer Science 199 (2022)*: 1066-1073.

[5] Jain, Anil K., and Stan Z. Li. *Handbook of face recognition*. Vol. 1. New York: springer, 2011.

[6] https://github.com/timesler/facenet-pytorch