

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN ĐIỆN TỬ - VIỆN THÔNG



BÁO CÁO ĐỒ ÁN 3

Đề tài: Xây dựng hệ thống điểm danh qua camera

Giảng viên hướng dẫn: cô Trần Thị Ngọc Lan

Sinh viên thực hiện: Nguyễn Quý Hoàng

Hà Nội, 3-2023

Lời nói đầu

Hệ thống điểm danh qua camera là một đề tài ứng dụng công nghệ nhận diện khuôn mặt và hình ảnh để đánh giá chấm công và điểm danh. Đây là một ứng dụng của lĩnh vực Computer Vision và Machine Learning.

Hệ thống này sử dụng các kỹ thuật nhận diện khuôn mặt, xác thực và định danh người dùng từ hình ảnh được chụp bởi camera. Hệ thống sử dụng mô hình Machine Learning để học và nhận diện các đặc trưng của khuôn mặt, sau đó so sánh với dữ liệu đã lưu trữ để xác định danh tính của người đó.

Hệ thống điểm danh qua camera có nhiều ứng dụng, chẳng hạn như trong các công ty, trường học, các tổ chức cộng đồng, nơi cần kiểm soát lưu lượng người ra vào và thực hiện chấm công cho nhân viên. Việc sử dụng hệ thống này giúp tiết kiệm thời gian và giảm thiểu những sai sót xảy ra trong quá trình chấm công truyền thống.

Tuy nhiên, hệ thống này cũng đặt ra nhiều thách thức, chẳng hạn như đảm bảo tính chính xác và bảo mật của dữ liệu, đồng thời phải đối mặt với các vấn đề về ánh sáng, góc chụp và độ phân giải của camera. Do đó, để phát triển một hệ thống điểm danh qua camera hiệu quả và đáng tin cậy, cần phải kết hợp nhiều kỹ thuật và công nghệ khác nhau.

Em xin gửi lời cảm ơn chân thành đến cô Trần Thị Ngọc Lan - giảng viên hướng dẫn của em trong suốt quá trình làm đồ án. Cô đã giúp đỡ và hỗ trợ em rất nhiều trong quá trình nghiên cứu và hoàn thiện đề tài này.

Contents

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI	5
1.1 Giới thiệu đề tài.....	5
1.2 Nội dung đề tài	6
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	6
2.1 Neural Network.....	6
2.2 Mạng Neural tích chập	8
2.2.1 Lớp tích chập.....	10
2.2.2 Pooling Layer	13
2.2.3 Fully Connected Layer	13
2.2.4 Ứng dụng của CNN trong các bài toán Computer Vision.....	14
2.3 Bài toán Nhận diện khuôn mặt.....	17
2.3.1 Phương pháp tiếp cận bài toán Bài toán	17
2.3.2 Hệ thống nhận diện khuôn mặt	18
2.4 Face Detection	19
2.4.1 Non-maximum Suppression.....	20
2.4.2 MTCNN.....	20
2.4.3 Sample and Computation Redistribution for Face Detection	24
2.5 Face Alignment.....	24
2.6 Feature Extraction	25
2.7 Feature Matching.....	25
2.8 Loss Function.....	25
2.8.1 Softmax Loss.....	26
2.8.2 Triplet Loss	26
2.9 Kiến trúc Resnet.....	29
CHƯƠNG 3. MÔ HÌNH.....	31
3.1 Kiến trúc	31
3.1.1 Backbone.....	31
3.1.2 Head.....	35

CHƯƠNG 4. XÂY DỰNG HỆ THỐNG NHẬN DIỆN	35
4.1 Hệ thống	36
4.1.1 Face Detection	36
4.1.2 Face Alignment	37
4.1.3 Feature Extraction	37
4.1.4 Feature Matching	37
4.2 Thử nghiệm	39
4.3 Điểm danh với webcam	40

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI

1.1 Giới thiệu đề tài

Ngày nay, với sự tiến bộ vượt bậc của công nghệ thông tin, ngày càng có nhiều những ứng dụng của công nghệ thông tin được áp dụng trong đời sống thường ngày. Trong những năm gần đây, một trong những bài toán nhận được nhiều sự quan tâm nhất của xã hội, có tính ứng dụng cao vào đời sống, đó chính là bài toán Nhận diện khuôn mặt – Face Recognition. Nhận diện khuôn mặt được ứng dụng trong điện thoại thông minh mở khóa bằng khuôn mặt, các hệ thống xác thực người dùng, điểm danh nhân viên trong công ty, các hệ thống giám sát an ninh.

Khác với những phương pháp khác trong hệ thống nhận dạng người bằng sinh trắc học như vân tay, móng mắt, nhận dạng qua khuôn mặt có thể thực hiện một cách đơn giản, tiết kiệm thời gian. Điều này đặc biệt có lợi cho các hệ thống nhận diện điểm danh hoặc với mục đích an ninh, giám sát.

Hệ thống điểm danh sử dụng camera là một ứng dụng của công nghệ nhận diện khuôn mặt. Hệ thống này sẽ sử dụng camera để chụp ảnh và sau đó áp dụng các thuật toán nhận diện khuôn mặt để xác định danh tính của người đó.

Trong hệ thống điểm danh sử dụng camera, chúng ta sẽ sử dụng các thuật toán nhận diện khuôn mặt như face detection để phát hiện khuôn mặt trong ảnh và face recognition để nhận dạng danh tính của người đó. Sau khi nhận dạng được danh tính, hệ thống sẽ ghi lại thời gian điểm danh của người đó và lưu lại vào cơ sở dữ liệu.

Hệ thống điểm danh sử dụng camera sẽ giúp cho việc điểm danh trở nên nhanh chóng và chính xác hơn, đồng thời giảm thiểu những sai sót khi điểm danh bằng phương pháp thủ công. Nó cũng có thể được sử dụng trong nhiều lĩnh vực khác nhau như trong các công ty, trường học, bệnh viện, siêu thị, các tổ chức tổ chức sự kiện,..vv.

Ngày nay, các phương pháp hiện đại đã có thể đạt được độ chính xác hơn 98% trên những tập dữ liệu chất lượng cao (LFW, CFP-FP) [6],[7]. Tuy nhiên, trong các hệ thống giám sát, điều tra an ninh, quân sự... ảnh thu được từ camera hay máy bay không người lái đa phần là những ảnh nhiễu, mờ, chất lượng không tốt thì các hệ thống rất khó để nhận diện một cách chính xác. Khi chất lượng hình ảnh thấp thì các đặc trưng của khuôn mặt sẽ bị mờ đi làm cho khả năng nhận diện của mô hình sẽ thiếu chính xác hoặc cũng có thể không nhận diện được. Hình ảnh

sau đây cho thấy các ví dụ về khả năng nhận diện của hình ảnh chất lượng cao và chất lượng thấp.

Recognizability Image Quality	Easy to Recognize	Hard to Recognize	Impossible to Recognize
High Quality			
Low Quality			

 : Images contain enough clues to identify the subject

 : Images **do not** have enough clues to identify the subject

Hình 1.1: Khả năng nhận diện với chất lượng hình ảnh

1.2 Nội dung đề tài

Mục tiêu chính của đề tài là nghiên cứu, phát triển bài toán Nhận diện khuôn mặt với chất lượng ảnh mờ với khuôn mặt của người Việt Nam và xây dựng 1 hệ thống nhận diện khuôn mặt.

Mục tiêu khi thực hiện đồ án này bao gồm:

- Nghiên cứu mô hình Nhận diện khuôn mặt AdaFace.
- Xây dựng, phát triển một hệ thống nhận diện khuôn mặt và thử nghiệm

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

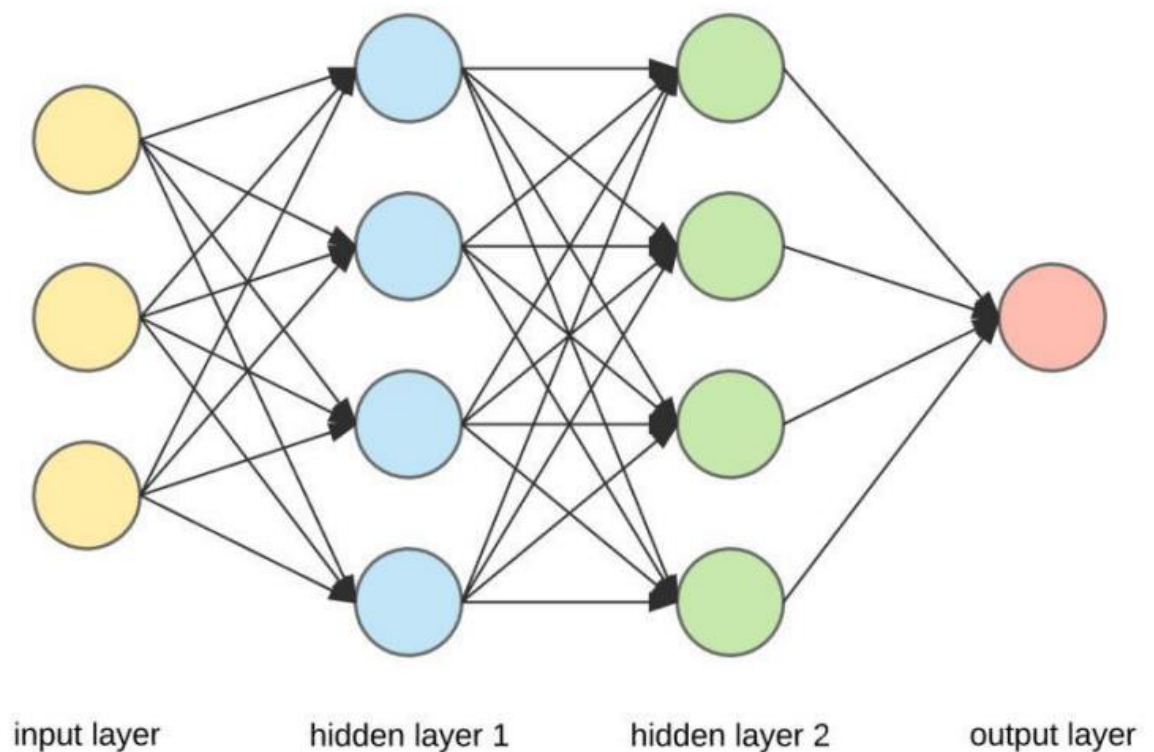
2.1 Neural Network

Neural Network hay còn được gọi là mạng Neural nhân tạo là một chuỗi những thuật toán được đưa ra để tìm kiếm các mối quan hệ cơ bản trong tập hợp các dữ liệu. Thông qua việc bắt chước cách thức hoạt động từ não bộ con người. Nói cách khác, mạng neural nhân tạo được xem là hệ thống của các tế bào thần kinh nhân tạo.

Mạng neural được ứng dụng rất nhiều để giải quyết các bài toán trong thực tế như phân loại (ảnh, giọng nói, tín hiệu), dự báo, nhận dạng hệ thống và thiết kế bộ điều khiển.

Mạng Neural nhân tạo là sự kết hợp của những tầng perceptron hay còn gọi là perceptron đa tầng. Và mỗi một mạng Neural nhân tạo thường bao gồm 3 kiểu tầng là:

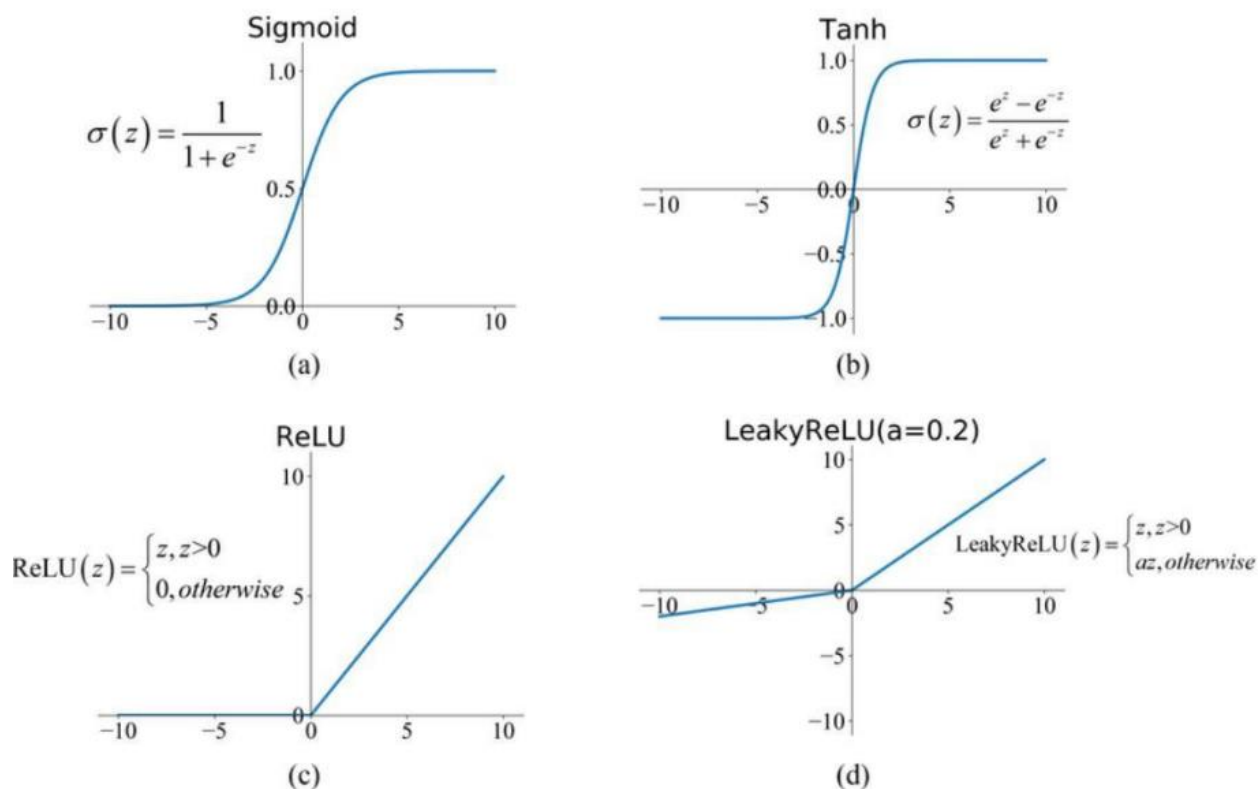
- Input layer (tầng vào): Tầng này nằm bên trái cùng của mạng, thể hiện cho các đầu vào của mạng.
- Output layer (tầng ra): Là tầng bên phải cùng và nó thể hiện cho những đầu ra của mạng.
- Hidden layer (tầng ẩn): Tầng này nằm giữa tầng vào và tầng ra nó thể hiện cho quá trình suy luận logic của mạng.



Hình 2.1: Kiến trúc mạng Neural Network

Mỗi một mạng Neural nhân tạo chỉ có một tầng input layer, một tầng output layer, tuy nhiên có thể có nhiều tầng hidden layer.

Các mô hình Neural Network có khả năng giải quyết được những vấn đề về tính chất phi tuyến của dữ liệu. Tầng hidden layer có nhiệm vụ giải quyết những quan hệ phi tuyến phức tạp giữa các đặc điểm của dữ liệu và kết quả đầu ra của mô hình nhờ những hàm "phi tuyến hóa" thường được biến đến với tên "activation functions" hay là hàm kích hoạt.



Hình 2.2: Một vài hàm kích hoạt thường dùng

2.2 Mạng Neural tích chập

Convolutional Neural Network (CNN – Mạng neural tích chập) là một trong những mô hình Deep Learning tiên tiến, giải quyết được nhiều bài toán khác nhau trong nhiều lĩnh vực như xử lý ảnh, xử lý tín hiệu, giọng nói. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao.

CNN cũng có lịch sử khá lâu đời. Kiến trúc gốc của mô hình CNN được giới thiệu bởi một nhà khoa học máy tính người Nhật vào năm 1980. Sau đó, năm 1998, Yan LeCun lần đầu huấn luyện mô hình CNN với thuật toán backpropagation cho bài toán nhận dạng chữ viết tay. Tuy nhiên, mãi đến năm 2012, khi một nhà khoa học máy tính người Ukraine Alex Krizhevsky xây dựng

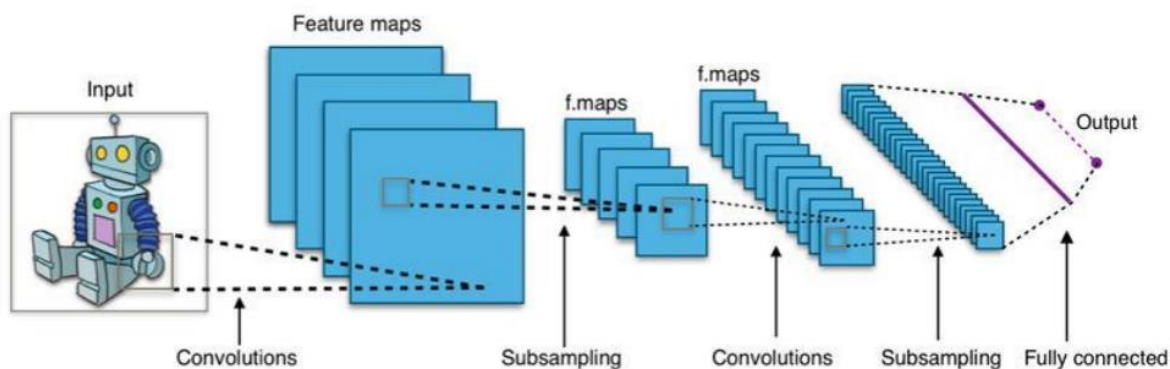
mô hình CNN (AlexNet) và sử dụng GPU để tăng tốc quá trình huấn luyện deep nets để đạt được top 1 trong cuộc thi Computer Vision thường niên ImageNet với độ lỗi phân lớp top 5 giảm hơn 10% so với những mô hình truyền thống trước đó, đã tạo nên làn sóng mãnh mẽ sử dụng deep CNN với sự hỗ trợ của GPU để giải quyết càng nhiều các vấn đề trong Computer Vision.

Mạng CNN là một tập hợp các lớp Convolution chồng lên nhau và sử dụng các hàm nonlinear activation như ReLU và tanh để kích hoạt các trọng số trong các node. Mỗi một lớp sau khi thông qua các hàm kích hoạt sẽ tạo ra các thông tin trừu tượng hơn cho các lớp tiếp theo. Trong mô hình mạng truyền ngược (feedforward neural network) thì mỗi neural đầu vào (input node) cho mỗi neural đầu ra trong các lớp tiếp theo. Mô hình này gọi là mạng kết nối đầy đủ (fully connected layer) hay mạng toàn vẹn (affine layer). Còn trong mô hình CNN thì ngược lại. Các layer liên kết được với nhau thông qua cơ chế convolution.

Layer tiếp theo là kết quả convolution từ layer trước đó, nhờ vậy mà ta có được các kết nối cục bộ. Như vậy mỗi neuron ở lớp kế tiếp sinh ra từ kết quả của filter áp đặt lên một vùng ảnh cục bộ của neuron trước đó.

Mỗi một lớp được sử dụng các filter khác nhau thông thường có hàng trăm hàng nghìn filter như vậy và kết hợp kết quả của chúng lại. Ngoài ra có một số layer khác như pooling/subsampling layer dùng để chắt lọc lại các thông tin hữu ích hơn (loại bỏ các thông tin nhiễu).

Trong quá trình huấn luyện mạng (training) CNN tự động học các giá trị qua các lớp filter dựa vào cách thức mà bạn thực hiện. Ví dụ trong tác vụ phân lớp ảnh, CNNs sẽ cố gắng tìm ra thông số tối ưu cho các filter tương ứng theo thứ tự raw pixel > edges > shapes > facial > high-level features. Layer cuối cùng được dùng để phân lớp ảnh.



Hình 2.3: Kiến trúc của mạng CNN đơn giản

Trong mô hình CNN có 2 khía cạnh cần quan tâm là tính bất biến (Location Invariance) và tính kết hợp (Compositionality). Với cùng một đối tượng, nếu đối tượng này được chiếu theo các góc độ khác nhau (translation, rotation, scaling) thì độ chính xác của thuật toán sẽ bị ảnh hưởng đáng kể. Pooling layer sẽ cho bạn tính bất biến đối với phép dịch chuyển (translation), phép quay (rotation) và phép co giãn (scaling). Tính kết hợp cục bộ cho ta các cấp độ biểu diễn thông tin từ mức độ thấp đến mức độ cao và trừu tượng hơn thông qua convolution từ các filter.

2.2.1 Lớp tích chập

Lớp tích chập (Convolutional Layer) sử dụng các bộ lọc để thực hiện phép tích chập khi đưa chúng qua đầu vào theo các chiều của nó. Khi nhắc đến lớp Convolutional Layer, chúng ta cần làm rõ một số khái niệm đó là: Stride, Padding, Feature Map. Các tham số của các bộ lọc này gồm kích thước bộ lọc F và độ trượt (stride) S . Kết quả đầu ra O được gọi là feature map hay activation map.

- 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.

Các convolutional layer có các parameter(kernel) đã được học để tự điều chỉnh lấy ra những thông tin chính xác nhất mà không cần chọn các feature.

Trong hình ảnh ví dụ dưới, ma trận bên trái là một hình ảnh trắng đen được số hóa. Ma trận có kích thước 5×5 và mỗi điểm ảnh có giá trị 1 hoặc 0 là giao điểm của dòng và cột. Convolution hay tích chập là nhân từng phần tử trong ma trận 3. Sliding Window hay còn gọi là kernel, filter hoặc feature detect là một ma trận có kích thước nhỏ như trong ví dụ trên là 3×3 . Convolution hay tích chập là nhân từng phần tử bên trong ma trận 3×3 với ma trận bên trái. Kết quả được một ma trận gọi là Convolved feature được sinh ra từ việc nhân ma trận Filter với ma trận ảnh 5×5 bên trái.

1 _{x1}	1 _{x0}	1 _{x1}	0	0
0 _{x0}	1 _{x1}	1 _{x0}	1	0
0 _{x1}	0 _{x0}	1 _{x1}	1	1
0	0	1	1	0
0	1	1	0	0



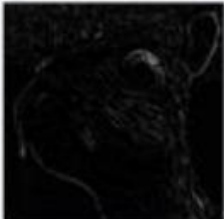
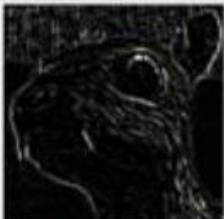


Image

4		

Convolved
Feature

Hình 2.4: Tích chập của ảnh và Filter

Mục đích của phép tính convolution trên ảnh là làm mờ, làm nét ảnh; xác định các đường;... Mỗi kernel khác nhau thì sẽ phép tính convolution sẽ có ý nghĩa khác nhau.

Operation	Kernel ω	Image result $g(x,y)$
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

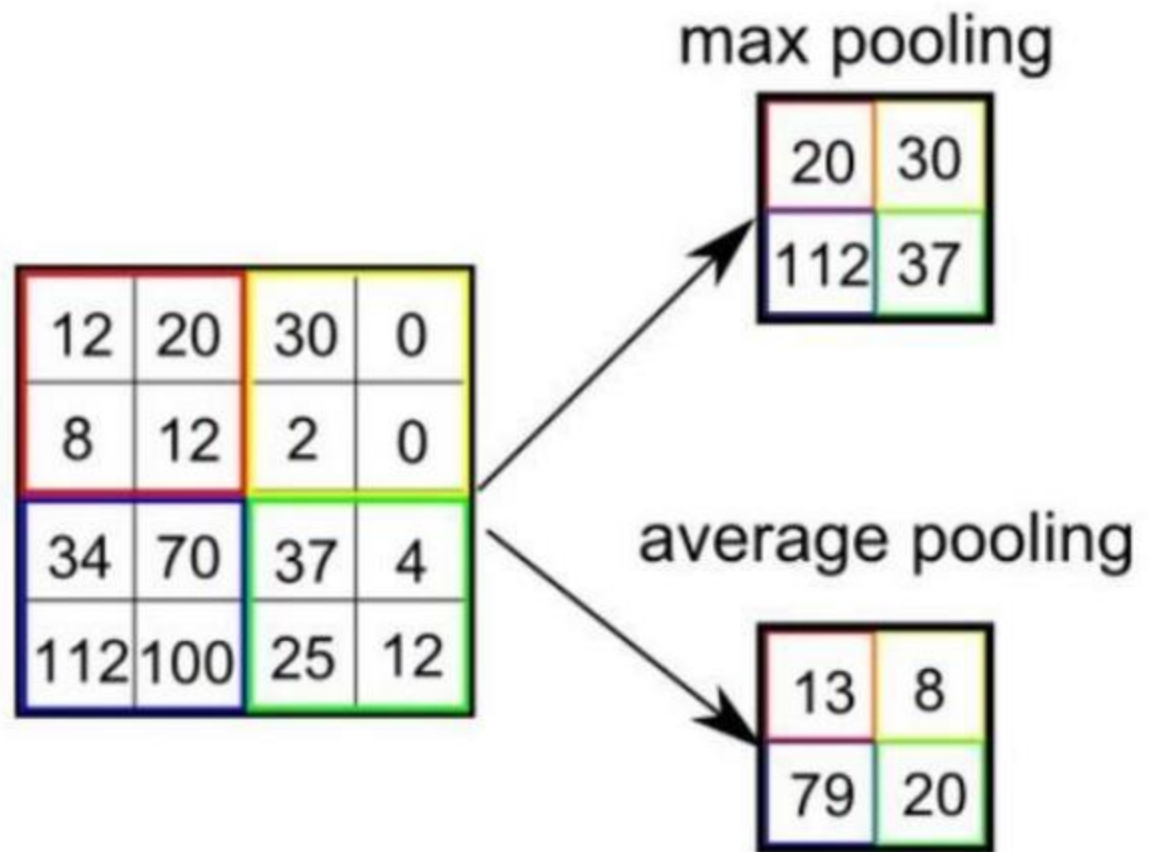
Hình 2.5: Một số kernel hay sử dụng và ứng dụng

2.2.2 Pooling Layer

Pooling Layer có tác dụng giảm kích thước của dữ liệu hình ảnh từ đó giúp cho mạng có thể học được các thông tin có tính chất khái quát hơn, đồng thời quá trình này giảm số lượng các thông số trong mạng.

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 2.6: Max Pooling và Average Pooling

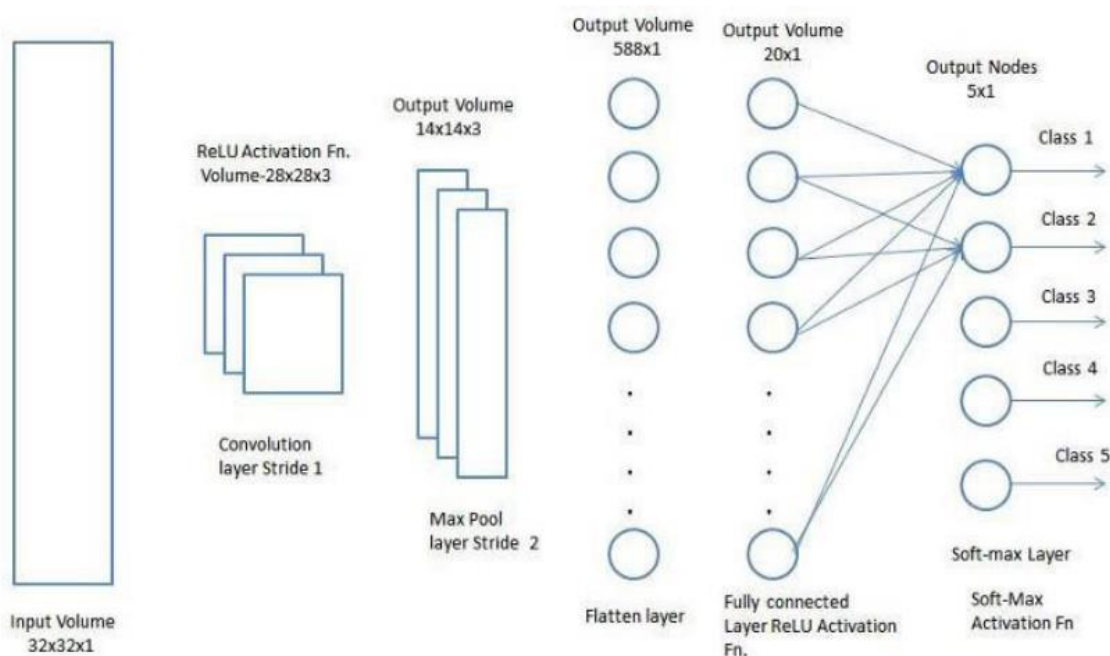
2.2.3 Fully Connected Layer

Tầng cuối cùng của mô hình CNN trong bài toán phân loại ảnh là tầng fully connected layer. Tầng này có chức năng chuyển ma trận đặc trưng ở tầng trước thành vector chứa xác suất của các đối tượng cần được dự đoán. Ví dụ, trong bài

toán phân loại số viết tay MNIST có 10 lớp tương ứng 10 số từ 0-1, tầng fully connected layer sẽ chuyển ma trận đặc trưng của tầng trước thành vector có 10 chiều thể hiện xác suất của 10 lớp tương ứng.

Sử dụng mạng neural kết nối đầy đủ là cách làm phổ biến nhất để học các tổ hợp phi tuyến từ các đặc trưng trích xuất từ kết quả ma trận tích chập đầu ra. Mạng neural kết nối đầy đủ có thể học được các đặc trưng trong không gian phi tuyến này.

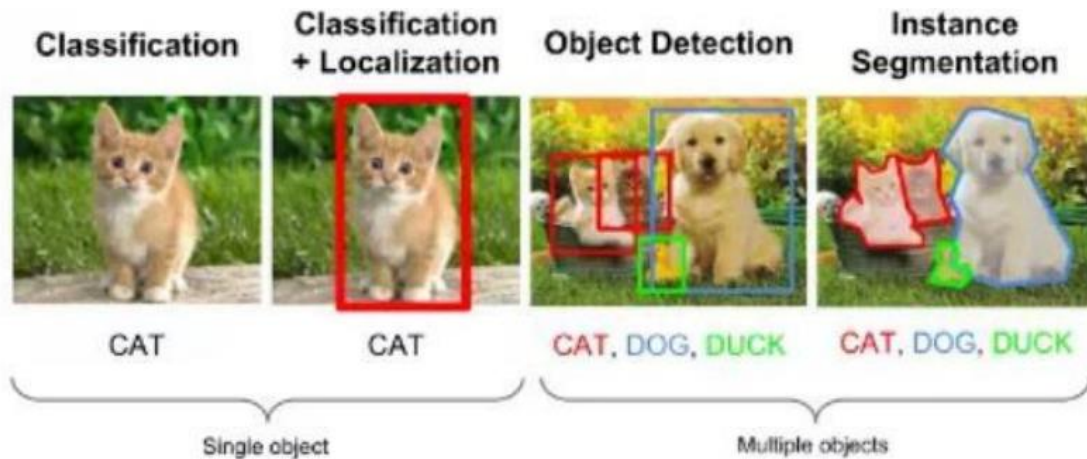
Chúng ta đã chuyển đổi hình ảnh đầu vào thành một dạng thích hợp cho mạng neural đa lớp. Chúng ta sẽ làm phẳng hình ảnh đầu vào này thành một vector cột. Vector đầu ra đã được làm phẳng sẽ được đưa vào một mạng neural suy luận tiến (feedforward) và phương pháp truyền ngược (backpropagation) được áp dụng cho quá trình huấn luyện. Qua một loạt lần lặp, mô hình có thể phân biệt giữa các đặc trưng cốt lõi và các đặc trưng không quan trọng trong hình ảnh và phân loại chúng bằng kỹ thuật Phân loại Softmax (softmax classification).



Hình 2.7: Quá trình đào tạo, suy luận của CNN

2.2.4 Ứng dụng của CNN trong các bài toán Computer Vision

CNN chủ yếu được sử dụng cho các vấn đề về computer vision như Classification, Localization, Object Detection, Segmentation...



Hình 2.8: Ứng dụng của mạng CNN

a) Classification

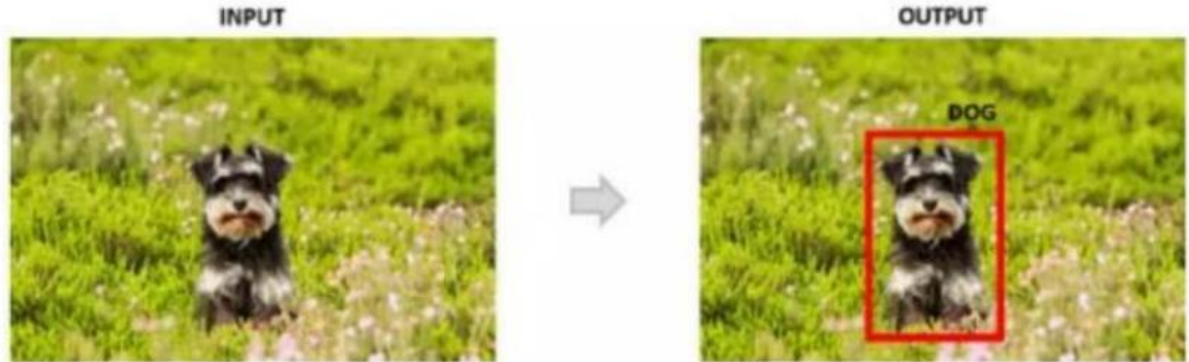
Đây là nhiệm vụ được biết đến nhiều nhất trong computer vision. Ý tưởng chính là phân loại nội dung chung của hình ảnh thành một tập hợp các danh mục, được gọi là nhãn. Ví dụ: phân loại có thể xác định xem một hình ảnh có phải là của một con chó, một con mèo hay bất kỳ động vật khác. Việc phân loại này được thực hiện bằng cách xuất ra xác suất của hình ảnh thuộc từng lớp, như được thấy trong hình ảnh sau:



Hình 2.9: Classification

b) Localization:

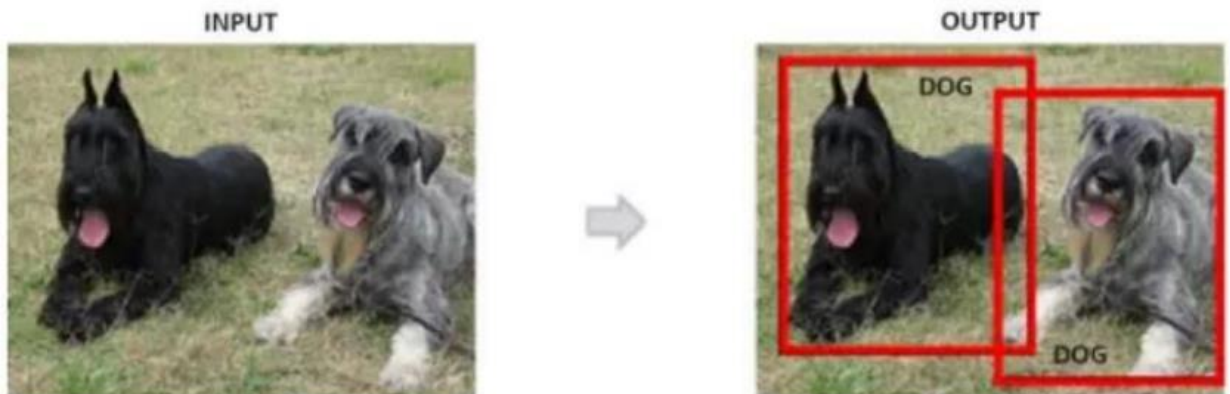
Mục đích chính của localization là tạo ra một hộp giới hạn mô tả vị trí của đối tượng trong hình ảnh hay còn gọi là bounding box. Đầu ra bao gồm một nhãn lớp và một hộp giới hạn. Tác vụ này có thể được sử dụng trong cảm biến để xác định xem một đối tượng ở bên trái hay bên phải của màn hình:



Hình 2.10: Localization

c) Detection:

Nhiệm vụ này bao gồm thực hiện localization trên tất cả các đối tượng trong ảnh. Các đầu ra bao gồm nhiều hộp giới hạn, cũng như nhiều nhãn lớp:



Hình 2.11: Detection

d) Segmentation:

Nhiệm vụ ở đây là xuất ra cả nhãn lớp và đường viền của mỗi đối tượng hiện diện trong hình ảnh. Điều này chủ yếu được sử dụng để đánh dấu các đối tượng quan trọng của hình ảnh cho phân tích sâu hơn. Ví dụ: tác vụ này có thể được sử dụng để phân định rõ ràng khu vực tương ứng với khối u trong hình ảnh phổi của bệnh nhân. Hình sau mô tả cách vật thể quan tâm được phác thảo và gán nhãn:



Hình 2.12: Segmentation

2.3 Bài toán Nhận diện khuôn mặt

Nhận diện khuôn mặt có thể nói bao gồm hai bài toán con:

- Xác nhận cá nhân (Face identification): là bài toán one-to-many.

Input là ảnh một khuôn mặt, và mô hình sẽ so sánh với những khuôn mặt đã có trong database và trả ra danh tính của người trong ảnh nếu người đó có trong database và ngược lại trả ra “Unknow” nếu người đó không có trong database.

- Xác thực khuôn mặt (Face verification, authentication): là bài toán one-to-one, được ứng dụng nhiều trong các hệ thống nhận dạng trên điện thoại thông minh. Input vẫn là ảnh khuôn mặt. Mô hình sẽ so sánh ảnh đó với ảnh khuôn mặt đã có trong database và trả ra kết quả xem 2 ảnh này có cùng 1 người hay không rồi từ đây chấp nhận hoặc từ chối nhận dạng.

2.3.1 Phương pháp tiếp cận bài toán Bài toán

Nhận diện khuôn mặt có 2 phương pháp tiếp cận:

a) One-shot learning

Là phương pháp phân loại các đối tượng chỉ sử dụng 1 hoặc 1 vài mẫu và sử dụng kiến trúc CNN đơn giản để dự đoán người đó. Về cơ bản nó bài toán classification.

Nhược điểm của phương pháp này là chỉ cần xuất hiện thêm người mới, chúng ta phải huấn luyện lại toàn bộ mô hình (shape của output thay đổi tăng lên). Đối với các bài toán chấm công hay camera an ninh khu chung cư, điều này gây bất lợi rất lớn, vì số lượng người luôn biến động theo thời gian.

Để khắc phục được vấn đề này, chúng ta sử dụng phương pháp learning similarity.

b) Learning Similarity

Giống như ý tưởng của các thuật toán phân cụm, hiểu một cách đơn giản, hai ảnh càng giống nhau (cùng 1 người), thì khoảng cách giữa chúng càng bé, và ngược lại, hai ảnh càng xa nhau thì khoảng cách càng phải lớn. Khoảng cách ở đây có thể là Norm chuẩn L1 hoặc L2 (euclidean distance).

$$\begin{cases} d(\text{img1}, \text{img2}) \leq \tau \rightarrow \text{sameclass} \\ d(\text{img1}, \text{img2}) > \tau \rightarrow \text{differentclass} \end{cases}$$

Thay vì dự báo một phân phối xác suất để tìm ra nhãn phù hợp nhất với ảnh đầu vào. Thuật toán sẽ so sánh khoảng cách giữa ảnh đầu vào với lần lượt toàn bộ các ảnh còn lại. Ta cần chọn một ngưỡng threshold để quyết định ảnh là giống hoặc khác.

Learning similarity có thể trả ra nhiều hơn một ảnh là cùng loại với ảnh đầu vào tùy theo ngưỡng threshold. Ngoài ra phương pháp này không bị phụ thuộc vào số lượng classes. Do đó không cần phải huấn luyện lại khi xuất hiện class mới.

Như vậy learning similarity có ưu điểm hơn so với one-shot learning khi không phải huấn luyện lại model khi mà vẫn tìm ra được ảnh tương đồng.

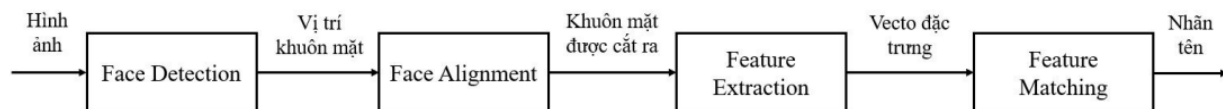
Để có thể sử dụng phương pháp learning similarity, chúng ta cần sử dụng 1 mạng để có thể biểu diễn thông tin của ảnh mà không lo vấn đề số class đầu ra thay đổi, chúng ta sẽ sử dụng Siam Network.

Siam network được xây dựng dựa trên base network là một CNN architecture (Resnet, Inception, InceptionResnet, MobileNet,...) đã được loại bỏ output layer, có tác dụng encoding ảnh thành vecto embedding.

Như vậy, vấn đề còn lại là xây dựng được một mô hình có thể biểu thị 1 bức ảnh về 1 vecto (trong bài sẽ sử dụng vecto 512 chiều), vừa đủ nhỏ để tính toán lại vừa biểu diễn đủ thông tin, sau đó sử dụng khoảng cách giữa các vecto hoặc sử dụng hàm cosine để quyết định nhãn của chúng.

2.3.2 Hệ thống nhận diện khuôn mặt

Hệ thống nhận diện khuôn mặt hoàn chỉnh được chia làm 4 khối như dưới đây:



Hình 2.13: Hệ thống Nhận diện khuôn mặt

- Face Detection: phát hiện vị trí khuôn mặt trong ảnh
- Face Alignment: cắt khuôn mặt ra và thực hiện căn chỉnh.
- Feature Extraction: biểu diễn khuôn mặt thành vecto feature
- Feature Matching: So sánh các feature với database và trả ra tên người đó.

2.4 Face Detection

Face Detection là giai đoạn đầu tiên trong hệ thống nhận dạng khuôn mặt và được coi là một phần quan trọng của hệ thống nhận diện khuôn mặt. Face Detection là một thách thức bởi tính chất thay đổi liên tục của khuôn mặt và các yếu tố bên ngoài. Khuôn mặt phải được phát hiện bất kể góc mặt, cường độ sáng, quần áo và phụ kiện, màu tóc, tóc trên khuôn mặt, trang điểm, tuổi,... Các mô hình phát hiện khuôn mặt và hiệu suất thuật toán đã tiến triển từ các kỹ thuật Computer Vision thô sơ đến học máy (Machine Learning) sau đó đến các mạng học sâu (Deep Learning).

Mục tiêu của Face Detection là phát hiện khuôn mặt phía trước của con người từ hình ảnh và xác định vị trí khuôn mặt được phát hiện. Như chúng ta có thể thấy trong hình dưới, đầu vào là một hình ảnh và đầu ra là hình ảnh với một hộp giới hạn xung quanh mặt gọi là bounding box.



Hình 2.14: Input, Output của Face Detection

Ngày nay, có rất nhiều mô hình và thuật toán đã đạt được hiệu suất to lớn cho Face Detection. Tuy nhiên, trước khi đi vào Face Detection, em sẽ mô tả phương pháp Non maximum Suppression.

2.4.1 Non-maximum Suppression

Một đối tượng có khả năng sẽ có nhiều bounding box được sinh ra. Non-maximum suppression có thể giúp chúng ta loại ra những bounding box của cùng một đối tượng đó.



Hình 2.15: Non-Maximum Suppression

Các bước của non-maximum suppression:

- Bước 1: Đầu tiên sẽ tìm cách giảm bớt số lượng các bounding box bằng cách lọc bỏ toàn bộ những bounding box có xác suất chứa vật thể nhỏ hơn một ngưỡng threshold nào đó, thường là 0.5.
- Bước 2: Đối với các bounding box giao nhau, non-max suppression sẽ lựa chọn ra một bounding box có xác suất chứa vật thể là lớn nhất. Sau đó tính toán chỉ số giao thoa IoU với các bounding box còn lại. Nếu chỉ số này lớn hơn ngưỡng threshold thì điều đó chứng tỏ 2 bounding boxes đang overlap nhau rất cao. Ta sẽ xóa các bounding có có xác suất thấp hơn và giữ lại bounding box có xác suất cao nhất. Cuối cùng, ta thu được một bounding box duy nhất cho một vật thể.

2.4.2 MTCNN

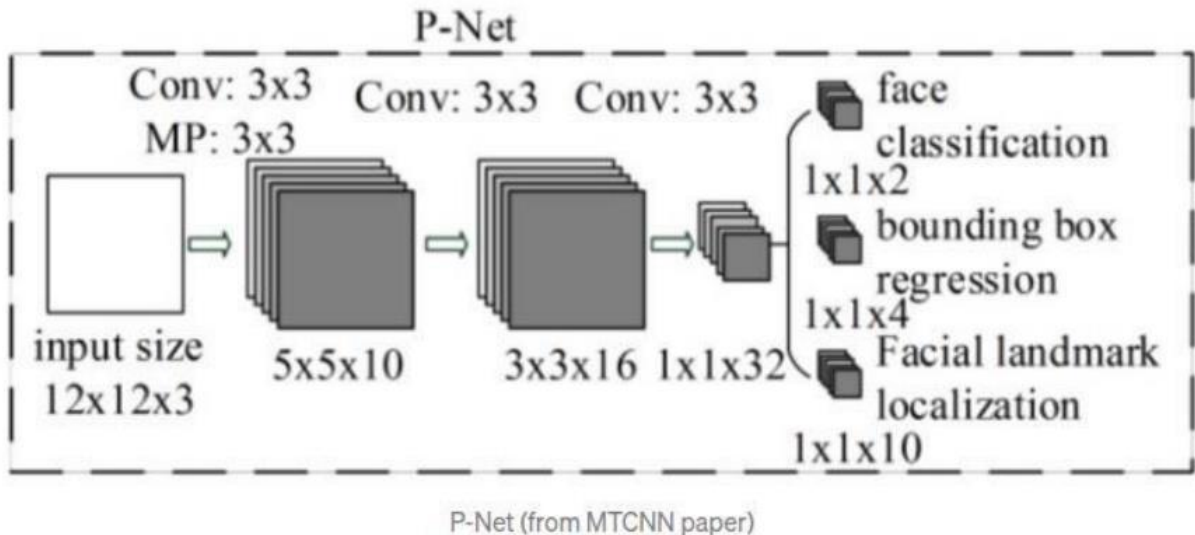
MTCNN [25] là viết tắt của Multi-task Cascaded Convolutional Networks. Thuật toán gồm 3 bước:

Bước 1: P-Net

Trước hết, một bức ảnh thường sẽ có nhiều hơn một người - một khuôn mặt. Ngoài ra, những khuôn mặt thường sẽ có kích thước khác nhau. Ta cần một phương thức để có thể nhận dạng toàn bộ số khuôn mặt đó, ở

các kích thước khác nhau. MTCNN đưa cho chúng ta một giải pháp, bằng cách sử dụng phép Resize ảnh, để tạo một loạt các bản copy từ ảnh gốc với kích cỡ khác nhau, từ to đến nhỏ, tạo thành 1 Image Pyramid.

Với mỗi một phiên bản copy-resize của ảnh gốc, ta sử dụng kernel 12×12 pixel và $\text{stride} = 2$ để đi qua toàn bộ bức ảnh, dò tìm khuôn mặt. Vì các bản copies của ảnh gốc có kích thước khác nhau, cho nên mạng có thể dễ dàng nhận biết được các khuôn mặt với kích thước khác nhau, mặc dù chỉ dùng 1 kernel với kích thước cố định (Ảnh to hơn, mặt to hơn; Ảnh nhỏ hơn, mặt nhỏ hơn). Sau đó, ta sẽ đưa những kernels được cắt ra từ trên và truyền qua mạng P-Net (Proposal Network). Kết quả của mạng cho ra một loạt các bounding boxes nằm trong mỗi kernel, mỗi bounding boxes sẽ chứa tọa độ 4 góc để xác định vị trí trong kernel chứa nó (đã được normalize về khoảng từ $(0,1)$) và điểm confident tương ứng.



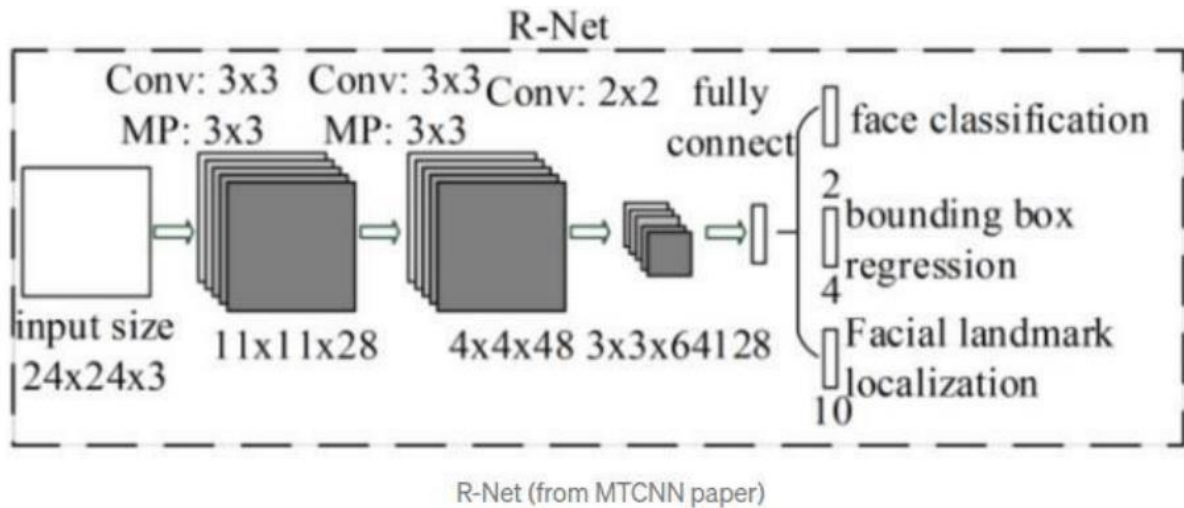
Hình 2.16: Cấu trúc P-Net trong MTCNN

Để loại trừ bớt các bounding boxes trên các bức ảnh và các kernels, ta sử dụng 2 phương pháp chính là lập mức Threshold confident - nhằm xóa đi các box có mức confident thấp và sử dụng NMS (Non-Maximum Suppression) để xóa các box có tỷ lệ trùng nhau vượt qua 1 mức threshold tự đặt nào đó.

Sau khi đã xóa bớt các box không hợp lý, ta sẽ chuyển các tọa độ của các box về với tọa độ gốc của bức ảnh thật. Do tọa độ của box đã được normalize về khoảng $(0,1)$ tương ứng như kernel, cho nên công việc lúc này chỉ là tính toán độ dài và rộng của kernel dựa theo ảnh gốc, sau đó nhân tọa độ đã được normalize của box với kích thước của kernel và cộng với tọa độ

của các góc kernel tương ứng. Kết quả của quá trình trên sẽ là những tọa độ của box tương ứng ở trên ảnh kích thước ban đầu. Cuối cùng, ta sẽ resize lại các box về dạng hình vuông, lấy tọa độ mới của các box và feed vào mạng tiếp theo, mạng R (Refine Network).

Bước 2: R-Net



Hình 2.17: Cấu trúc R-Net trong MTCNN [25]

Mạng R (Refine Network) thực hiện các bước như mạng P. Tuy nhiên, mạng còn sử dụng một phương pháp tên là padding, nhằm thực hiện việc chèn thêm các zero pixels vào các phần thiếu của bounding box nếu bounding box bị vượt quá biên của ảnh. Tất cả các bounding box lúc này sẽ được resize về kích thước 24x24, được coi như 1 kernel và feed vào mạng R. Kết quả sau cũng là những tọa độ mới của các box còn lại và được đưa vào mạng tiếp theo, mạng O.

Bước 3: O-Net

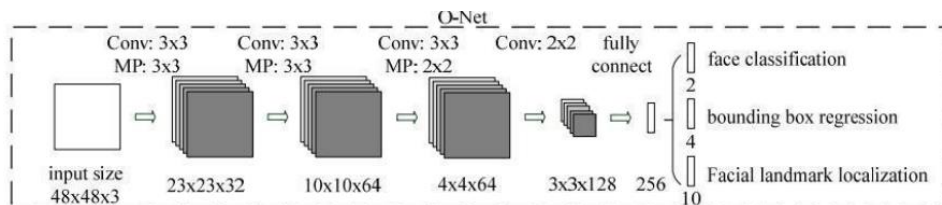
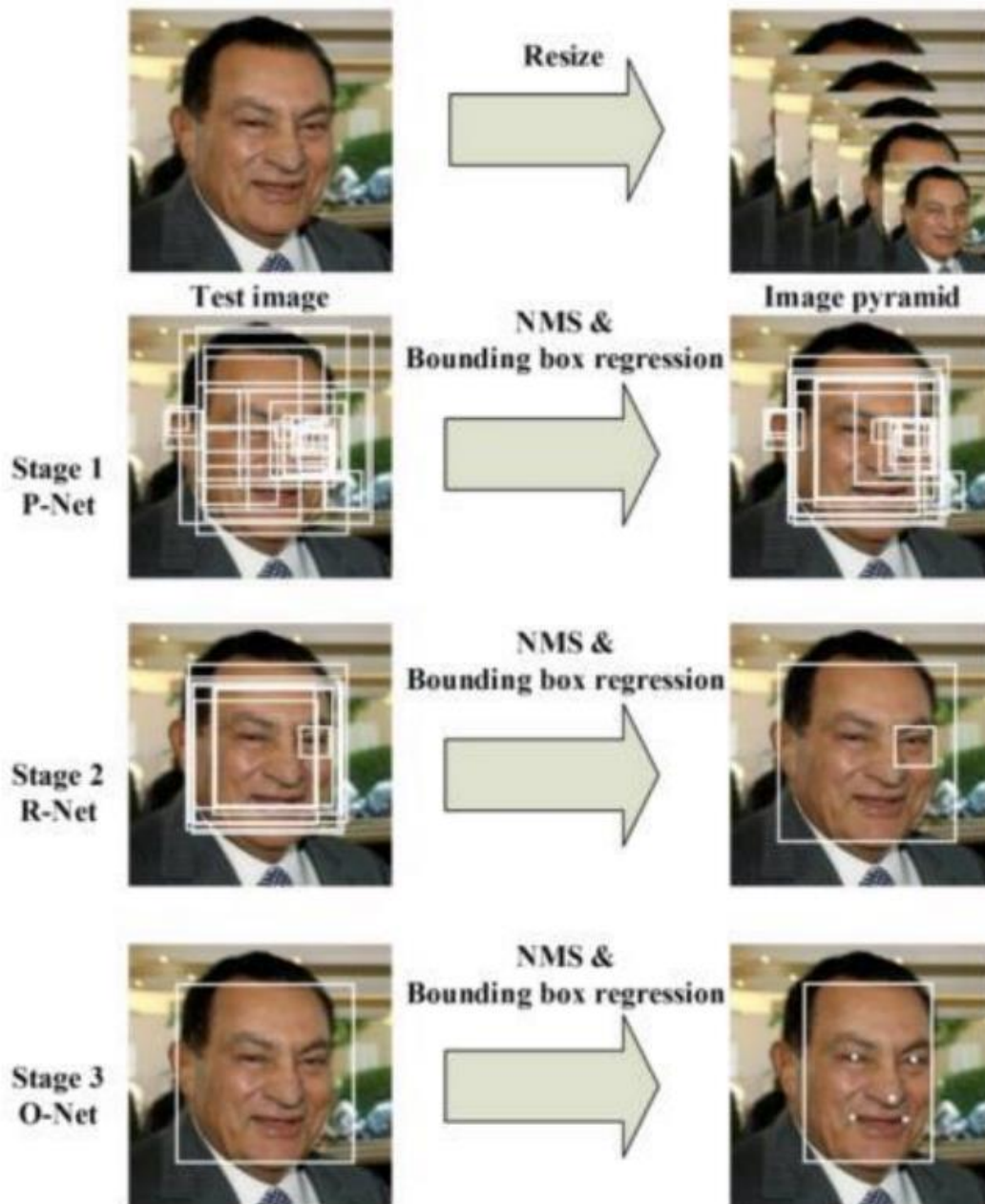


Fig. 2. The architectures of P-Net, R-Net, and O-Net, where “MP” means max pooling and “Conv” means convolution. The step size in convolution and pooling is 1 and 2, respectively.

Hình 2.18: Cấu trúc O-Net trong MTCNN

Cuối cùng là mạng O (Output Network), mạng cũng thực hiện tương tự như việc trong mạng R, thay đổi kích thước thành 48x48. Tuy nhiên, kết quả đầu ra của mạng lúc này không còn chỉ là các tọa độ của các box nữa, mà trả về 3 giá trị bao gồm: 4 tọa độ của bounding box (out[0]), tọa độ 5 điểm landmark trên mặt, bao gồm 2 mắt, 1 mũi, 2 bên cánh môi (out[1]) và điểm confident của mỗi box (out[2]). Tất cả sẽ được lưu vào thành 1 dictionary với 3 keys kể trên.



Hình 2.19: Quá trình 3 bước của MTCNN

2.4.3 Sample and Computation Redistribution for Face Detection

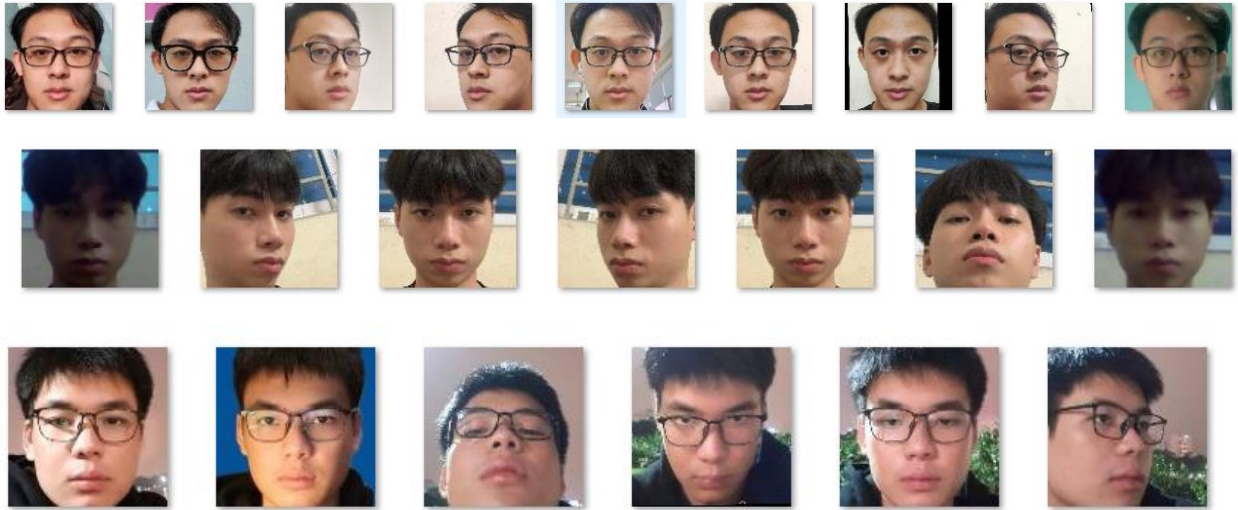
Hiện nay, bài toán Face Detection đã được giải quyết khá trọn vẹn cho dù kích thước khuôn mặt trong bức ảnh có nhiều kích thước khác nhau. Mô hình hiện đại nhất có thể kể đến RetinaNet [8]. RetinaNet đạt được hiệu suất rất lớn trên bộ WIDER FACE. Tuy nhiên RetinaNet triển khai thực tế lại mất quá nhiều thời gian để phát hiện khuôn mặt. Một mô hình mới sinh ra nhằm cải thiện nhược điểm này là mô hình SCRFD. Sample and Computation Redistribution for Face Detection (SCRFD) là một mô hình Face Detection cải tiến từ RetinaNet [8], TinaFace [9] để có thể chạy được trên thời gian thực.

Kiến trúc của SCRFD gồm 3 phần: backbone, neck, head. SCRFD dựa theo RetinaNet và TinaFace dùng Resnet-50 làm backbone và sử dụng Path Aggregation Feature Pyramid Network (PAFPN) [8] làm neck để xây dựng khối Feature Extractor. Đối với phần head, SCRFD sử dụng module tăng cường feature trên mỗi feature pyramid để có thể tìm hiểu môi trường xung quanh thông qua inception block [10]. Sau head 4 lớp tích chập 3×3 nối liên tiếp với nhau.

SCRFD sử dụng độ phân giải chuẩn VGA (640x480) và có kết quả tốt hơn TinaFace trên bộ dữ liệu WIDER FACE. Khi cạnh dài của mẫu trong tập WIDER FACE được cố định là 640 pixel, hầu hết các mẫu dễ thì kích thước khuôn mặt đều lớn hơn 32×32 pixel và hầu hết các mẫu trung bình, kích thước khuôn mặt lớn hơn 16×16 pixel. Đối với các mẫu khó, 78,93% kích thước nhỏ hơn 32×32 , 51,85% kích thước nhỏ hơn 16×16 và 13,36% kích thước nhỏ hơn 8×8 .

2.5 Face Alignment

Face Alignment là giai đoạn thứ 2 trong hệ thống Nhận diện khuôn mặt. Face Alignment sẽ trích ra khuôn mặt, căn chỉnh, thay đổi kích thước, chuẩn hóa rồi đưa vào mô hình để huấn luyện. Mục tiêu của Face Alignment nhằm cải thiện độ chính xác



Hình 2.20: Khuôn mặt được cắt ra từ Face Detection

2.6 Feature Extraction

Feature Extraction là khối quan trọng nhất trong hệ thống nhận dạng khuôn mặt. Nó trích xuất một vector feature hoặc có thể gọi là vector embedding từ kết quả của giai đoạn Face Alignment. Mục tiêu của Feature Extraction là trích xuất được một vector feature chứa đủ thông tin để biểu diễn khuôn mặt và phân biệt giữa các khuôn mặt khác nhau. Một hệ thống nhận diện khuôn mặt tốt thì phải đảm bảo phát hiện được khuôn mặt đồng thời vector feature phải đại diện được cho chính người đó.

Đồ án này sẽ tập trung vào khối Feature Extraction hay chính là model Face Recognition. Chi tiết về Feature Extraction sẽ được trình bày trong Chương 3.

2.7 Feature Matching

Feature Matching sử dụng vector feature từ giai đoạn Feature Extraction để phân loại một người trong hình ảnh. Thuật toán sử dụng cơ sở dữ liệu của các vector feature và so sánh chúng với vector feature mới được trích xuất. Sau đó sẽ trả ra tên của người đó.

2.8 Loss Function

Mục tiêu chính của Loss Function là thiết kế một hàm mục tiêu sao cho mô hình có thể phân loại tốt nhất có thể. Khả năng phân loại của mô hình có thể được đo bằng khoảng cách của các vector feature trong cùng 1 cụm (intra-loss) và khoảng cách giữa chúng (inter-loss). Mục tiêu là để

khoảng cách các vecto feature trong cùng 1 cụm càng nhỏ càng tốt đồng thời khoảng cách giữa các lớp càng lớn càng tốt.

2.8.1 Softmax Loss

Softmax là 1 hàm mất mát được sử dụng nhiều nhất trong bài toán phân loại.

$$L_s = -\frac{1}{N} \sum_{i=1}^N \log \frac{\exp(W_{y_i}^T x_i + b_{y_i})}{\sum_{j=1}^n \exp(W_j^T x_i + b_j)}$$

$x_i \in \mathbb{R}^d$ là vecto feature thứ i trong y_i class.

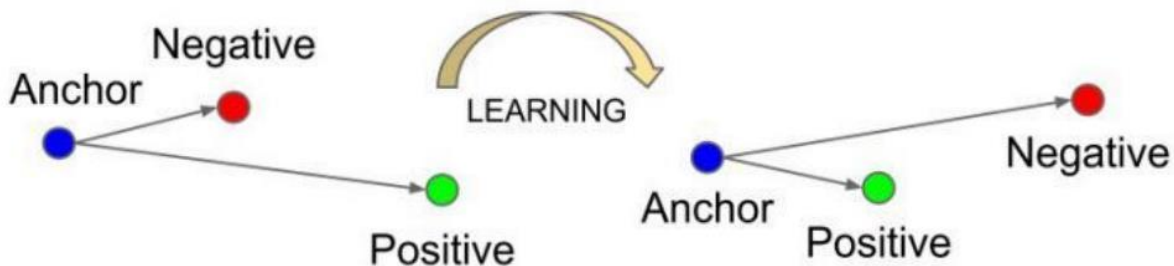
$W_j \in \mathbb{R}^d$ là vecto cột thứ j của ma trận trọng số $W \in \mathbb{R}^{d \times n}$.

b là bias. N là batch size. n là số class.

Trong bài toán Nhận diện khuôn mặt, Softmax Loss có 1 nhược điểm là nó không giảm khoảng cách giữa các vecto trong cụm với nhau. Điều này làm cho các vecto feature không thể phân loại khi so sánh 2 khuôn mặt khác nhau trong tập dữ liệu.

2.8.2 Triplet Loss

Triplet Loss là một hàm mất mát được tối ưu hóa bằng cách giảm thiểu khoảng cách giữa anchor và positive trong khi tối đa hóa khoảng cách giữa anchor và negative. Những điểm này được thể hiện bởi các vectơ trong không gian feature. Quá trình học được minh họa trong Hình dưới.



Hình 2.21: Minh họa quá trình học của Triplet Loss

Để áp dụng triplet loss, chúng ta cần lấy ra 3 bức ảnh trong đó có một bức ảnh là anchor. Trong 3 ảnh thì ảnh anchor được cố định trước. Chúng ta sẽ lựa chọn 2 ảnh còn lại sao cho một ảnh là negative (của một người khác với anchor) và một ảnh là positive (cùng một người với anchor).



Anchor



Positive

Hình 2.22: Anchor vs Positive



Anchor



Negative

Hình 2.23: Anchor vs Negative

Kí hiệu ảnh Anchor, Positive, Negative lần lượt là A, P, N.

Mục tiêu của hàm loss function là tối thiểu hóa khoảng cách giữa 2 ảnh khi chúng là negative và tối đa hóa khoảng cách khi chúng là positive. Như vậy chúng ta cần lựa chọn các bộ 3 ảnh sao cho:

- Ảnh Anchor và Positive khác nhau nhất: cần lựa chọn để khoảng cách $d(A,P)$ lớn. Điều này cũng tương tự như bạn lựa chọn một ảnh của mình hồi nhỏ so với hiện tại để thuật toán học khó hơn. Nhưng nếu nhận biết được thì nó sẽ thông minh hơn.

- Ảnh Anchor và Negative giống nhau nhất: cần lựa chọn để khoảng cách $d(A,N)$ nhỏ. Điều này tương tự như việc thuật toán phân biệt được ảnh của một người anh em giống bạn với bạn. Triplot loss lấy 3 bức ảnh làm input và trong mọi trường hợp ta kì vọng:

$$d(A,P) < d(A,N)$$

Để làm cho khoảng cách giữa vế trái và vế phải lớn hơn, chúng ta sẽ cộng thêm vào vế trái một hệ số α không âm rất nhỏ. Khi đó (1) trở thành:

$$\begin{aligned} d(A,P) + \alpha &\leq d(A,N) \\ \|f(A) - f(P)\|_2^2 + \alpha &\leq \|f(A) - f(N)\|_2^2 \\ \|f(A) - f(P)\|_2^2 - \|f(A) - f(N)\|_2^2 + \alpha &\leq 0 \end{aligned}$$

Sẽ không ảnh hưởng gì nếu ta nhận diện đúng ảnh Negative và Positive là cùng cặp hay khác cặp với Anchor. Mục tiêu của chúng ta là giảm thiểu các trường hợp mô hình nhận diện sai ảnh Negative thành Positive nhất có thể. Do đó để loại bỏ ảnh hưởng của các trường hợp nhận diện đúng Negative và Positive lên hàm loss function. Ta sẽ điều chỉnh giá trị đóng góp của nó vào hàm loss function về 0.

Suy ra, hàm Loss trở thành:

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

Trong đó n là số lượng các bộ 3 hình ảnh được đưa vào huấn luyện.

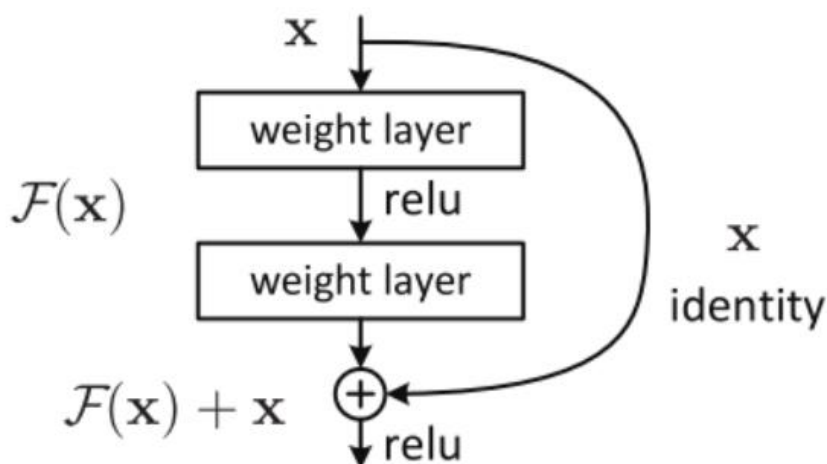
Như vậy khi áp dụng Triple loss vào các mô hình convolutional neural network chúng ta có thể tạo ra các biểu diễn vecto tốt nhất cho mỗi một bức ảnh. Những biểu diễn vecto này sẽ phân biệt tốt các ảnh Negative rất giống ảnh Positive. Và đồng thời các bức ảnh thuộc cùng một label sẽ trở nên gần nhau hơn trong không gian chiều euclidean.

Một chú ý quan trọng khi huấn luyện mô hình siam network với triplet function đó là chúng ta luôn phải xác định trước cặp ảnh (A,P) thuộc về cùng một người. Ảnh N sẽ được lựa chọn ngẫu nhiên từ các bức ảnh thuộc các nhãn còn lại. Do đó cần thu thập ít nhất 2 bức ảnh/1 người để có thể chuẩn bị được dữ liệu huấn luyện.

Tuy nhiên, hạn chế của Triplet Loss chính là việc xây dựng được mẫu bộ 3 khá khó khăn đồng thời số lượng các cặp mẫu bộ 3 tìm được rất lớn sẽ dẫn tới sự bùng nổ tính toán và ổn định chậm. Đây là một vấn đề đặc biệt nghiêm trọng đối với các bộ dữ liệu lớn.

2.9 Kiến trúc Resnet

ResNet sử dụng kết nối "tắt" đồng nhất để xuyên qua các lớp, khối như vậy gọi là Residual Block:



Hình 2.24: Residual Block

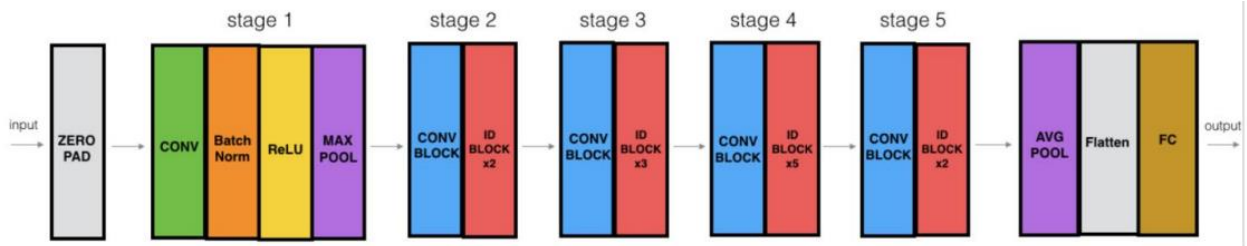
Các layer của ResNet gồm có convolution, pooling, activation và fully-connected layer. Một nối tắt từ input x tới cuối Residual Block bởi phép cộng, việc này sẽ chống lại việc đạo hàm bằng 0. Với $H(x)$ là giá trị dự đoán, $F(x)$ là giá trị thật (nhãn), chúng ta muốn $H(x)$ bằng hoặc xấp xỉ $F(x)$. Việc $F(x)$ có được từ x như sau:

$$x \rightarrow \text{weight}_1 \rightarrow \text{ReLU} \rightarrow \text{weight}_2$$

$$\text{Giá trị } H(x) \text{ có được bằng cách: } F(x) + x \rightarrow \text{ReLU}$$

Khi tăng số lượng các layer sẽ làm cho model giảm độ chính xác nhưng kiến trúc mạng sâu hơn thì mới có thể hoạt động tốt.

Hình dưới đây mô tả chi tiết kiến trúc mạng neural ResNet :



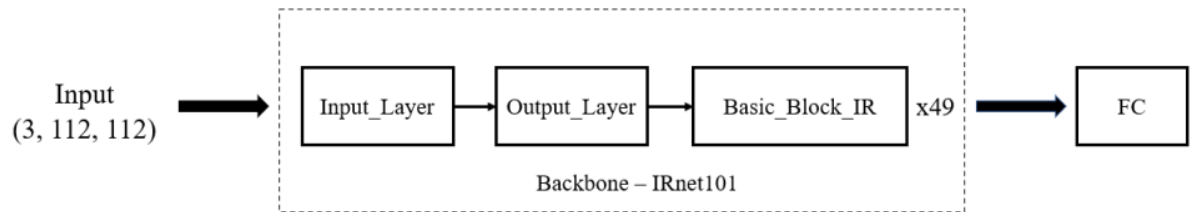
Hình 2.25: Kiến trúc Resnet-50

"ID BLOCK" là Identity block. Kiến trúc của Resnet-50 được mô tả như sau:

- Zero-padding : Input với (3,3)
- Stage 1 : Tích chập (Conv1) với 64 filters với shape(7,7), sử dụng stride (2,2). BatchNorm, MaxPooling (3,3).
- Stage 2 : Convolutional block sử dụng 3 filter với size 64x64x256, $f=3$, $s=1$. Có 2 Identity blocks với filter size 64x64x256, $f=3$.
- Stage 3 : Convolutional sử dụng 3 filter size 128x128x512, $f=3$, $s=2$. Có 3 Identity blocks với filter size 128x128x512, $f=3$.
- Stage 4 : Convolutional sử dụng 3 filter size 256x256x1024, $f=3$, $s=2$. Có 5 Identity blocks với filter size 256x256x1024, $f=3$.
- Stage 5 : Convolutional sử dụng 3 filter size 512x512x2048, $f=3$, $s=2$. Có 2 Identity blocks với filter size 512x512x2048, $f=3$.
- The 2D Average Pooling : sử dụng với kích thước (2,2).
- The Flatten. • Fully Connected (Dense) : sử dụng softmax activation.

CHƯƠNG 3. MÔ HÌNH

3.1 Kiến trúc

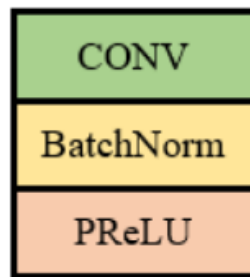


Hình 3.1: Kiến trúc mô hình

Input là 1 tensor 3 chiều với kích thước mỗi chiều là 112x112. Giá trị tensor được chuẩn hóa trong khoảng (-1;1)

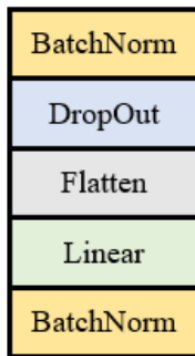
3.1.1 Backbone

a) Input_layer



```
(input_layer): Sequential(
  (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
  (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (2): PReLU(num_parameters=64) )
```

b) Output_layer

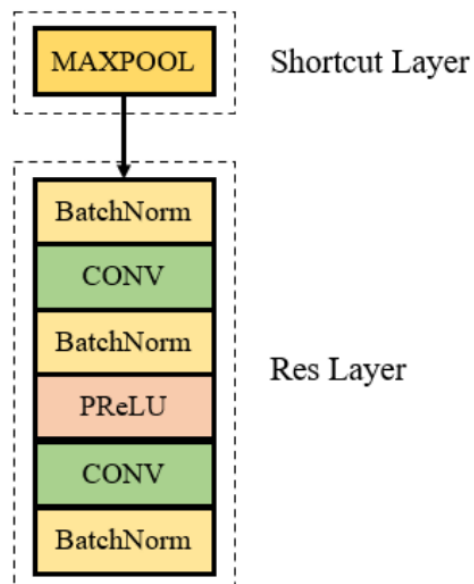


```

(output_layer): Sequential(
  (0): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
  (1): Dropout(p=0.4, inplace=False)
  (2): Flatten()
  (3): Linear(in_features=25088, out_features=512, bias=True)
  (4): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=False,
track_running_stats=True)
)

```

c) Basic_Block_IR



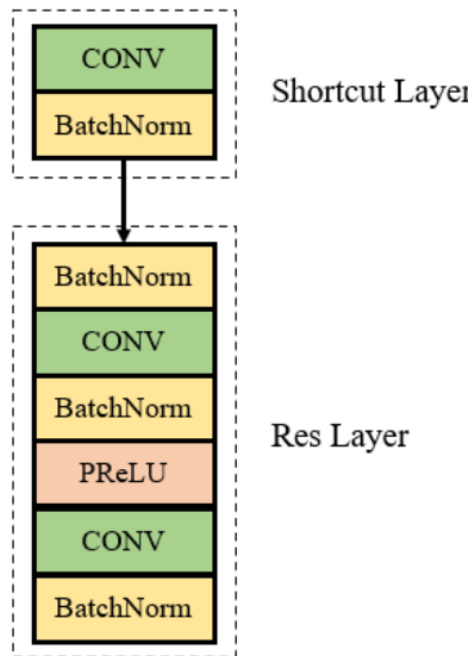
```
BasicBlockIR(
```



```

(shortcut_layer): MaxPool2d(kernel_size=1, stride=2, padding=0, dilation=1, ceil_mode=False)
(res_layer): Sequential(
  (0): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
  (2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (3): PReLU(num_parameters=64)
  (4): Conv2d(64, 64, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
  (5): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
)
)

```



```

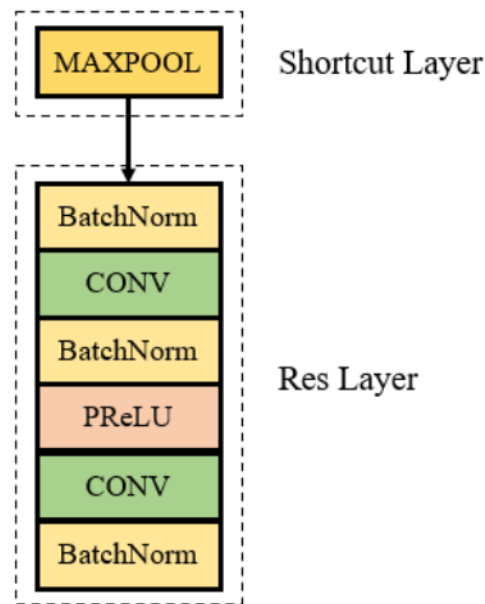
BasicBlockIR(
  (shortcut_layer): Sequential(
    (0): Conv2d(64, 128, kernel_size=(1, 1), stride=(2, 2), bias=False)
    (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)

```

```

)
(res_layer): Sequential(
  (0): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (1): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
  (2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (3): PReLU(num_parameters=128)
  (4): Conv2d(128, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
  (5): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
)
)

```



```

BasicBlockIR(
  (shortcut_layer): MaxPool2d(kernel_size=1, stride=1, padding=0, dilation=1, ceil_mode=False)
  (res_layer): Sequential(
    (0): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (1): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
    (2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (3): PReLU(num_parameters=256)
  )
)

```

```

(4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
(5): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
)
)
BasicBlockIR(
(shortcut_layer): MaxPool2d(kernel_size=1, stride=1, padding=0, dilation=1, ceil_mode=False)
(res_layer): Sequential(
(0): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(1): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
(2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(3): PReLU(num_parameters=512)
(4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
(5): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
))

```

3.1.2 Head

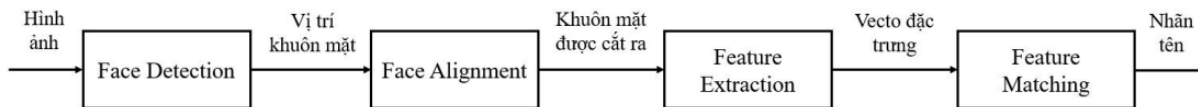


Feature đi qua backbone sẽ được làm dẹt thành 1 vecto thông qua khối Flatten và sử dụng khối Fully Connected để phân loại.

CHƯƠNG 4. XÂY DỰNG HỆ THỐNG NHẬN DIỆN

Trong chương này, em sẽ trình bày về hệ thống nhận diện em xây dựng được. Sử dụng bộ dữ liệu thành viên lớp điện tử 07-K63 đã trình bày ở phần trước làm database.

4.1 Hệ thống



Hình 4.1: Hệ thống Nhận diện khuôn mặt

4.1.1 Face Detection

SCRFD (Scale-aware Compact Region-based Face Detector) là một trong những mô hình face detection được sử dụng trong thư viện InsightFace. Mô hình này được xây dựng dựa trên kiến trúc RetinaFace, tuy nhiên nó được cải tiến để cải thiện hiệu suất và độ chính xác của việc phát hiện khuôn mặt trong các ảnh có độ phân giải khác nhau.

SCRFD sử dụng phương pháp region-based để phát hiện khuôn mặt, tức là mô hình sẽ tìm kiếm các vùng ứng cử trên ảnh có khả năng chứa khuôn mặt, sau đó sử dụng một mạng neural để xác định xem mỗi vùng ứng cử có chứa khuôn mặt hay không.

Mô hình SCRFD có ba phần chính: backbone, neck và head. Backbone được sử dụng để trích xuất đặc trưng từ ảnh đầu vào. SCRFD sử dụng một backbone dựa trên kiến trúc EfficientNet để cân bằng giữa độ phức tạp và hiệu suất của mô hình.

Neck của SCRFD được thiết kế để kết hợp các đặc trưng từ backbone và giảm độ phân giải của ảnh để làm cho việc phát hiện khuôn mặt trở nên dễ dàng hơn. Cụ thể, neck sử dụng một mạng neural tích chập để giảm độ phân giải của đặc trưng và tạo ra một số đặc trưng mới để cải thiện độ chính xác của việc phát hiện khuôn mặt.

Head của SCRFD được sử dụng để phân loại các vùng ứng cử đã được tạo ra bởi neck, xác định các vùng ứng cử có chứa khuôn mặt hay không và dự đoán hộp giới hạn của khuôn mặt nếu có. Head sử dụng một mạng neural đa lớp với nhiều đầu ra để thực hiện các tác vụ này.

Tóm lại, SCRFD là một mô hình phát hiện khuôn mặt hiệu quả và chính xác trong việc xác định khuôn mặt trong các ảnh có độ phân giải khác nhau. Mô hình này được sử dụng trong thư viện InsightFace để cung cấp các giải pháp nhận diện khuôn mặt và phân tích khuôn mặt cho các ứng dụng nhận diện khuôn mặt.

Em sử dụng model SCRFD trong Lib InsightFace. Kích thước ảnh đưa về 640x640. Model sẽ trả ra bounding box của từng khuôn mặt.

4.1.2 Face Alignment

Face alignment là quá trình tìm kiếm và điều chỉnh vị trí của các điểm đặc trưng trên khuôn mặt để đảm bảo chúng nằm trên cùng một vị trí tương đối trên mỗi bức ảnh hoặc video của một người.

Các điểm đặc trưng trên khuôn mặt bao gồm các đặc trưng như mũi, mắt, miệng, lông mày và cằm. Face alignment giúp cải thiện độ chính xác của các thuật toán phân tích khuôn mặt bằng cách đảm bảo rằng các điểm đặc trưng được định vị chính xác trên khuôn mặt.

Trong đồ án này em cắt từng khuôn mặt trong ảnh, resize kích thước về 128x128. Chuẩn hóa ma trận ảnh đầu vào trong khoảng $(-1; 1)$.

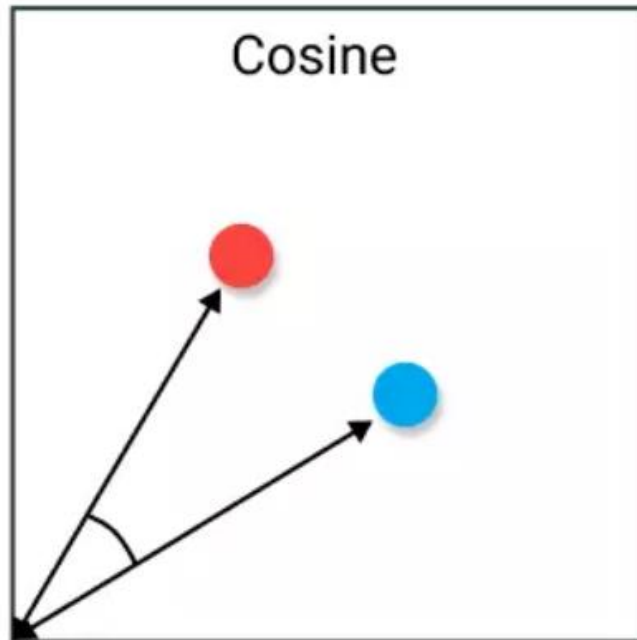
4.1.3 Feature Extraction

Đây chính là phần trọng tâm, sử dụng model AdaFace. Em đã đề cập ở trang 3

4.1.4 Feature Matching

Feature matching là quá trình tìm kiếm các đặc trưng (feature) của một bức ảnh trong một bức ảnh khác. Các đặc trưng được sử dụng trong feature matching thường là các điểm đặc biệt của bức ảnh, như các cạnh, góc, điểm nổi bật, hay các đường cong đặc biệt khác. Mục đích của feature matching là tìm kiếm các điểm tương đồng giữa hai hoặc nhiều bức ảnh để có thể ghép nối chúng lại với nhau hoặc so sánh chúng.

Quá trình feature matching bao gồm hai giai đoạn chính: trích xuất đặc trưng và matching. Trong giai đoạn trích xuất đặc trưng, các đặc trưng của bức ảnh được tìm kiếm và trích xuất ra. Trong giai đoạn matching, các đặc trưng này được so sánh với các đặc trưng trong bức ảnh khác để tìm ra các cặp đặc trưng tương đồng. Việc so sánh này có thể được thực hiện bằng cách tính khoảng cách giữa các vector đặc trưng, chẳng hạn như khoảng cách Euclidean hay cosine similarity.



Cosine similarity thường được sử dụng để giải quyết vấn đề của Euclidean distance ở không gian nhiều chiều. Ý tưởng đơn giản là tính góc tạo thành giữa 2 vector. Giá trị sẽ tương đương với phép dot product nếu cả 2 vector được norm về giá trị 1.

2 vector cùng hướng sẽ có cosine similarity bằng 1 và ngược hướng sẽ có giá trị -1. Lưu ý rằng, chiều dài không được sử dụng do đây là phương pháp tính theo hướng.

$$D(x, y) = \cos(\theta) = \frac{x \cdot y}{||x|| ||y||}$$

Nhược điểm:

Không tận dụng độ lớn của vector, chỉ tính theo hướng

Điều này vô tình làm mất mát thông tin so sánh

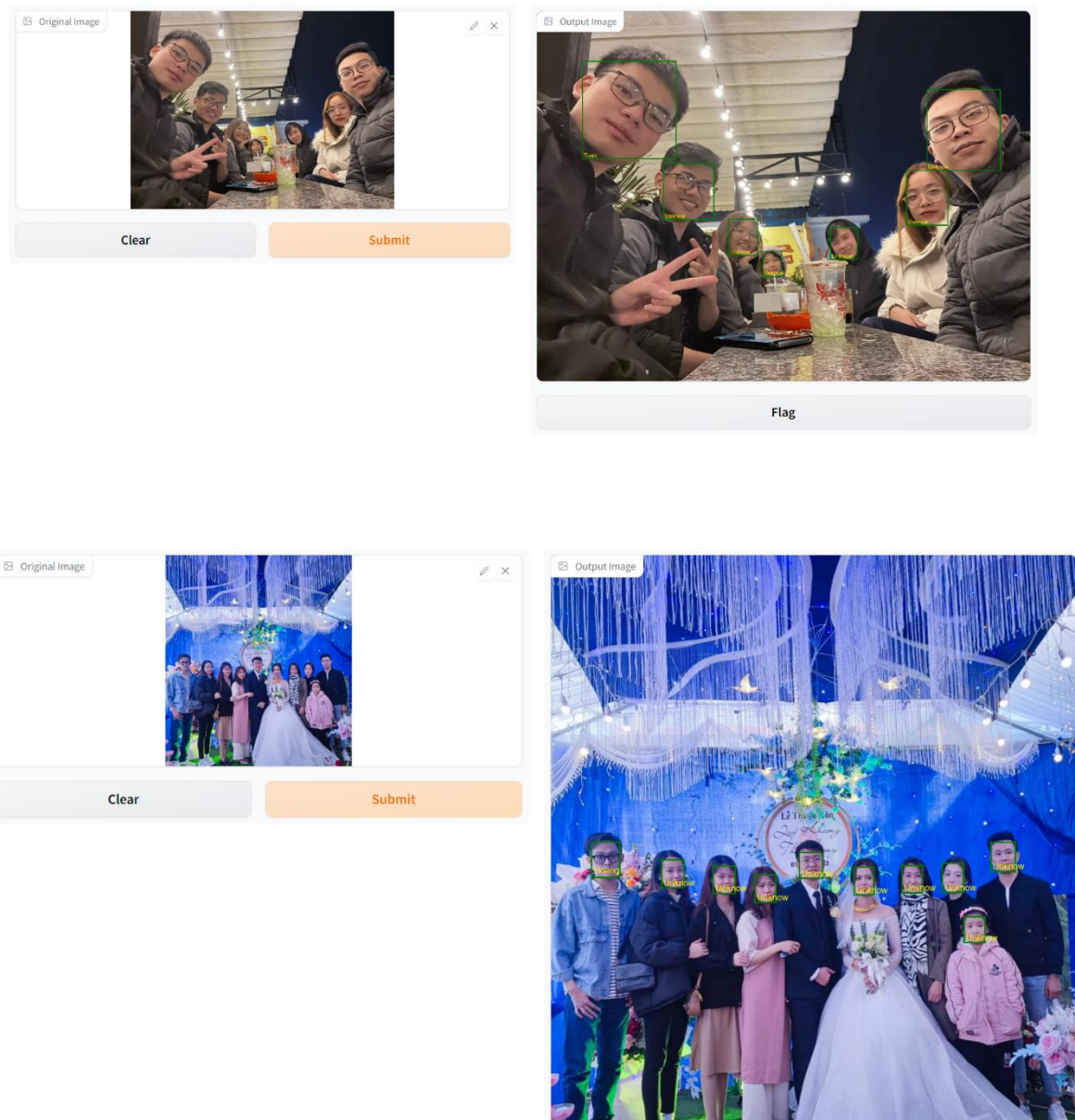
Use case: Thường được sử dụng trong các dữ liệu đa chiều và không quá phụ thuộc vào độ lớn của vector.

Em lưu sẵn các feature có trong database. Khi có 1 ảnh được đưa vào, hệ thống sẽ so sánh từng khuôn mặt trong ảnh với tất cả khuôn mặt có trong database.

Để so sánh 2 vecto feature, chúng ta sẽ dùng hàm cosine. 2 khuôn mặt giống nhau sẽ có giá trị cosine giữa 2 vecto feature lớn. Hệ thống sẽ trả ra tên dự đoán được.

4.2 Thử nghiệm

Để dễ dàng thao tác và sử dụng, em có viết một API cho hệ thống nhận diện khuôn mặt sử dụng Gradio. Thử nghiệm đối với 1 vài ảnh được cho bởi các hình bên dưới.

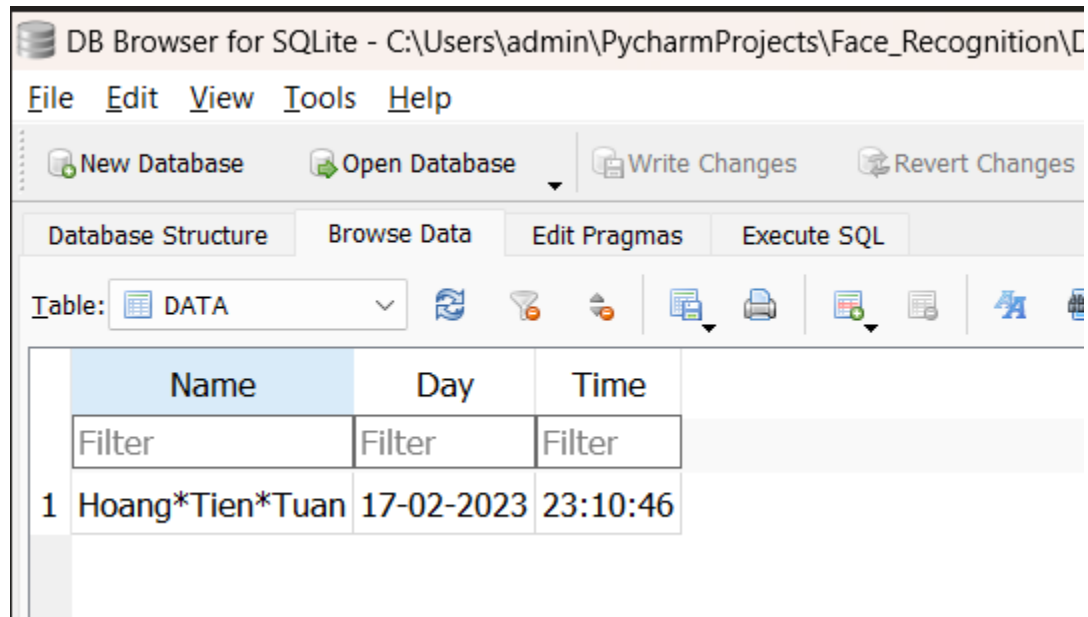


Hình 4.2: Kết quả thử nghiệm khi các thành viên có trong database và không có trong database

4.3 Điểm danh với webcam

Hệ thống điểm danh sử dụng webcam cho phép nhận diện khuôn mặt và xác định danh tính của các cá nhân từ hình ảnh chụp nếu có trong database.

Danh sách những người có trong database



DB Browser for SQLite - C:\Users\admin\PycharmProjects\Face_Recognition\...

File Edit View Tools Help

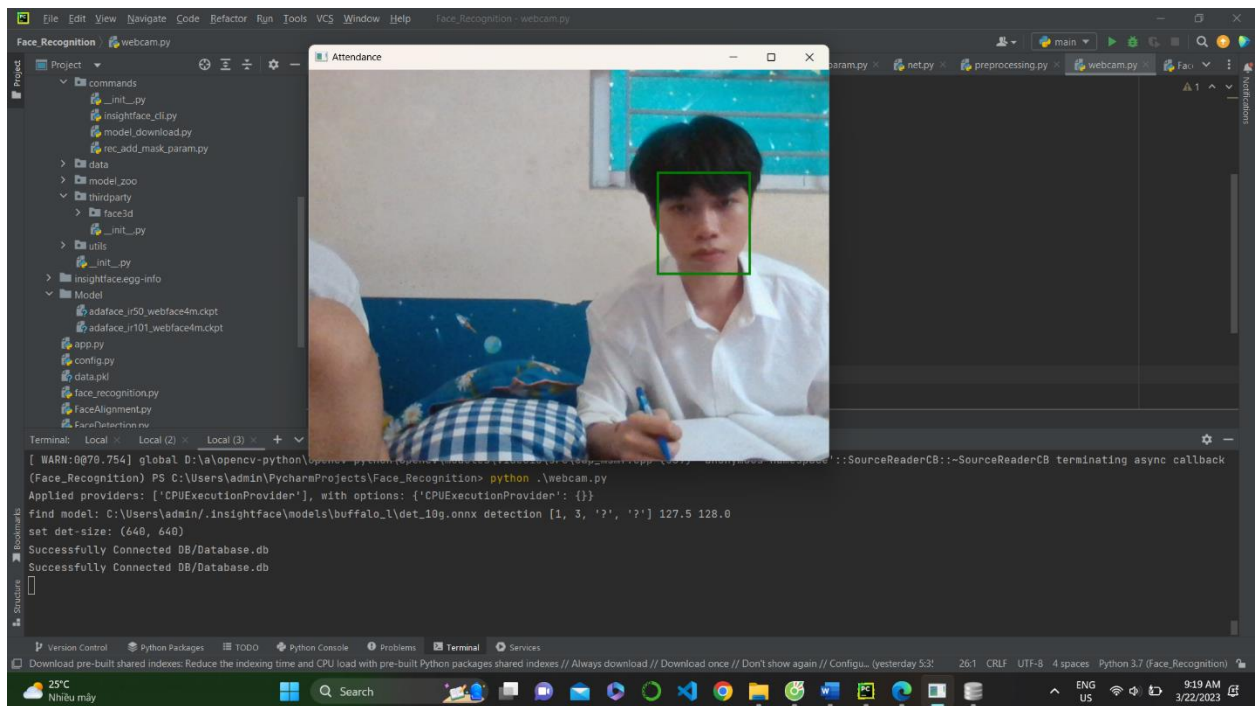
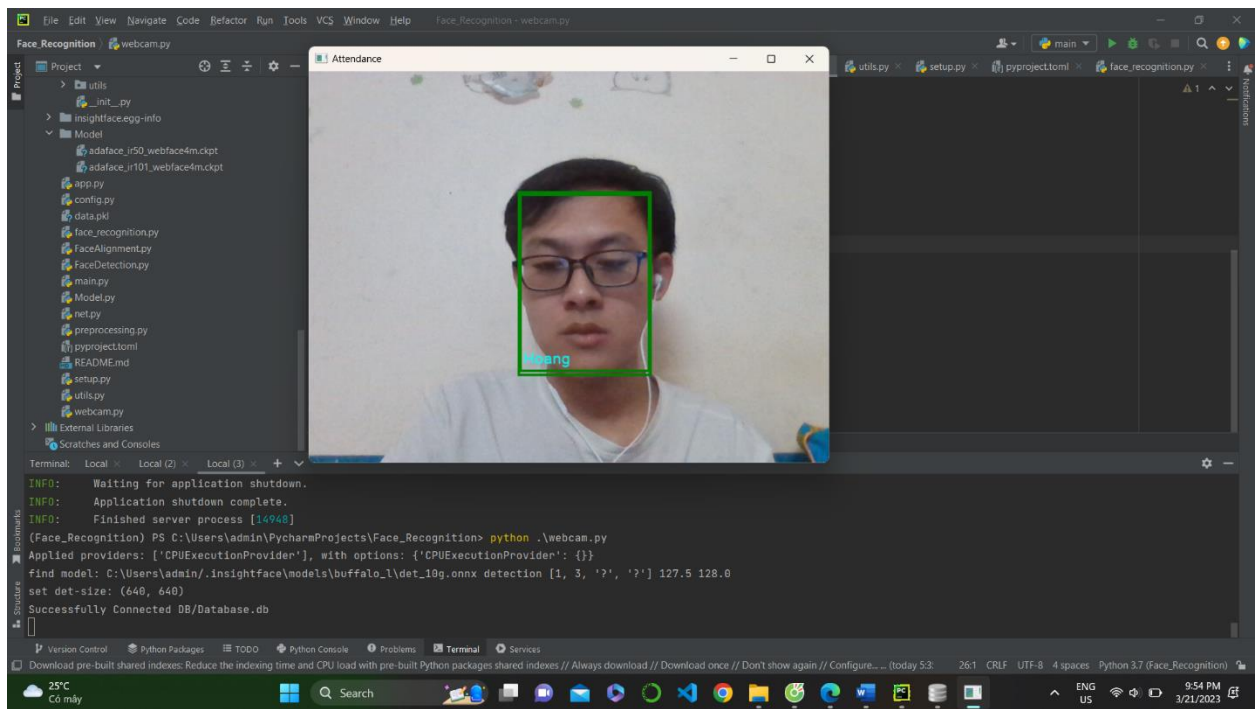
New Database Open Database Write Changes Revert Changes

Database Structure Browse Data Edit Pragmas Execute SQL

Table: DATA

	Name	Day	Time
	Filter	Filter	Filter
1	Hoang*Tien*Tuan	17-02-2023	23:10:46

Hệ thống sẽ báo kết nối thành công đến database. Nếu có trong danh sách sẽ trả về tên của người đó còn nếu không có thì sẽ trả lại unknow.



Kết quả điểm danh được lưu lại trong database

DB Browser for SQLite - C:\Users\admin\PycharmProjects\Face_Rec

File Edit View Tools Help

New Database Open Database Write Changes Revert

Database Structure Browse Data Edit Pragmas Execute SQL

Table: Attendance

	Name	Day	Time
	Filter	Filter	Filter
13	Hoang	17-02-2023	22:46:25
14	Hoang	17-02-2023	22:54:28
15	Hoang	17-02-2023	22:54:29
16	Hoang	17-02-2023	22:54:32
17	Hoang	17-02-2023	22:54:39
18	Hoang	17-02-2023	22:54:46
19	Hoang	17-02-2023	22:56:10
20	Hoang	17-02-2023	22:56:13
21	Hoang	17-02-2023	23:01:14
22	Hoang	17-02-2023	23:14:09
23	Hoang	17-02-2023	23:14:14
24	Hoang	17-02-2023	23:15:16
25	Hoang	17-02-2023	23:15:31
26	Hoang	21-03-2023	21:33:56
27	Hoang	21-03-2023	21:34:46
28	Hoang	21-03-2023	21:35:53
29	Hoang	21-03-2023	21:36:06
30	Hoang	21-03-2023	21:54:07
31	Hoang	21-03-2023	21:54:23
32	Tien	22-03-2023	09:18:48
33	Hoang	22-03-2023	09:18:48
34	Tien	22-03-2023	09:18:58
35	Hoang	22-03-2023	09:19:11

12 - 35 of 35

TÀI LIỆU THAM KHẢO

- [1] Kohonen, T. (1989). Self-organizing feature maps. In Self-organization and associative memory (pp. 119-157). Springer, Berlin, Heidelberg.
- [2] Turk, M. A., & Pentland, A. P. (1991, January). Face recognition using eigenfaces. In Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition (pp. 586-587). IEEE Computer Society.
- [3] Viennet, E., & Soulié, F. F. (1998). Connectionists methods for human face processing. In Face Recognition (pp. 124-156). Springer, Berlin, Heidelberg.
- [4] Colmenarez, A. J., & Huang, T. S. (1998). Face detection and recognition. In Face Recognition (pp. 174-185). Springer, Berlin, Heidelberg
- [5] Guo, G., Li, S. Z., & Chan, K. L. (2001). Support vector machines for face recognition. Image and Vision computing, 19(9-10), 631-638.
- [6] Guo, G., Li, S. Z., & Chan, K. L. (2001). Support vector machines for face recognition. Image and Vision computing, 19(9-10), 631-638.
- [7] Soumyadip Sengupta, Jun-Cheng Chen, Carlos Castillo, Vishal M Patel, Rama Chellappa, and David W Jacobs. Frontal to profile face verification in the wild. In 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 1–9, 2016.
- [8] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In CVPR, 2018.
- [9] Yanjia Zhu, Hongxiang Cai, Shuhan Zhang, Chenhao Wang, and Yichao Xiong. Tinaface: Strong but simple baseline for face detection. arXiv:2011.13183, 2020
- [10] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In CVPR, 2015
- [11] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. IEEE Transactions on Image Processing, 21(12):4695–4708, 2012

[12] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. CurricularFace: adaptive curriculum learning loss for deep face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5901–5910, 2020

[13] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4690–4699, 2019

[14] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. CosFace: Large margin cosine loss for deep face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5265–5274, 2018