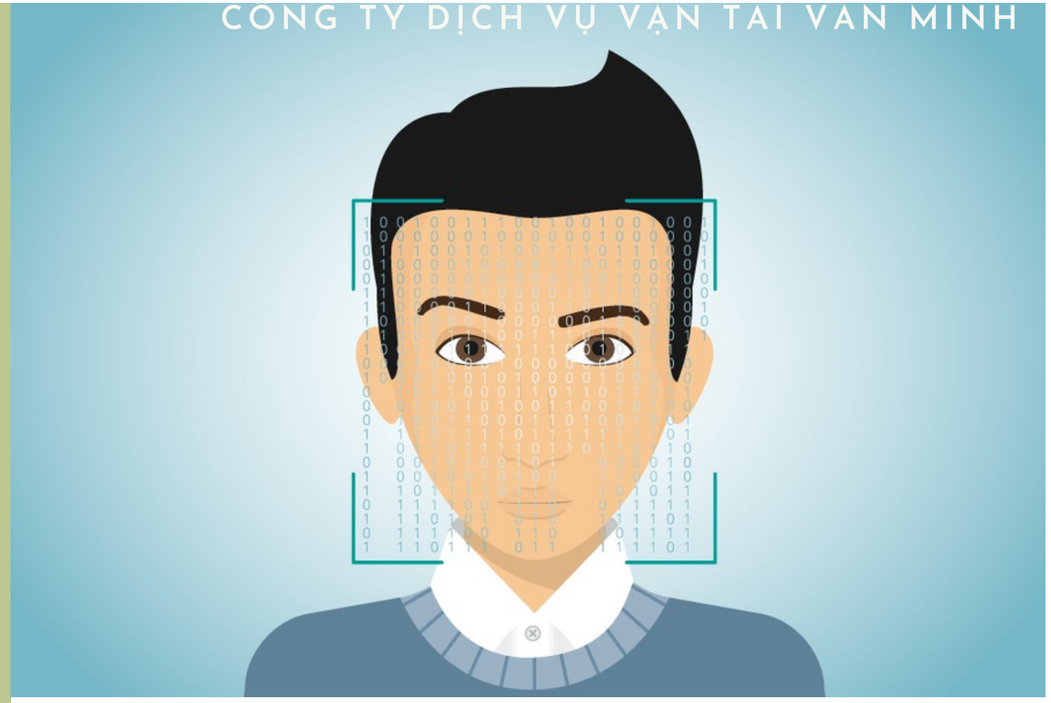
MTCNN FACE DETECTION



CHỦ ĐỀ ĐƯỢC THẢO LUẬN

Giới thiệu, Tổng quan

Stage 1 : P-net

Stage 2: R-Net

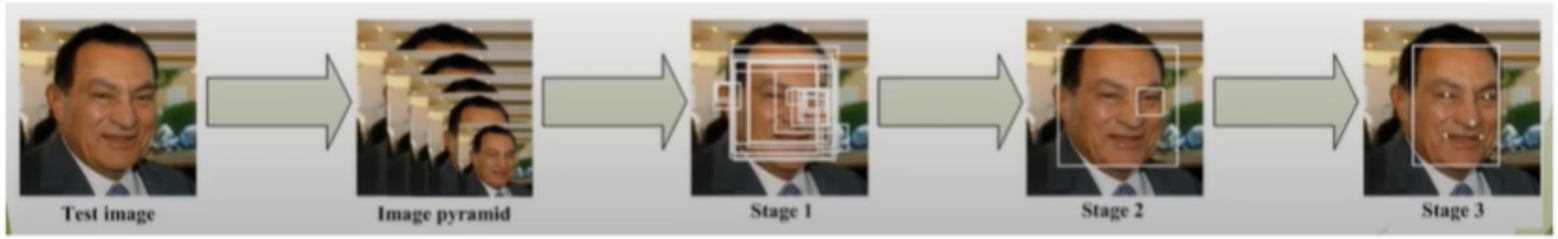
Stage 3: O-Net

Sử dụng MTCNN phát hiện khuôn mặt

Haar Cascade vs MTCNN

Tổng quan

MTCNN (Mullti-task Cascaded Convolutional Networks) là một mạng tích chập có với mục đích phát hiện các khuôn mặt trong ảnh hoặc frame trong video. Mạng MTCNN gồm 3 lớp mạng(stage) lần lượt là P-Net, R-Net, O-Net



- Resize ảnh, để tạo một loạt các bản copy từ ảnh gốc với kích cỡ khác nhau, từ to đến nhỏ, tạo thành Image pyramid
- Stage 1: ra một loạt các bounding boxes nằm trong bộ lọc
- Stage 2: loại bỏ những điểm giống nhau tạo ra những tọa độ mới của các box
- Stage 3: phát hiện khuôn mặt nằm trong bounding box, tọa độ của bounding box, và các mốc trên khuôn mặt(vị trí mắt, mũ và miệng)

Tổng quan

Image pyramid

Một bức ảnh, hay 1 frame trong video thường sẽ có nhiều khuôn mặt với các kích thước khác nhau=> yêu cần cần một method để nhận dạng toàn bộ các khuôn mặt ở mọi kích thước => Cần resize ảnh, tạo ra các bản copy từ ảnh gốc với kích cỡ từ to đến nhỏ (Image pyramid)

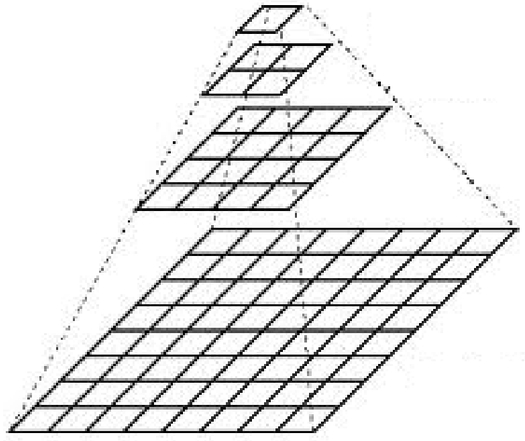


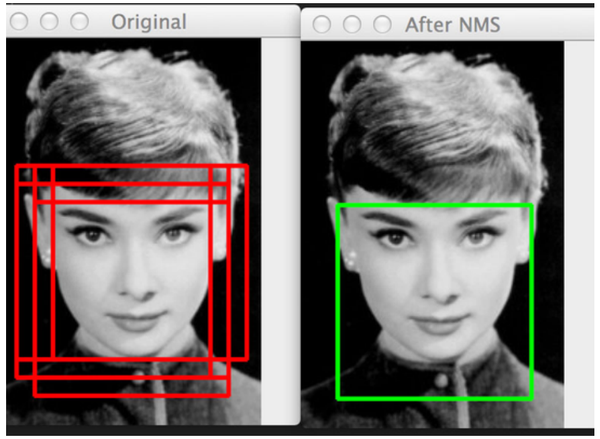
Image pyramid

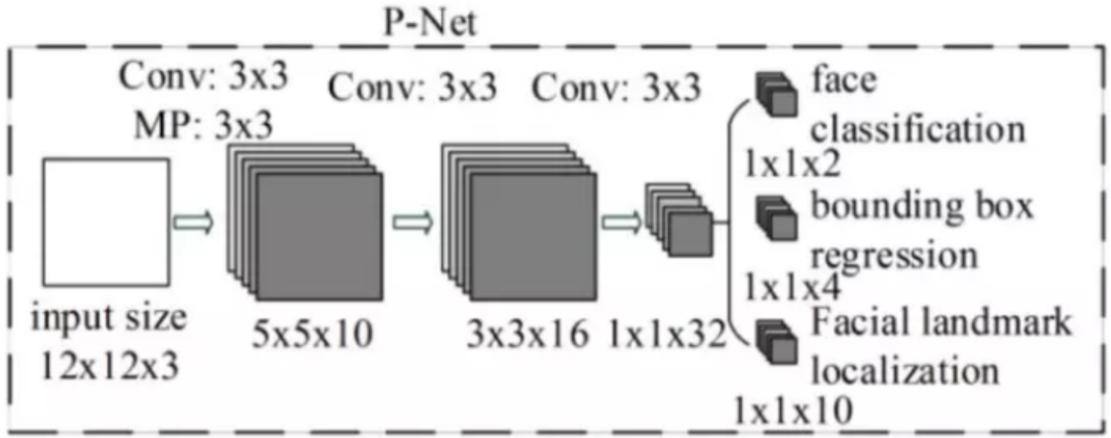
Trên mỗi một phiên bản ảnh đã resize từ ảnh gốc, sử dụng một kernel 12x12 pixel, stride =2 để duyệt qua ảnh và tìm mặt.=> với nhiều bản copy được resize khác nhau, dùng 1 kernel cố định thì được ảnh to thì mặt to, ảnh nhỏ thì mặt nhỏ

Tổng quan

Stage 1: P-Net:

- Dự đoán các vùng trong ảnh hoặc frame video có thể là khuôn mặt
- Tạo 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(normailize về khoảng từ (0,1))
- Lập mức threshold confident đẻ xáo đi cac box có mức confient thấp
- NMS(Non-maximum suppression): xoác các box có tỷ lệ trùng nhau vượt quá một threshold tự đặt.





P-Net (from MTCNN paper)

Tổng quan

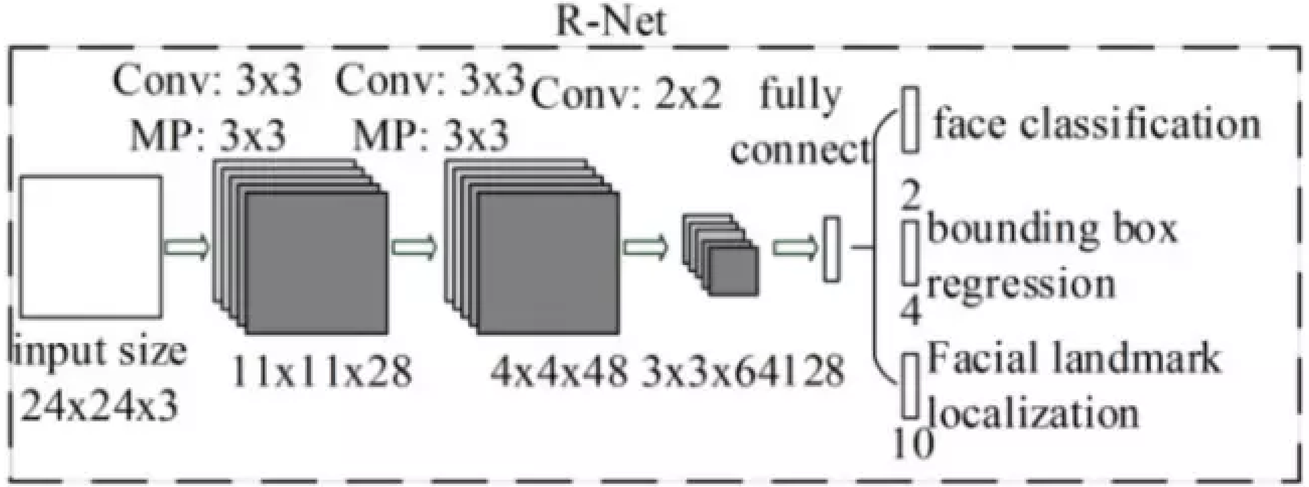
Stage 1: P-Net:

- Sau khi xóa các box thừa, chuyển tọa độ các box về với tọa độ gốc của ảnh thật. Khi này các tọa độ đang trong range (0.1) tương ứng với kernel => tính toán độ dài độ rộng của kernel ảnh gốc, rồi nhân tọa độ trong range 0-1 kia với kernel và cộng với tọa độ của các góc kernel tương ứng
- Vậy ta sẽ thu được tọa độ các box tương ứng với ảnh kích thước ban đầu, rồi resize lại các box về hình vuông, lấy tọa độ của các box rồi fit tiếp vào mạng tiếp theo

Tổng quan

Stage 2: R-NET:

- (Refind Network) thực hiện các bước như mạng P.
- Sử dụng thêm padding để chèn các điểm ảnh có giá trị 0 pixels vào các phần thiều của bouding box nếu bounding box bị vượt quá biên của ảnh.
- Các bounding box được resize về kích thước 24x24 pixels, được đánh giá coi như là 1 kernel và fit vào mạng R
- Ta thu được các tọa độ mới của các box và đưa vào mạng tiếp là là mạng O



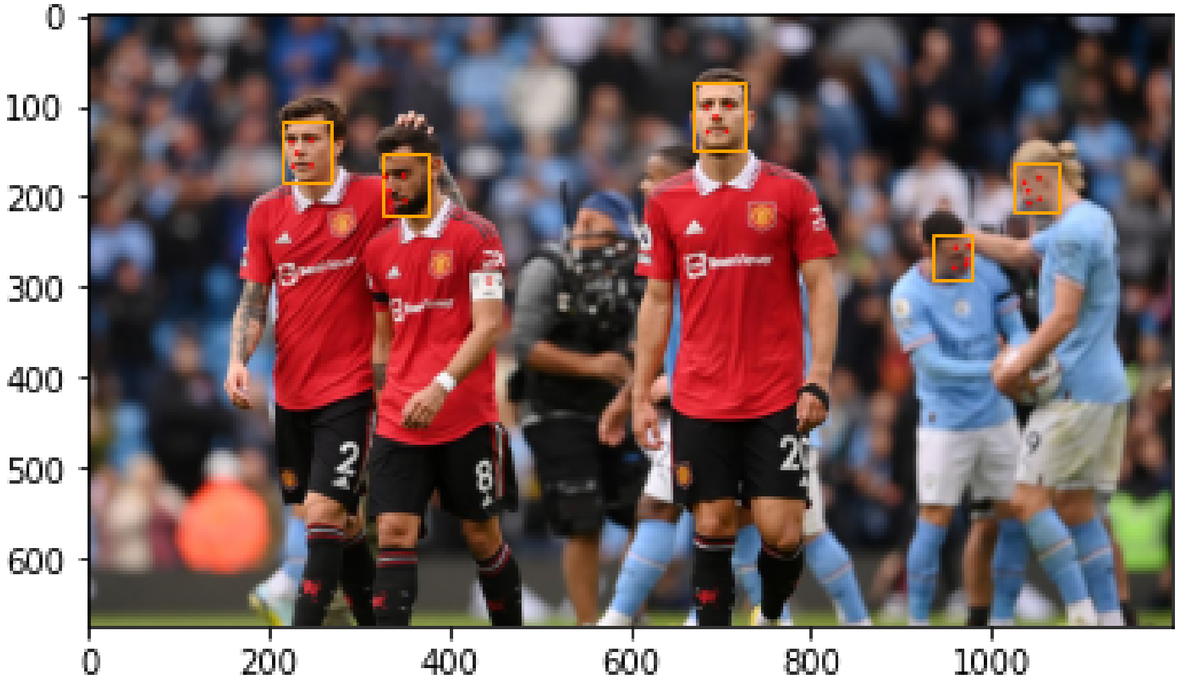
R-Net (from MTCNN paper)

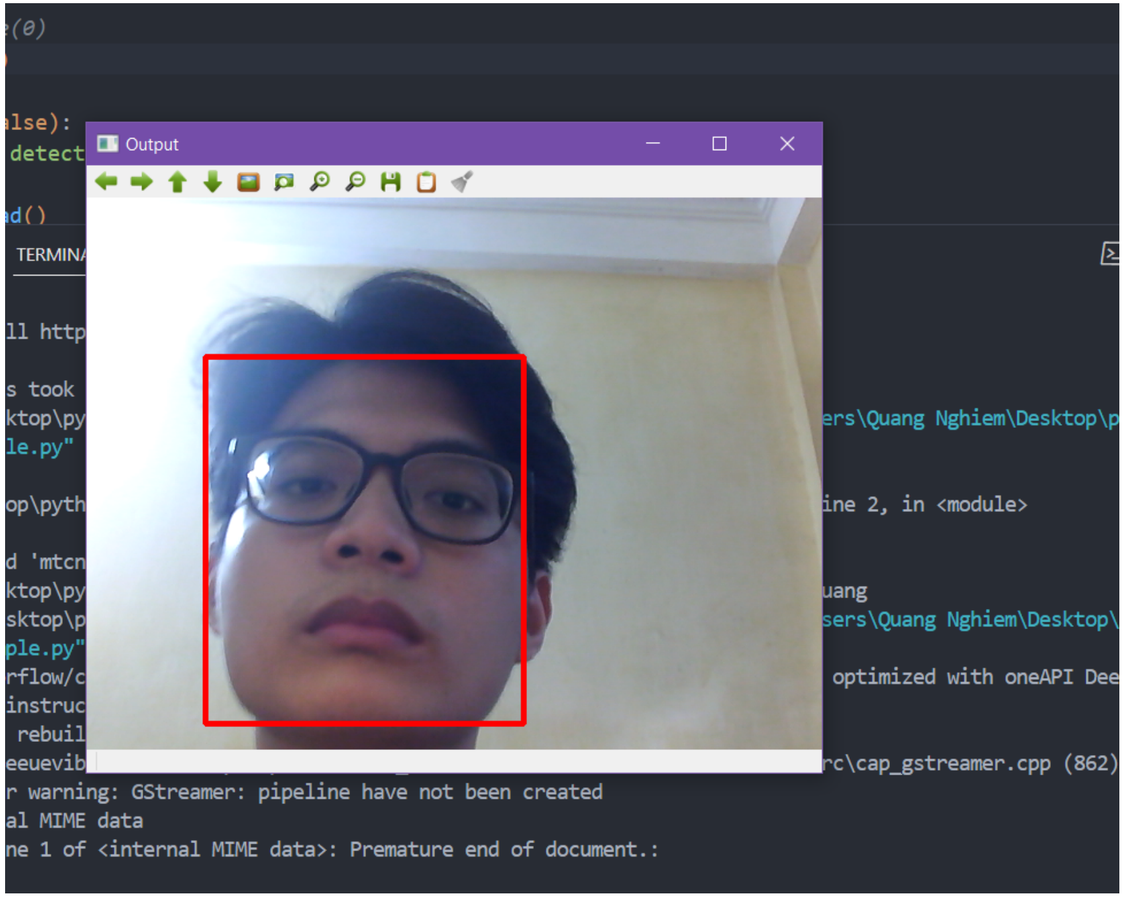
Tổng quan

Stage 1: O-NET:

- (Output Network), mạng cũng thực hiện như mạng R chỉ khác input đầu vào là ảnh 48x48.
- Output trả về 3 giá trị bao gồm: 4 tọa độ của các bounding box (out[0]), tạo độ 5 điểm landmark trên mặt, bao gồm 2 mắt, 1 mũi , 2 bên cánh miệng (out[1]) và điểm confident của mỗi box (out[2]). Tất cả được lưu trên 1 dict.

Nhận diện khuôn mặt với MTCNN





Haar Casade vs MTCNN

Haarcascade

- Number of images in UTK Face: 24,111
- Number of cropped faces using haarcascade: 19,915
- Total number of extra faces from a single image: 947

Recall = (19915 / 24111)*100 = 82.60%
Precision = (18968 / 19915)*100 = 95.24%

MTCNN

- Number of images in UTK Face: 24,111
- Number of cropped faces using MTCNN: 21,666
- Total number of extra faces from a single image: 428

Recall = (21666 / 24111)*100 = 89.85%
Precision = (21238/21666)*100 = 98.02%

Facial landmark

Facial landmarks được sử dụng để định vị và biểu diễn salient regions (các vùng nổi bật) của khuôn mặt như:

- Các mắt
- Lông mày (eyebrows)
- Mũi (nose)
- Miệng (mounth)
- Đường dưới của hàm (jawline)

Facial landmarks được ứng dụng rộng rãi trong face alignment, head pose estimation, face swapping (hoán đổi khuôn mặt), blink detection (phát hiện chớp mắt)...

facial landmark using python and dlib