**HÀ NỘI – 2023**

Mục lục

[I. Giới thiệu: 3](#_Toc138872471)

[II. Cơ sở lý thuyết và công nghệ sử dụng: 3](#_Toc138872472)

[2. Phát hiện khuôn mặt: 3](#_Toc138872473)

[a. Giới thiệu về thuật toán sử dụng: 4](#_Toc138872474)

[b. Tại sao lại là thuật toán này: 4](#_Toc138872475)

[3. Phân loại khuôn mặt người tốt/xấu: 5](#_Toc138872476)

[a. Giới thiệu mô hình sử dụng: 5](#_Toc138872477)

[b. Tại sao lại là EfficientNet: 6](#_Toc138872478)

[4. Thuật toán làm mờ: 7](#_Toc138872479)

[III. Thiết kế và triển khai ứng dụng: 7](#_Toc138872480)

[1. Xác định yêu cầu và chức năng của ứng dụng: 7](#_Toc138872481)

[2. Quá trình thu thập tập dữ liệu và xây dựng tập dữ liệu huấn luyện: 8](#_Toc138872482)

[3. Quá trình triển khai ứng dụng: 8](#_Toc138872483)

[IV. Kết quả và đánh giá: 8](#_Toc138872484)

[1. Đánh giá mô hình nhận diện khuôn mặt: 8](#_Toc138872485)

[2. Phân tích kết quả phân loại khuôn mặt người tốt/xấu: 9](#_Toc138872486)

[3. Đánh giá thuật toán làm mờ khuôn mặt: 10](#_Toc138872487)

[V. Kết luận: 11](#_Toc138872488)

[1. Về các hạn chế còn hiện hữu của ứng dụng: 11](#_Toc138872489)

[2. Lời kết: 12](#_Toc138872490)

[VI. Tài liệu tham khảo 12](#_Toc138872491)

# 

# Giới thiệu:

Phân vùng ảnh và phân loại ảnh là hai bài toán đặc trưng nhất trong lĩnh vực áp dụng Deep Learning vào thị giác máy tính. Những năm gần đây, vô số các thuật toán, công nghệ mới được ra đời và nâng cấp liên tục như Yolo đã ra đến v8, hay vô số các mạng học sâu được nâng cấp liên tục như Resnet, EfficientNet.

Và bài toán được áp dụng công nghệ này nhiều nhất trong những năm gần đây chính là bài toán nhận diện khuôn mặt. Vô số những công nghệ đã ra đời nhờ sự nâng cấp và cải thiện này như: Điểm danh bằng nhận diện khuôn mặt, face id trên các điện thoại thông minh, …

Mục tiêu của bài tập này là để áp dụng những kiến thức đã có về Deep Learning và thị giác máy tính để tạo ra một sản phẩm hoàn thiện và có các chức năng cơ bản nhất.

Làm mờ ảnh, làm mờ khuôn mặt một người nhất định vốn là một công đoạn tuy đơn giản nhưng tốn kha khá thời gian và công sức của người chỉnh sửa video, chỉnh sửa ảnh. Đặc biệt, nó được áp dụng khá nhiều trong lĩnh vực truyền thông, khi người ta muốn đưa lên một phóng sự về những người tù, những bài phỏng vấn tội phạm, người ta thường phải làm mờ khuôn mặt người này. Và để giảm thiểu thời gian cũng như công sức của công đoạn này thì tôi đã nảy ra ý tưởng để làm một chương trình mà người dùng chỉ cần tải video lên và chương trình sẽ tự động nhân diện khuôn mặt có trong video, phân loại nó xem liệu đó là người tốt hay xấu và làm mờ khuôn mặt của những người được phân loại là người xấu.

Và đây sẽ là tài liệu báo cáo về quá trình cũng như những điều cơ bản có ở trong chương trình này như thuật toán, cấu trúc, … Trong quá trình thực hiện và triển khai ý tưởng sẽ không tránh khỏi sai sót, tôi mong sẽ nhận được sự góp ý và đánh giá của thầy. Tôi xin chân thành cảm ơn!

# Cơ sở lý thuyết và công nghệ sử dụng:

## Phát hiện khuôn mặt:

### Giới thiệu về thuật toán sử dụng:

Ở đây, tôi sẽ sử dụng một mô hình Caffe dựa trên thuật toán Single Shot-Multibox Detector (SSD) và sử dụng mạng Restnet-10 làm backbone của nó. Đây là một mô hình được hỗ trợ và giới thiệu bởi OpenCV.

Mô hình cần có hai file đó là: file .prototxt để chứa cấu trúc mô hình và file .caffemodel để chứa các trọng số của mô hình này. Bạn có thể lấy file .prototxt này có thể tìm ở github <https://github.com/opencv/opencv/tree/master/samples/dnn/face_detector> do chính opencv triển khai.

Về cơ bản, một mạng để có thể phát hiện khuôn mặt gồm hai phần, một là để phân vùng, phát hiện ra các vật thể trên ảnh (SSD) và một phần là để có thể phân loại xem nó là vật gì (Resnet-10).Và rồi khi ta kết hợp hai phần này là thì ta có thể phân vùng khuôn mặt có trong các ảnh ta truyền vào.

### Tại sao lại là thuật toán này:

Sau một quá trình tìm hiểu và tham khảo, tôi đã tìm hiểu được một bài viết khá hay về phát hiện khuôn mặt (1). Trong bài viết này, tác giả đã thử nghiệm việc phát hiện khuôn mặt trên 4 thuật toán/mô hình: Haar, dlib, Multi-task Cascaded Convolutional Neural Network (MTCNN), và DNN module của OpenCV. Ở đây tôi sẽ không đi sâu vào từng thuật toán, mà sẽ chỉ trích dẫn lại những kết quả mà tác giả đã đạt được sau quá trình thử nghiệm cả 4 thuật toán này.

Trong bài viết, tác giả đã thử nghiệm triển khai 4 thuật toán đề cập ở 2 loại dữ liệu khác nhau là: hình ảnh vào video. Hiển nhiên ở đề tài này, ta chỉ quan tâm đến những kết quả về video.

Và để so sánh kết quả của các thuật toán này, ta sẽ cần so sánh nó ở những khía cạnh sau với kích cỡ truyền vào là 640x360 và với module DNN của Opencv là 300x300:

* Góc cạnh khác nhau của khuôn mặt, đầu di chuyển:

[https://miro.medium.com/v2/resize:fit:640/1\*vdelr5dnogvKCS\_1dCio\_A.gif](https://miro.medium.com/v2/resize:fit:640/1*vdelr5dnogvKCS_1dCio_A.gif)

Ở khía cạnh này, module dnn làm tốt hơn hẳn so với 3 model còn lại khi các model khác đều fail với những chuyển động đầu mạnh hay khi khuôn mặt quay những góc khó

* Khuôn mặt bị khuyết thiếu:

[https://miro.medium.com/v2/resize:fit:640/1\*-1DbF76CfRDLrGMP0MIhEQ.gif](https://miro.medium.com/v2/resize:fit:640/1*-1DbF76CfRDLrGMP0MIhEQ.gif)

Tương tự, module dnn vượt trội so với các module khác khi có thể nhận diện đến 601/642 frames ở trong video này trong khi các model khác lần lượt là 479 với Haar, MTCNN là 464 và dlib là 401

* Độ sáng khác nhau:

[https://miro.medium.com/v2/resize:fit:640/1\*TEI\_i-ivVFgjfTvZVWLRxQ.gif](https://miro.medium.com/v2/resize:fit:640/1*TEI_i-ivVFgjfTvZVWLRxQ.gif)

Trong khi Haar có thể phát hiện ở một vài frames khi ánh sáng tắt, thì DNN cho ra các kết quả dương tính giả, dlib tương tự, còn MTCNN thì kết quả khá đáng thất vọng.

* Frame rate đạt được:

Trong bài viết, tác giả sử dụng chip Intel i5 7th và đạt được kết quả như sau:

**Haar** — 9.25 fps

**Dlib** — 5.41 fps

**MTCNN** — 7.92 fps

**DNN module của OpenCV** — 12.95 fps

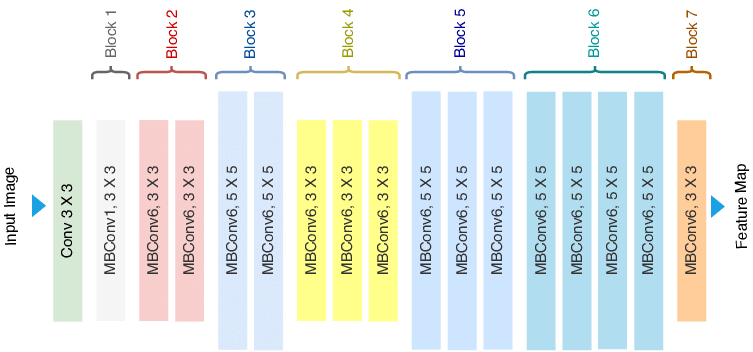
Như vậy, ta thấy rõ ràng module DNN của OpenCV chiếm ưu thế tuyệt đối ở 4/5 hạng mục, vậy nên ta sẽ chọn nó làm thuật toán để phát hiện khuôn mặt.

## Phân loại khuôn mặt người tốt/xấu:

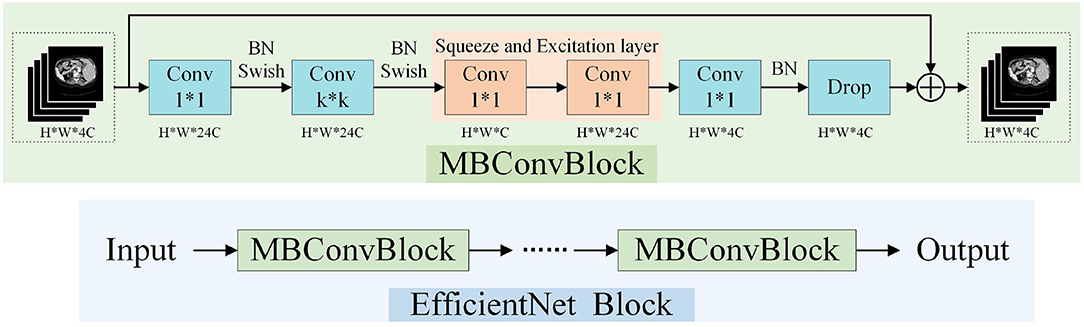
### Giới thiệu mô hình sử dụng:

Sau khi đã biết được và nhận về offset của bounding box chứa khuôn mặt, ta sẽ cần phải phân loại khuôn mặt đó là của người tốt hay xấu.

Để phân tích bài toán, ta thấy đây là một bài toán phân loại nhị phân, đầu vào là một ảnh, nên ta hoàn toàn có thể sử dụng các mô hình pretrain để có thể tăng độ chính xác cho kết quả. Ở đây, tôi quyết định lựa chọn mô hình EfficientNetv2B0 có cấu trúc như sau:



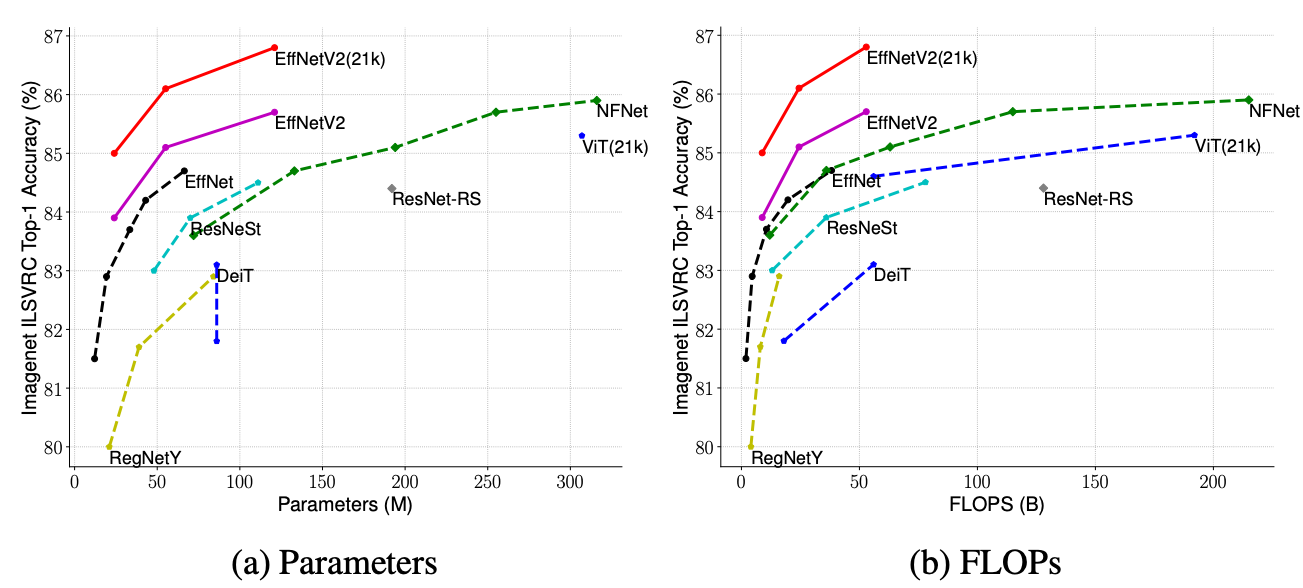
Nguồn: https://www.researchgate.net/figure/Architecture-of-EfficientNet-B0-with-MBConv-as-Basic-building-blocks\_fig4\_344410350



Nguồn: <https://www.frontiersin.org/articles/10.3389/fnins.2022.876065/full>

### Tại sao lại là EfficientNet:

Hiện nay, EfficientNet là một trong những mô hình mạnh mẽ nhất trong các mô hình học sâu, đặc biệt, số lượng param của nó không quá lớn như các mô hình khác, đúng với tiêu chí: mạnh hơn, nhanh hơn và nhỏ hơn.



Nguồn: <https://arxiv.org/pdf/2104.00298v3.pdf>

Từ hình bên trái, ta có thể dễ dàng thấy rằng, so với các mô hình đi trước EffNetV2 có số lượng params nhỏ nhưng độ chính xác vẫn được đảm bảo cực kì xuất sắc.

## Thuật toán làm mờ:

Trong đề tài này, tôi sử dụng thuật toán làm mờ pixelate thay cho các thuật toán làm mờ cơ bản có sẵn của OpenCV.

Ý tưởng của thuật toán này rất đơn giản, ta sẽ truyền vào nó một bức ảnh, rồi chia bức ảnh đấy ra thành các một lưới các khối vuống có kích thước hxh. Rồi trong mỗi ô vuông này, ta áp dụng thuật toán làm mờ mean blur để cho ra kết quả. Lý do sử dụng mean blur là do đây là thuật toán làm mờ có tốc độ nhanh nhất và kết quả cho ra tương đối tốt trong phạm vi đề tài trong 3 thuật toán làm mờ có sẵn của openCV là mean blur, gaussian blur và meadian blur.

# Thiết kế và triển khai ứng dụng:

## 1. Xác định yêu cầu và chức năng của ứng dụng:

Đầu vào: Một video của người dùng

Đầu ra: Một video với khuôn mặt của người xấu bị che mờ

Các chức năng chính:

\_ Nút Browse, khi nhấn vào sẽ cho phép người dùng tải lên Video và đưa vào xử lý

\_ Nhận diện khuôn mặt người

\_ Phân loại khuôn mặt tốt hay xấu

\_ Làm mờ khuôn mặt đã phân loại

\_ Sau khi đã làm mờ xong, đổi màn hình

\_ Chức năng hiển thị video đã làm mờ lên giao diện cho người dùng xem thử

\_ Nút Save, nhấn vào sẽ mở một cửa sổ cho phép người dùng lưu lại video đã làm mờ

\_ Nút Cancel, về lại màn hình chính mà không lưu lại video

## Quá trình thu thập tập dữ liệu và xây dựng tập dữ liệu huấn luyện:

Bộ dữ liệu huấn luyện được sử dụng để huấn luyện mô hình để phân biệt ảnh người tốt/xấu được lấy từ: <https://miai.vn/thu-vien-mi-ai/>

## Quá trình triển khai ứng dụng:

Xây dựng các module liên quan đến xử lý video gồm: mở video, lưu video, làm mờ khuôn mặt trong từng frame, nhận diện khuôn mặt trong từng frame, di chuyển video, xóa video (4 ngày)

Xây dựng giao diện người dùng: Các class Screen, Button, cài đặt chức năng cho từng screen, từng button (Save, Browse, Cancel), các cửa sổ thông báo, các chức năng liên quan đến logic của giao diện (7 ngày)

Xây dựng mô hình phân loại người tốt/xấu: Tìm dữ liệu, lựa chọn mô hình, tiền xử lý dữ liệu, train mô hình, kiểm thử/đánh giá mô hình, lưu trọng số của mô hình, đưa mô hình vào sản phẩm (7 ngày)

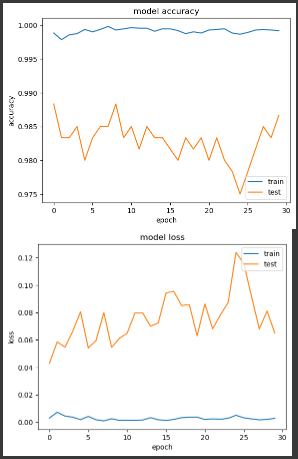
# Kết quả và đánh giá:

## 1. Đánh giá mô hình nhận diện khuôn mặt:

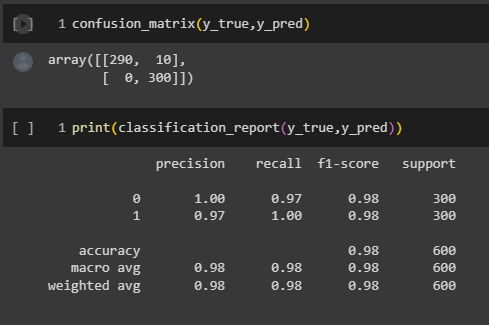
Như đã nêu ở mục II.1.b, mô hình hoạt động khá tốt với đa số các yêu cầu chính ở một mô hình nhân diện khuôn mặt, và mô hình có cách triển khai khá dễ dàng, đạt hiệu quả cao sau khi triển khai trong chương trình, tuy nhiên tốc độ còn chưa quá cao.

## Phân tích kết quả phân loại khuôn mặt người tốt/xấu:

Mô hình đạt độ chính xác tương đối cao, lên đến 100% ở tập train, 98.67% ở tập valid và 98.33% ở tập test, một kết quả vượt ngoài mong đợi.



Nhìn vào loss và accuracy của mô hình, ta thấy mô hình luôn dao động trong một khoảng accuracy và loss khá tốt ở cả tập train và test khi accuracy thấp nhất cũng chỉ tiệm cận 97.5%

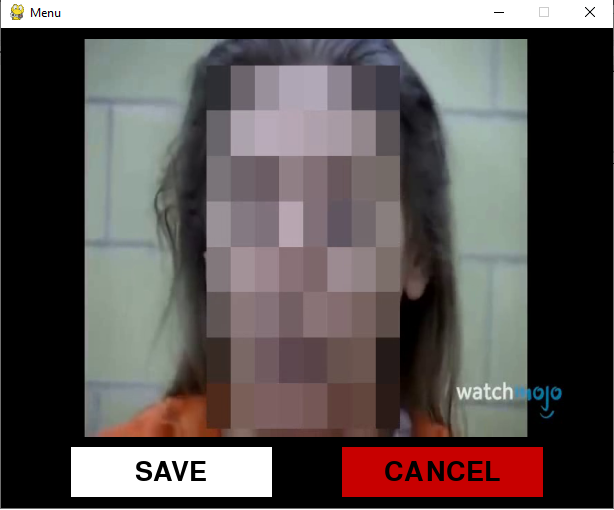


Nhìn vào confusion matrix (với nhãn 0: người tốt và nhãn 1: người xấu) ta thấy rằng không có giá trị nào là 1 mà bị đoán nhầm 0, đây là một tín hiệu tốt, vì ta có thể làm mờ nhầm khuôn mặt của những người tốt, chứ không nên nhầm lẫn khuôn mặt kẻ xấu là người tốt mà bỏ qua việc làm mờ.

Tuy nhiên, việc kết hợp cả nhận diện khuôn mặt và đưa khuôn mặt ấy qua một mô hình mạng neural khác làm cho thời gian tính toán tăng lên tương đối nhiều

## Đánh giá thuật toán làm mờ khuôn mặt:

Thuật toán hoạt động rất tốt, tốc độ khá nhanh và đưa ra kết quả đúng như mong đợi:



# Kết luận:

## Về các hạn chế còn hiện hữu của ứng dụng:

Ta có thể dễ dàng nhận ra trong quá trình sử dụng đó là tốc độ chạy của chương trình đã còn khá chậm, 1 video 10 giây thôi nhưng thời gian để xử lý có thể tốn đến 4-5 phút, thời gian này với 1 video 1 phút có thể lên tới 30-40 phút, và thường các video phỏng vấn thì sẽ không có video nào dưới 15 phút cả.

Nhược điểm thứ 2 của chương trình, đó là video output được đưa ra sẽ bị mặc định là 30fps. Điều này được đặt mặc định để cải thiện thời gian thực thi của chương trình.

Nhược điểm thứ 3 là chương trình xử lý multi threading chưa được mượt mà, còn đang gặp tình trạng bị đơ trong lúc xử lý video.

Cuối cùng, chương trình có vẻ còn gặp một số vấn đề về việc xử lý đường dẫn, khi ta chuyển file qua .pyw hay .exe thì video có vấn đề với việc lưu và mở video sau khi xử lý xong.

Và cuối cùng cũng như là quan trọng nhất, đó là ở trong tập dữ liệu vẫn còn một số sự thiên vị nhất định như sau:

\_ Cảm xúc: Đa số các ảnh trong tập dữ liệu gốc, người tốt thường hay cười khi được chụp ảnh. Tác giả đã sửa bằng cách thay thế các ảnh người tốt cười bằng ảnh với nhưng cảm xúc trung tính

\_ Kính: Nhiều người trong tập dữ liệu về người tốt đeo kính, nên tác giả cũng sửa bằng cách thay thế bằng các ảnh cảm xúc trung tính, nhưng vẫn để lại một vài người đeo kính

\_ Râu, tóc: Người có tóc dài và nhiều râu có tỷ trọng lớn trong tập dữ liệu người xấu

## Lời kết:

Nhìn chung, đề tài này đã được thực hiện một cách khá thành công và đạt được sự kỳ vọng của tác giả. Các lỗi, các vấn đề còn đang gặp phải không phải là vấn đề quá lớn, cũng như không phải là vấn đề có thể giải quyết được ngay với kiến thức của tác giả. Tuy nhiên, đề tài vẫn đạt được các tiêu chí là có một giao diện người dùng hoàn chỉnh, có sự kết hợp giữa các kiến thức trong lĩnh vực thị giác máy tính và học sâu, kết hợp các kiến thức về thị giác máy tính và các mạng nơ ron để hoàn thiện đề tài.

Tuy nhiên, vẫn có những đề xuất để cải tiến từ tác giả như: cải thiện tốc độ chạy của chương trình, để xử lý vấn đề này, ta có thể deploy chương trình lên web; xuất chương trình ra file exe để gọn nhẹ hơn; cải thiện phần multi threading trong chương trình, tránh tình trạng đơ khi xử lý những task lớn.

# Tài liệu tham khảo

1. <https://towardsdatascience.com/face-detection-models-which-to-use-and-why-d263e82c302c>
2. <https://keras.io/api/applications/>
3. <https://www.miai.vn/2021/12/03/phat-hien-va-lam-mo-mat-tu-dong-nhu-japan-av/>
4. <https://www.researchgate.net/figure/Architecture-of-EfficientNet-B0-with-MBConv-as-Basic-building-blocks_fig4_344410350>
5. <https://arxiv.org/pdf/2104.00298v3.pdf>
6. <https://www.frontiersin.org/articles/10.3389/fnins.2022.876065/full>
7. https://pyimagesearch.com/2017/09/11/object-detection-with-deep-learning-and-opencv/?fbclid=IwAR0A3p1trBTm8i7N5OcSQbEPWa0bXxq\_mejMPf1my05MNy\_pAN0-xjJe1lQ