Nguyễn Văn Long Mssv 1712024
Nguyễn Thị Bích Lan Mssv 1711884
Đề tài: Nhận diện khuôn mặt
GVHD: Trần Quốc Tiến Dũng

BÁO CÁO TIẾN ĐỘ ĐỒ ÁN MÔN HỌC

Các phần đã tìm hiểu:

- Đã hoàn thành lý thuyết 2 model phát hiện khuôn mặt: model Haar cascades và model MTCNN.
- Đã code (sử dụng thư viện) 2 model trên và chạy thử để đánh giá độ chính xác và thời gian.
- Tìm hiểu về phương pháp trích xuất đặc trưng ảnh (Facenet).

I. Model Haar cascades

Phương pháp Haar cascades của hai tác giả Paul Viola và Michael J.Jones là phương pháp xác định mặt người dựa theo hướng tiếp cận trên diện mạo.

Về tổng quan, phương pháp HA được xây dựng dựa trên sự kết hợp của 4 thành phần sau:

- **Các đặc trưng Haar-like**: các đặc trưng được đặt vào các vùng ảnh để tính toán các giá trị của đặc trưng, từ những giá trị đặc trưng này đưa vào bộ phân loại Adaboost ta sẽ xác định được ảnh có khuôn mặt hay không.
- **Ảnh tích hợp(Integral Image)**: thực ra đây là một công cụ giúp việc tính toán các giá trị đặc trưng Haar-like nhanh hơn.
- **Adaboost(Adaptive Boost)**: bộ phân loại (bộ lọc) hoạt động dựa trên nguyên tắc kết hợp các bộ phân loại yếu để tạo lên bộ phân loại mạnh. Adaboost sử dụng giá trị đặc trưng Haar-like để phân loại ảnh là mặt hay không phải mặt.
- **Cascade of Classifiers:** bộ phân loại tầng với mỗi tầng là một bộ phân loại Adaboost, có tác dụng tăng tốc độ phân loại.

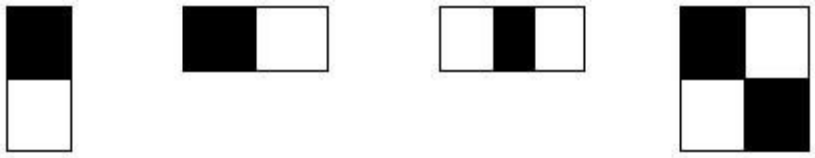
1. Haar like

Tất cả khuôn mặt người đều có chung những đặc điểm sau khi đã chuyển qua ảnh xám, ví dụ như:

- Vùng hai mắt sẽ tối hơn vùng má và vùng tráng, tức mức xám của vùng này cao hơn vượt trội so với hai vùng còn lại.

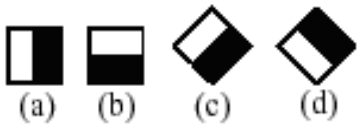
- Vùng giữa sống mũi cũng tối hơn vùng hai bên mũi...

Sau đây là 4 đặc trưng haar like cơ bản để xác định khuôn mặt:

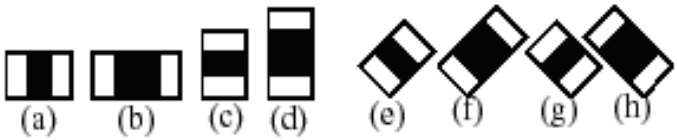


4 đặc trưng trên được mở rộng và chia làm 3 tập đặc trưng sau:

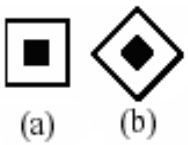
1. **Đặc trưng cạnh (edge feature):**



2. **Đặt trưng đường (line feature):**



3. **Đặt trưng xung quanh tâm:**



Giá trị của một đặc trưng Haar-like là sự khác biệt giữa tổng các giá trị xám của các pixel trong vùng "đen" với tổng các giá trị xám của các pixel trong vùng "trắng":

f(x) = Tổng vùng đen(pixel) – Tổng vùng trắng(pixel)

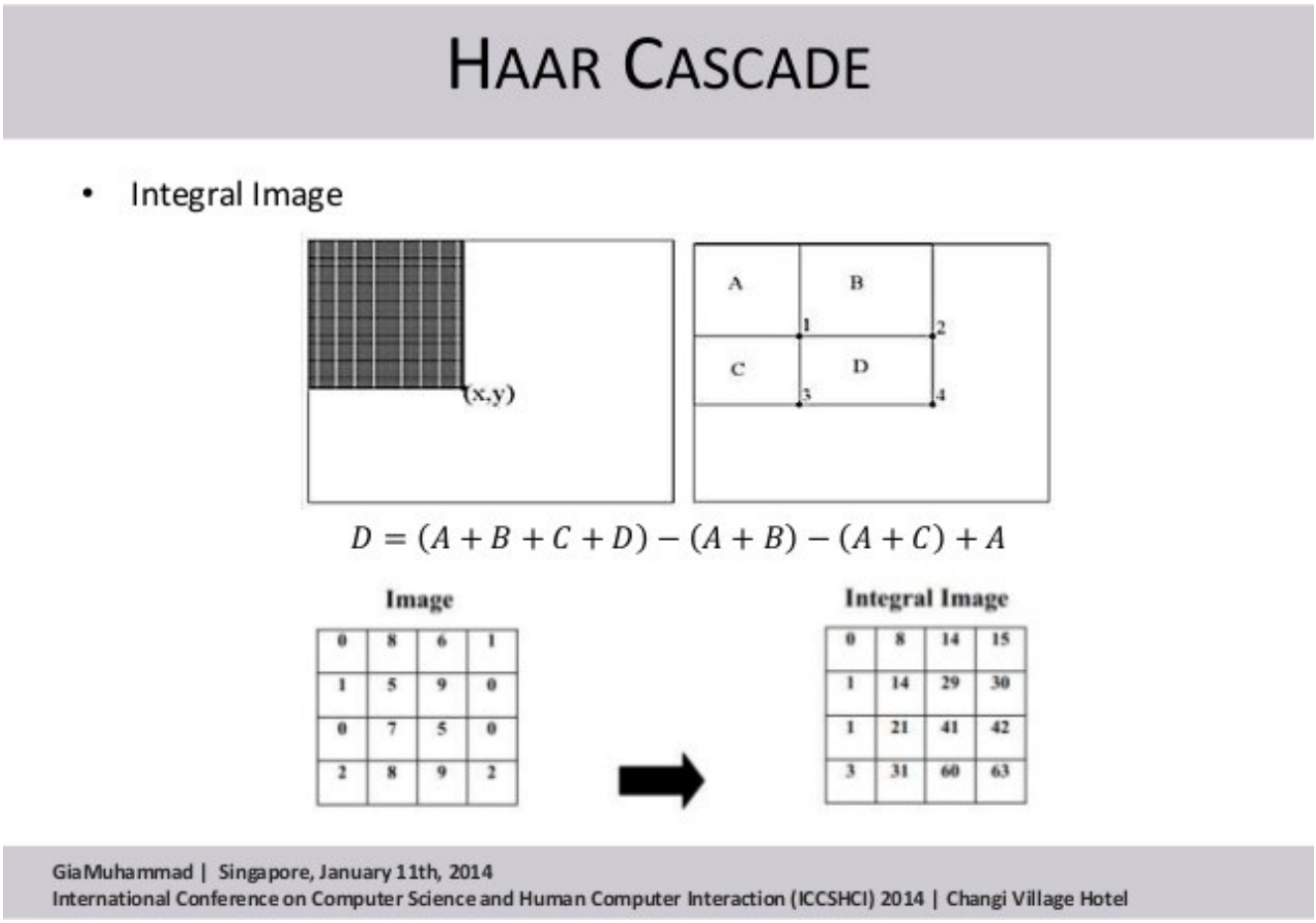
Vậy khi được đặt lên một vùng ảnh, đặc trưng Haar-like sẽ tính toán và đưa ra giá trị đặc trưng h(x) của vùng ảnh đó.

Để phát hiện khuôn mặt, hệ thống sẽ cho một cửa sổ con(sub-window) có kích thước cố định quét lên toàn bộ ảnh đầu vào. Như vậy sẽ có rất nhiều ảnh con ứng với từng cửa sổ con, các đặc trưng Haar-like sẽ được đặc lên các cửa sổ con này để từ đó tính ra giá trị của đặc trưng. Sau đó các giá trị này được bộ phân loại xác nhận xem khung hình đó có phải khuôn mặt hay không.

2. Ảnh tích hợp (integral image):

Số lượng đặc trưng Haar-like là rất nhiều và khối lượng tính toán giá trị các đặc trưng này là rất lớn. Vì vậy ảnh tích hợp được đưa ra nhằm tính toán nhanh chóng các đặc trưng, giảm thời gian xử lý.

Ví dụ chuyển một ảnh 4×4 có giá trị xám như bên dưới thành ảnh tích hợp:



Sau khi chuyển ảnh cần nhận dạng thành ảnh tích hợp, viêc tính toán giá trị các đặc trưng Haar-like sẽ rất đơn giản.

3. Adaboost(Adaptive Boost)

Boosting là 1 kỹ thuật dùng để tăng độ chính xác cho các thuật toán, nguyên lý cơ bản của nó là kết hợp nhiều bộ phân loại yếu thành 1 bộ phân loại mạnh.

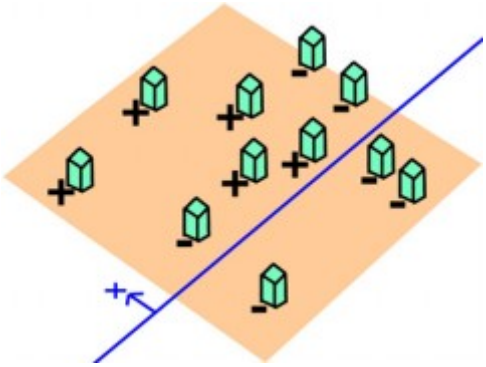
Adaboost là 1 thuật toán boosting dùng để xây dựng bộ phận phân lớp (classifier). Cải tiến trong Adaboost là gán thêm trọng số vào các mẫu (weight) để đánh dấu các mẫu khó nhận dạng.

Trong quá trình huấn luyện, cứ qua mỗi lần xây dựng bộ phân loại yếu ta sẽ cập nhật lại bộ trọng số cho các mẫu, cụ thể là: ta sẽ tăng giá trị trọng số cho các mẫu đã nhận dạng sai, giảm trọng số cho các mẫu nhận dạng đúng. Điều này có nghĩa là thuật toán sẽ tập trung vào các mẫu bị lớp trước đó phân loại sai.

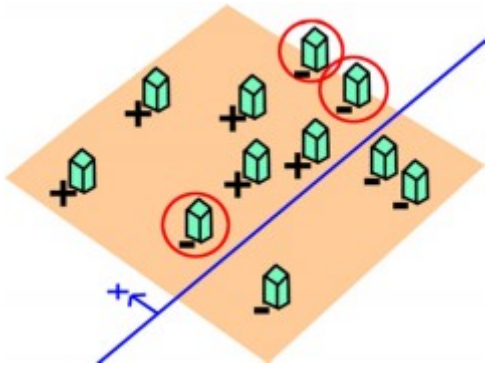
Thuật toán bắt đầu bằng việc khởi tạo trọng số cho các mẫu huấn luyện. Các trọng số này được khởi tạo bằng nhau. Các trọng số này cho thuật toán biết độ quan trọng của mẫu.

Ở mỗi vòng lặp, ta làm 2 việc:

- Thứ 1: tìm bộ phân lớp yếu dựa vào độ lỗi nhỏ nhất.

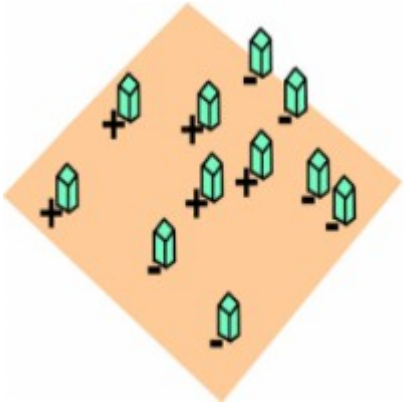


- Thứ 2: cập nhật trọng số theo nguyên tắc: ta sẽ tăng trọng số cho các mẫu hiện đang bị phân lớp sai và giảm trọng số cho các mẫu hiện đang được phân lớp đúng.

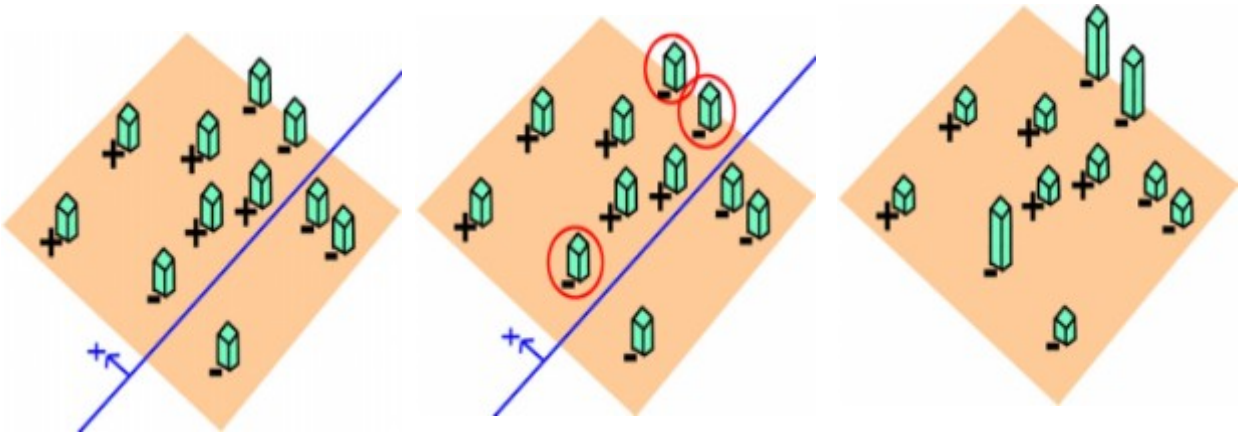


Để trực quan, ta hãy quan sát thuật toán thông qua chuỗi các hình vẽ dưới đây:

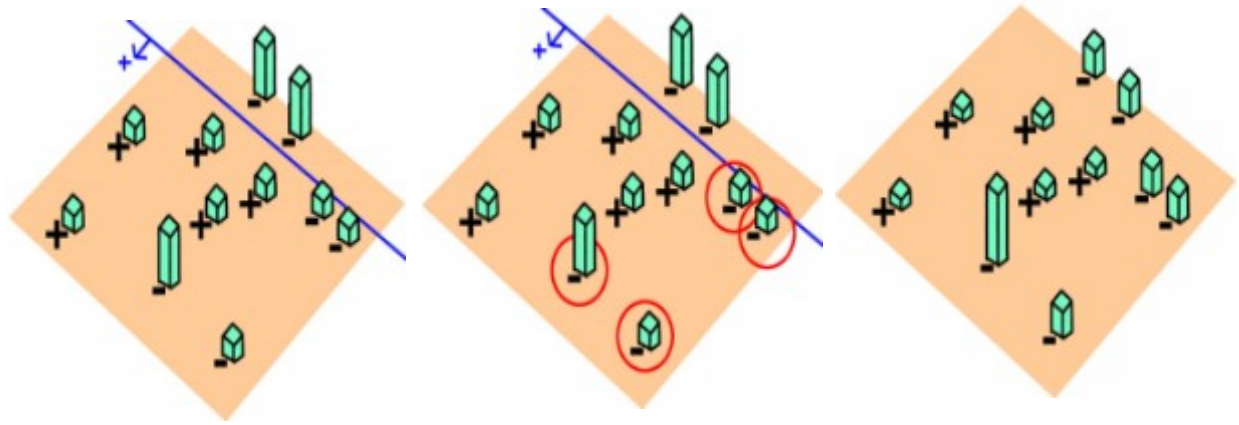
- Khởi tạo trọng số cho các mẫu:



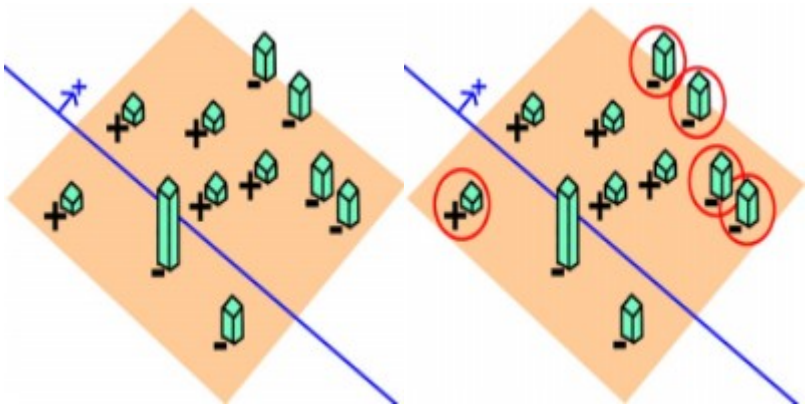
- Vòng lặp thứ 1:



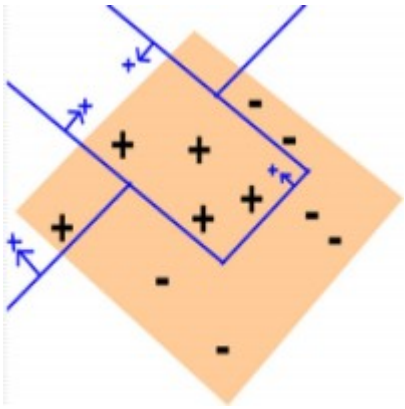
- Vòng lặp thứ 2:



- Vòng lặp thứ 3:



- Kết hợp các bộ phân lớp yếu lại:



- Cuối cùng, kết hợp tuyến tính các bộ phân lớp yếu lại ta được một bộ phân lớp mạnh.

4. Cascade of Classifiers

- Giả sử sau khi dùng Adaboost ta có 1 bộ phân loại mạnh gồm rất nhiều bộ phân loại yếu. Sau đó cho tất cả các cửa sổ con (sub window) qua lớp Adaboost để phát hiện khuôn mặt. Cách làm này sẽ rất tốn chi phí vì đa phần trong các tấm hình có rất ít cửa sổ con chứa khuôn mặt, và rất nhiều cửa sổ con là background rất dể nhận ra. Đối với các cửa sổ con này chỉ cần 1 lớp nhận dạng đơn giản là có thể phát hiện được.
- Mô hình cascade được xây dựng nhằm rút ngắn thời gian xử lý bằng 1 chuỗi phân lớp từ đơn giản đến phức tạp để loại bỏ từ từ các mẫu không phải khuôn mặt.
- Ta sẽ có một chuỗi các bộ phân lớp, trong đó mỗi bộ phân lớp được xây dựng bằng thuật toán Adaboost.
- Bộ phân lớp đầu tiên sẽ loại bỏ phần lớn các cửa sổ không phải khuôn mặt (negative sub window) và cho đi qua các cửa sổ được cho là khuôn mặt (positive sub window). Ở đây, bộ phân lớp này rất đơn giản và do đó, độ phức tạp tính toán cũng rất thấp. Tất nhiên, vì rằng nó đơn giản nên trong số các cửa sổ được nhận dạng là khuôn mặt sẽ có một số lượng lớn cửa sổ bị nhận dạng sai (không phải là khuôn mặt.)
- Những cửa chỗ được cho đi qua bởi bộ phân lớp đầu sẽ được xem xét bởi bộ phân lớp sau đó: nếu bộ phân lớp cho rằng đó không phải là khuôn mặt thì ta loại bỏ; nếu bộ phân lớp cho rằng đó là khuôn mặt thì ta lại cho đi qua và chuyển đến bộ phân lớp phía sau.
- Những bộ phân lớp càng về sau thì càng phức tạp hơn, đòi hỏi sự tính toán nhiều hơn. Người ta gọi những cửa sổ con (mẫu) mà bộ phân lớp không loại bỏ được là những mẫu khó

nhận dạng. Những mẫu này càng đi sâu vào trong chuỗi các bộ phân lớp thì càng khó nhận dạng. Chỉ những cửa sổ đi qua được tất cả các bộ phân lớp thì ta mới quyết định đó là khuôn mặt.

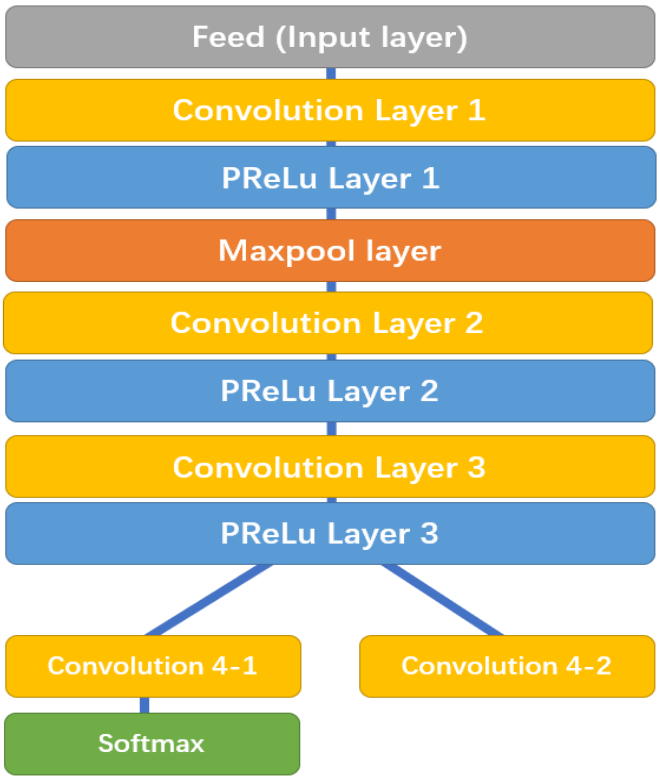
II. MTCNN (multi-task Cascaded convolutional neural network)

Trước khi tìm hiểu về MTCNN thì cần nắm về mạng neural.

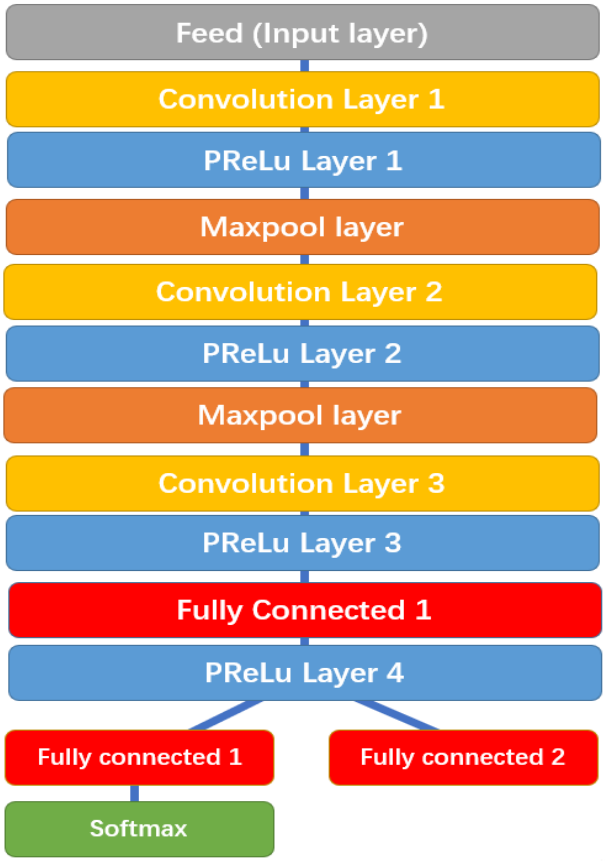
MTCNN là 1 model giúp chúng ta xác định khuôn mặt trong 1 bức hình.

MTCNN là 1 cấu trúc mạng 3 tầng: Pnet, Rnet, Onet.

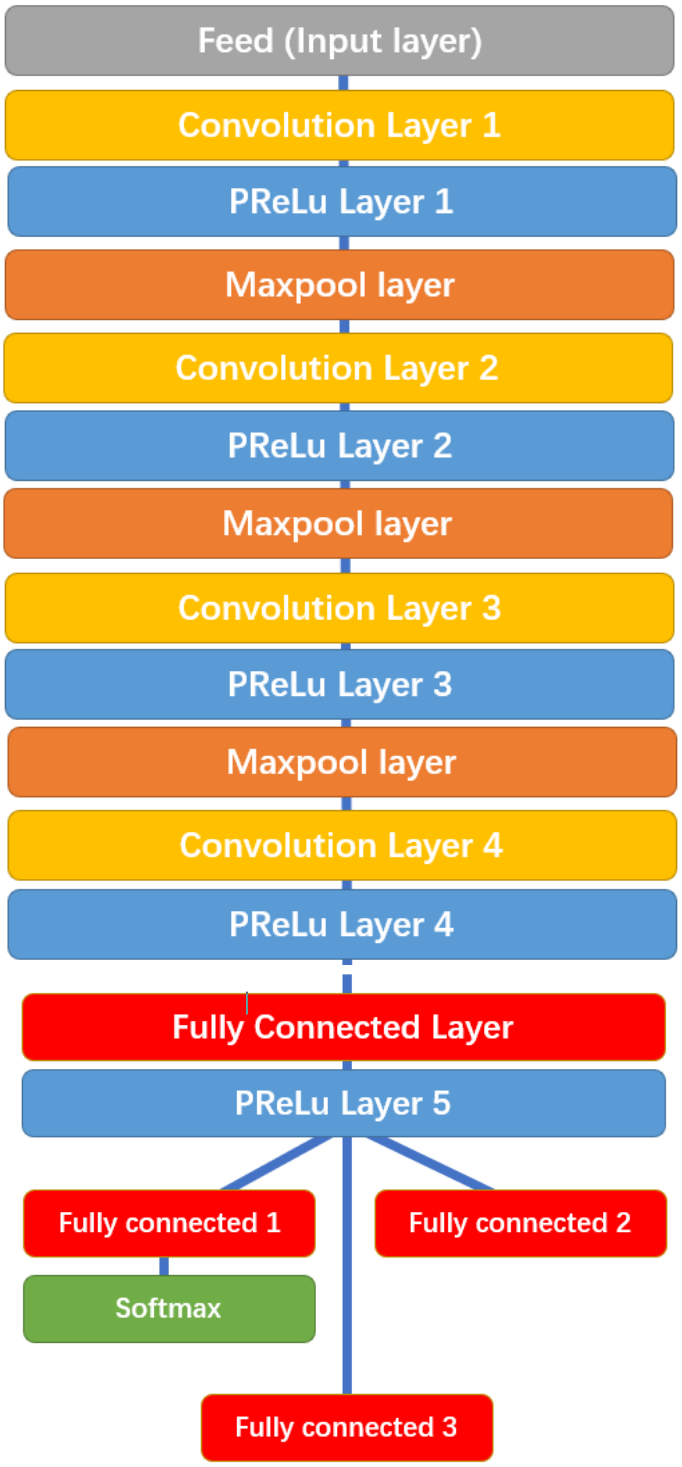
- **Pnet:** với input là 1 tấm ảnh, output là thông tin của các bounding boxes gồm 2 thành phần: 1 vector xác suất của các khuôn mặt nằm trong bounding box và tọa độ của các bounding boxes.



Rnet: có tác dụng lọc các bounding boxes kém chất lượng ở lớp Pnet cung cấp.



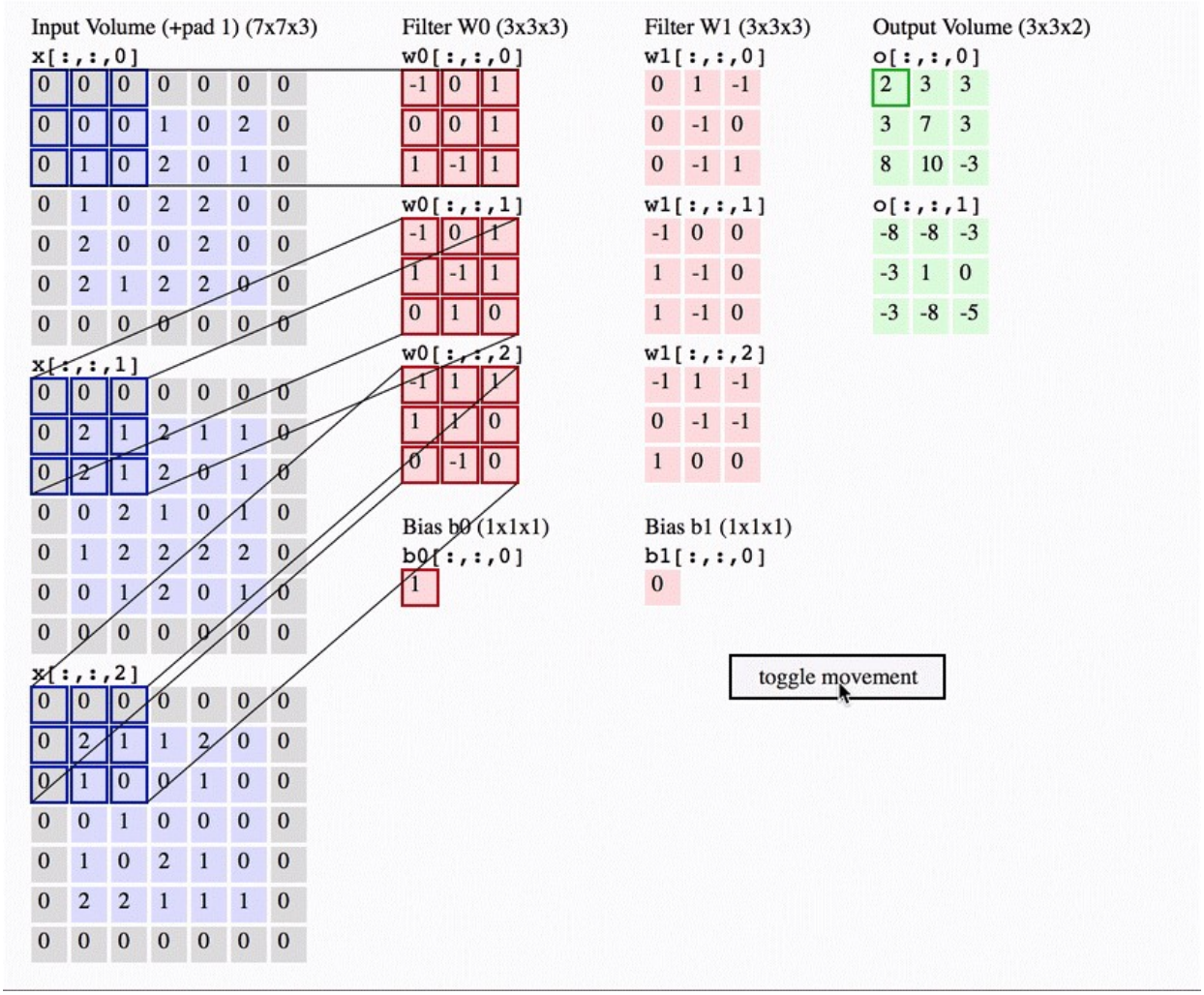
- **Onet:** input là các bounding boxes ở Rnet, output gồm 3 thành phần: 1 vector xác suất của khuôn mặt nằm trong bounding box, tọa độ của bounding box và tọa độ của các mốc trên khuôn mặt (vị trí mắt trái, phải, mũi, miệng trái, phải).



Sau đây em sẽ trình bày thêm 1 số khái niệm đã đề cập ở các mạng neural trên.

Convolution layer (lớp tích chập):

- Mục tiêu của các lớp tích chập này là để trích xuất các đặc trưng của ảnh đầu vào bằng cách so khớp với các filter.



Relu layer:

- ReLU layer áp dụng hàm kích hoạt (activation function) max(0,x) lên đầu ra của Conv Layer, có tác dụng đưa các giá trị âm về thành 0. Layer này không thay đổi kích thước của ảnh và không có thêm bất kì tham số nào.

- ReLu là đưa ảnh một mức ngưỡng, ở đây là 0. Để loại bỏ các giá trị âm không cần thiết mà có thể sẽ ảnh hưởng cho việc tính toán ở các layer sau đó.

Pooling layer:

- Pool Layer thực hiện chức năng làm giảm chiều không gian của đầu vào và giảm độ phức tạp tính toán của model

Fully connected layer:

Tên tiếng viết là Mạng liên kết đầy đủ. Tại lớp mạng này, mỗi một nơ-ron của layer này sẽ liên kết tới mọi nơ-ron của lớp trước đó.

Softmax:

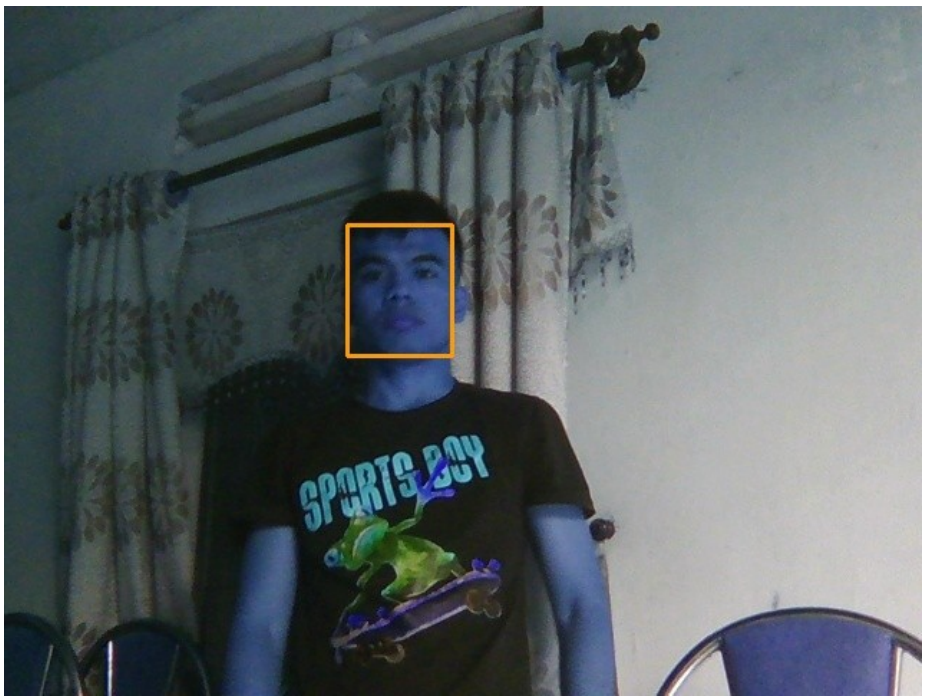
Là 1 hàm kích hoạt, tạo ra 1 vector phân phối xác suất với đầu vào là đầu ra của lớp fully connected layer trước đó.

IV. So sánh MTCNN và Haar cascades

- Bộ dữ liệu được tạo từ chụp trực tiếp từ webcam của laptop, đã tạo nhiều bộ dữ liệu khác nhau về góc chụp, độ sáng, độ khó nhưng vì sử dụng webcam nên chất lượng ảnh không được tốt có thể ảnh hưởng đến độ chính xác của thuật toán. Và có sử dụng thêm bộ dữ liệu tải từ google, chất lượng ảnh tốt hơn.

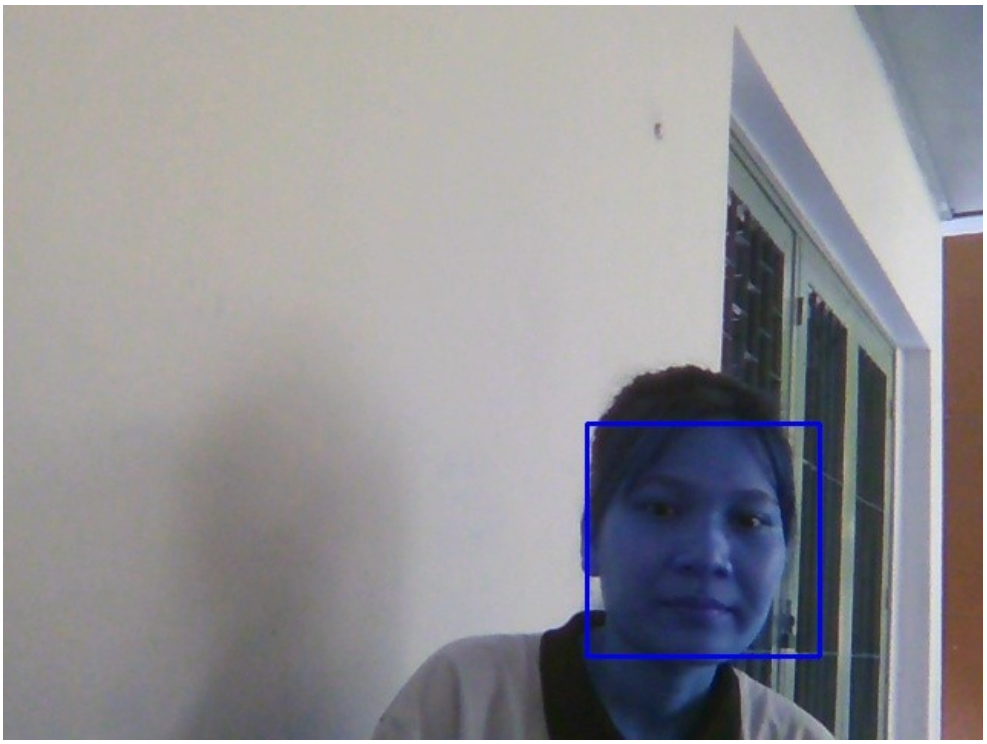
- MTCNN:

  - Độ chính xác của thuật toán này rất cao. Nhiều ảnh khuôn mặt nhỏ, mờ nhưng vẫn phát hiện được:



Độ chính xác của thuật toán này trong khoảng 93-98%, tốc độ xử lí khoảng 6 ảnh trong 1s. Với các ảnh trên thì thuật toán Haar cascade không phát hiện được khuôn mặt.

- Haar cascade có độ chính xác trong khoảng 60-85%, tốc độ xử lí là 11-14 tấm ảnh trong 1s. Thuật toán này thường chỉ có thể phát hiện được khuôn mặt chính diện, rõ:



- Các trường hợp mặt bị nghiêng, mờ, bị chói sáng thì thuật toán haar cascade phát hiện không tốt. Vì lí do thuật toán này chỉ sử dụng 1 ít đặc trưng cơ bản (đặc trưng cạnh, đường, xung quanh tâm), các đặc trưng này là cố định, không được học. Còn đối với thuật toán

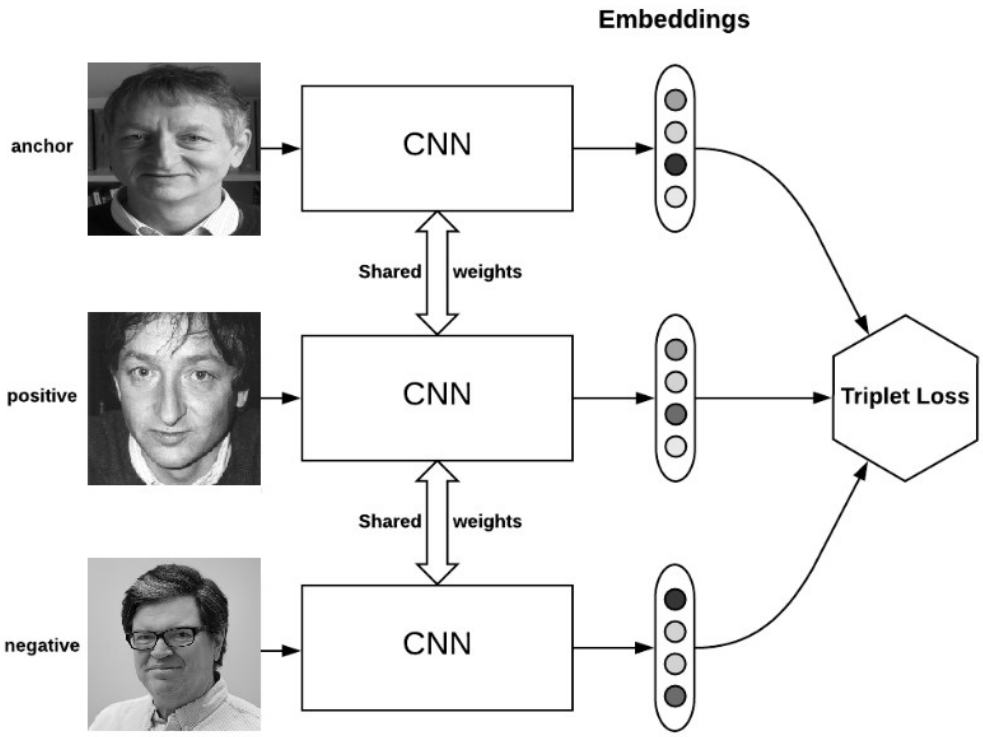
MTCNN sử dụng nhiều mạng CNN, các đặc trưng được học dựa trên bộ dữ liệu huấn luyện nên độ chính xác cao hơn nhưng vì phức tạp nên thời gian chạy sẽ chậm hơn.
- Để cải thiện độ chính xác ta có thể sử dụng một số kỹ thuật trong xử lý ảnh để tiền xử lý dữ liệu đầu vào. Ví dụ cân bằng sáng, tăng độ tương phản, làm rõ nét ảnh hơn.

III. Trích xuất đặc trưng ảnh sử dụng mạng Facenet

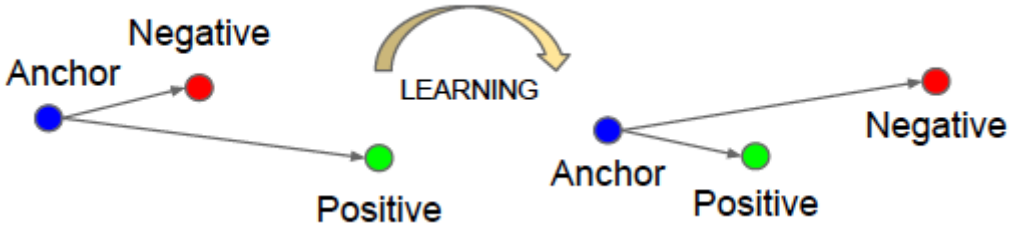
- Facenet là 1 mạng neural đa tầng được sử dụng để trích xuất đặc trưng từ bức ảnh mặt người.

- Input của facenet là 1 tấm ảnh chứa mặt người và output của nó là 1 vector chứa 128 giá trị (128 chiều), trong học máy (machine leanring) gọi là vector nhúng. Vector này chứa những thông tin quan trọng của tấm ảnh và đảm bảo các tấm ảnh tương tự nhau thì có vector nhúng cũng tương tự nhau.
- Điểm đặc biệt ở Facenet là hàm mất mát, thuật toán này sử dụng bộ 3 hàm mất mát (triplet loss function). Để tính toán bộ 3 hàm mất mát này ta cần 3 bức ảnh:

- Ảnh đang xét (anchor image) giả sử có nhãn là người A.
- Ảnh có cùng nhãn với ảnh đang xét (positivev image).
- Ảnh khác nhãn với ảnh đang xét (negative image).



- Chúng ta có thể hình dung, Facenet muốn các vector đặc trưng được phân bố theo các cụm, mỗi cụm là nhãn của 1 người. Chúng ta muốn vector của anchor image gần positive image hơn negative image và tất cả ảnh còn lại, (gần xa ở đây được tính là khoảng các của các vector trong không gian).



- Sau đây là quá trình mà Facenet thực hiện:

- B1: Chọn ngẫu nhiên anchor image
- B2: Chọn ngẫu nhiên ảnh có cùng nhãn với anchor image (positive example)
- B3: Chọn ngẫu nhiên ảnh khác nhãn với anchor image (negative example)
- B4: Điều chỉnh các tham số trong mạng Facenet sao cho vector của positive example gần anchor image hơn negative example).

****