

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO ĐỒ ÁN MÔN HỌC

**Đề tài: Xây dựng hệ thống nhận dạng khuôn mặt
và dự đoán tuổi con người**

Lớp : 136805

Học phần : Nhập môn Học máy và Khai phá dữ liệu

Mã học phần : IT3190

Giảng viên hướng dẫn : TS. Nguyễn Nhật Quang

Danh sách thành viên nhóm:

Họ và tên	Mã số sinh viên
Phạm Đức Hào	20200200
Trương Văn Hiền	20194276
Đinh Trọng Nghĩa	20194340
Phạm Phương Huy	20194300

Hà Nội, tháng 1 năm 2023

MỤC LỤC

LỜI NÓI ĐẦU.....	4
CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI	5
1.1. Lý do chọn đề tài	5
1.2. Yêu cầu bài toán.....	5
1.3. Ý tưởng thực hiện.....	5
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT.....	7
2.1. Bài toán 1: Phát hiện toạ độ khuôn mặt trong ảnh, video.....	7
2.1.1. Đặc trưng Haar Like	7
2.1.2. Thuật toán Adaboost	9
2.1.3. Mô hình phân tầng Cascade	12
2.2. Bài toán 2: Dự đoán độ tuổi trên khuôn mặt.....	13
2.2.1. Convolutional.....	13
2.2.2. Cấu trúc mạng CNN	14
CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG VÀ BỘ DỮ LIỆU.....	16
3.1. Công nghệ sử dụng.....	16
3.2. Bộ dữ liệu	16
CHƯƠNG 4. XÂY DỰNG CHƯƠNG TRÌNH VÀ TRIỂN KHAI CÀI ĐẶT	17
3.1. Xây dựng chương trình Training data	17
3.1.1. Các thư viện sử dụng.....	17
3.1.2. Xây dựng dữ liệu	17
3.1.3. Loading Images.....	18
3.1.4. Tạo Model.....	20

3.1.5. Train Model.....	21
3.2. Face_detection.py	22
3.3. Triển khai cài đặt	22
CHƯƠNG 4. KẾT QUẢ TỔNG QUAN	24
4.1. Đánh giá mô hình	24
4.2. Sản phẩm demo	24
4.2.1. Kiểm thử tập dữ liệu test	24
4.2.2. Demo app Age Prediction realtime.....	25
4.3. Nhận xét, đánh giá	26
4.4. Hướng phát triển	26
TÀI LIỆU THAM KHẢO.....	27

LỜI NÓI ĐẦU

Công nghệ thông tin ngày càng phát triển và có vai trò hết sức quan trọng không thể thiếu trong cuộc sống hiện đại. Trong thời đại 4.0, con người ngày càng tạo ra những cỗ máy thông minh có khả năng tự nhận biết và xử lý được các công việc một cách tự động, phục vụ cho lợi ích của con người. Trong những năm gần đây, một trong những bài toán nhận được nhiều sự quan tâm và tốn nhiều công sức nhất của lĩnh vực công nghệ thông tin, đó chính là bài toán nhận dạng.

Tuy mới xuất hiện chưa lâu nhưng nó đã rất được quan tâm vì tính ứng dụng thực tế của bài toán cũng như sự phức tạp của nó. Bài toán nhận dạng có rất nhiều lĩnh vực như: nhận dạng vật chất, nhận dạng chữ viết, nhận dạng giọng nói, nhận dạng khuôn mặt ... trong đó phổ biến và có tính ứng dụng nhiều hơn cả là bài toán nhận diện khuôn mặt. Để nhận dạng được khuôn mặt, bước đầu tiên để nhận dạng là phát hiện ra khuôn mặt, sau đó là nhận dạng, phân loại khuôn mặt.

Với sự hấp dẫn của bài toán và những thách thức còn đang ở phía trước, với niềm đam mê, mong muốn được học hỏi các công nghệ, tiếp xúc với bài toán nhận dạng, nhóm chúng em đã quyết định lựa chọn đề tài “Xây dựng hệ thống nhận dạng khuôn mặt và dự đoán tuổi con người” cho đồ án môn học của mình. Nhóm chúng em mong muốn có thể triển khai được một mô hình đáp ứng được tiêu chuẩn tốt, nhanh để phù hợp cho tính ứng dụng của nó.

Đồ án của nhóm chúng em bao gồm 4 nội dung chính:

- Tổng quan đề tài
- Cơ sở lý thuyết
- Công nghệ sử dụng và bộ dữ liệu
- Xây dựng chương trình và triển khai cài đặt
- Kết quả tổng quan

Mặc dù đã cố gắng hoàn thiện sản phẩm nhưng không thể tránh khỏi những thiếu hụt về kiến thức và sai sót trong kiểm thử. Chúng em rất mong nhận được những nhận xét thẳng thắn, chi tiết đến từ thầy để tiếp tục hoàn thiện hơn nữa. Cuối cùng, nhóm chúng em xin được gửi lời cảm ơn đến thầy TS. Nguyễn Nhật Quang đã hướng dẫn chúng em trong suốt quá trình hoàn thiện Đồ án môn học. Nhóm chúng em xin chân thành cảm ơn thầy.

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI

1.1. Lý do chọn đề tài

Với sự phát triển không ngừng của khoa học và công nghệ, đặc biệt là với những chiếc điện thoại thông minh ngày càng hiện đại và được sử dụng phổ biến trong đời sống con người đã làm cho lượng thông tin thu được bằng hình ảnh ngày càng tăng. Theo đó, lĩnh vực xử lý ảnh cũng được chú trọng phát triển, ứng dụng rộng rãi trong đời sống xã hội hiện đại. Không chỉ dừng lại ở việc chỉnh sửa, tăng chất lượng hình ảnh mà với công nghệ xử lý ảnh hiện nay chúng ta có thể giải quyết các bài toán nhận dạng chữ viết, nhận dạng dấu vân tay, nhận dạng khuôn mặt...

Một trong những bài toán được nhiều người quan tâm nhất của lĩnh vực xử lý ảnh hiện nay đó là nhận dạng khuôn mặt (Face Recognition). Như chúng ta đã biết, khuôn mặt đóng vai trò quan trọng trong quá trình giao tiếp giữa người với người, nó mang một lượng thông tin giàu có, chẳng hạn như từ khuôn mặt chúng ta có thể xác định giới tính, tuổi tác, chủng tộc, trạng thái cảm xúc, đặc biệt là xác định mối quan hệ với đối tượng (có quen biết hay không). Do đó, bài toán nhận dạng khuôn mặt đóng vai trò quan trọng trong nhiều lĩnh vực đời sống hàng ngày của con người như các hệ thống giám sát, quản lý vào ra, tìm kiếm thông tin một người nổi tiếng... đặc biệt là các vấn đề an ninh, bảo mật.

Trong khuôn khổ đồ án môn học, nhóm em rất mong muốn triển khai một mô hình nhận diện khuôn mặt có thể đáp ứng được tính thực tiễn yêu cầu độ chính xác tương đối như hệ thống gửi xe, hệ thống điểm danh... Vì vậy nhóm em đã lựa chọn đề tài “Xây dựng hệ thống nhận dạng khuôn mặt và dự đoán tuổi con người” để có thể tìm hiểu sâu hơn và hiểu hơn về bài toán.

1.2. Yêu cầu bài toán

- Phát hiện đúng khuôn mặt có trong ảnh, video.
- Mô hình đạt được tỉ lệ chính xác cao, tối thiểu sự sai số về độ tuổi giúp người dùng tin tưởng để sử dụng.
- Đảm bảo sự mượt mà khi chạy real-time với webcam.

1.3. Ý tưởng thực hiện

Nhóm chúng em chia nhỏ hệ thống thành 2 bài toán cần giải quyết:

- Bài toán 1: Phát hiện tọa độ khuôn mặt trong ảnh, video.

Bài tập lớn: Nhập môn Học máy và Khai phá dữ liệu

- Bài toán 2: Sau khi đã xác định được khuôn mặt, dự đoán độ tuổi trên khuôn mặt đó.

Ý tưởng về giải pháp cho từng bài toán con:

- Bài toán 1: Nhận dạng khuôn mặt người trong ảnh, video bằng bộ phân loại Haar Cascade.
- Bài toán 2: Xây dựng một mô hình mạng CNN để dự đoán độ tuổi cho input đầu vào.

Tập dữ liệu sử dụng cho việc huấn luyện mạng CNN là [Age prediction | Kaggle](#)

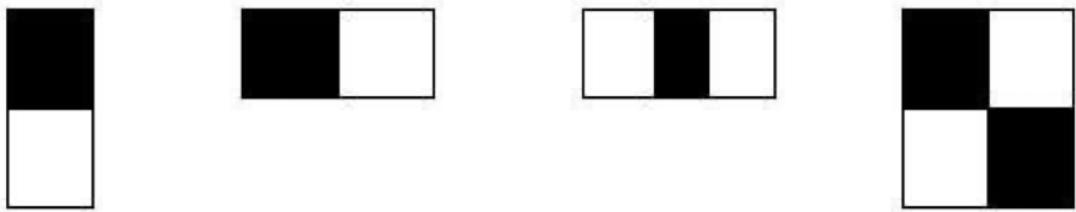
CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Bài toán 1: Phát hiện tọa độ khuôn mặt trong ảnh, video

Để có thể phát hiện tọa độ khuôn mặt người có trong ảnh hoặc video, chúng em sử dụng bộ phân loại Haar Cascade. Bộ phân loại Haar Cascade là một hướng tiếp cận hiệu quả cho bài toán nhận diện vật thể, được đề xuất trong bài báo “Rapid Object Detection using a Boosted Cascade of Simple Features” (2001) bởi Paul Viola và Michael Jones. Bộ phân loại Haar, được sử dụng trong bộ nhận diện khuôn mặt thời gian thực đầu tiên, thực chất là một hệ thống học máy mà hàm cascade được huấn luyện với rất nhiều ảnh, gồm cả ảnh dương bản và ảnh âm bản. Sau huấn luyện, hệ thống được sử dụng để nhận diện vật thể trong những hình ảnh khác. Bộ phân loại này về cơ bản là sử dụng các đặc trưng Haar Like và sau đó sử dụng thật nhiều đặc trưng đó qua nhiều lượt (Cascade) để tạo thành một cỗ máy nhận diện hoàn chỉnh.

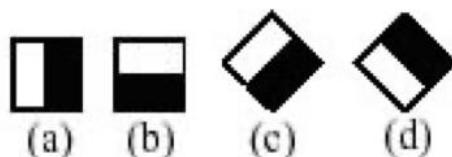
2.1.1. Đặc trưng Haar Like

Khuôn mặt được đặc trưng bởi tập hợp các pixel trong vùng khuôn mặt mà các pixel này tạo lên những điểm khác biệt so với các vùng pixel khác. Tuy nhiên với một ảnh đầu vào, việc sử dụng các pixel riêng lẻ lại không hiệu quả. Vì vậy những nhà nghiên cứu đã đưa ra tư tưởng kết hợp các vùng pixel với nhau tạo đặc trưng có khả năng phân loại tốt các vùng của khuôn mặt. Trong số đó đặc trưng Haar Like đã được ứng dụng. Mỗi đặc trưng Haar Like là một miền hình chữ nhật được chia thành 2, 3 hoặc 4 hình chữ nhật nhỏ phân biệt quy ước bằng màu trắng và màu đen như hình vẽ dưới đây:

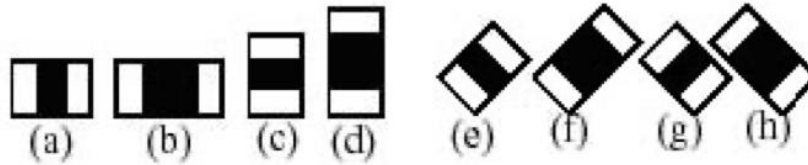


Từ 4 đặc trưng cơ bản mở rộng ra thành tập các đặc trưng:

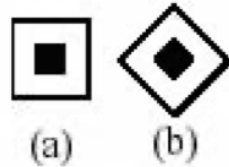
- Đặc trưng cạnh



- Đặc trưng đường



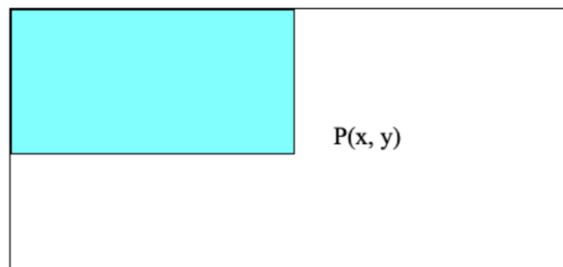
- Đặc trưng tâm - xung quanh



Giá trị của 1 đặc trưng Haar Like:

$$f(x) = \sum_{\text{vùng đen}} (pixel) - \sum_{\text{vùng trắng}} (pixel)$$

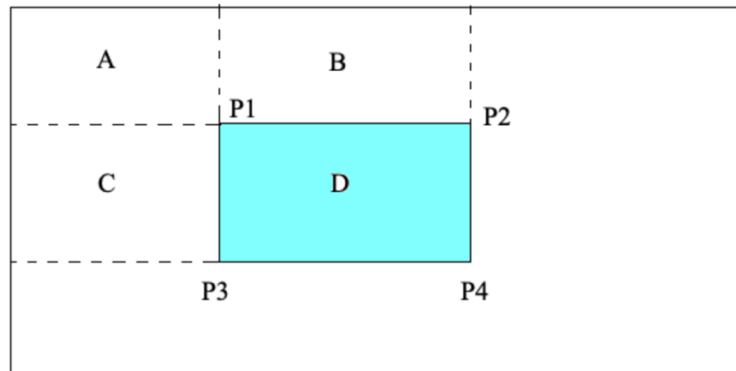
Để tính giá trị đặc trưng Haar Like, ta phải tính tổng của các vùng pixel trên ảnh. Nhưng để tính toán các giá trị của đặc trưng Haar Like cho tất cả các vị trí trên ảnh đòi hỏi chi phí tính toán khá lớn. Do đó để có thể tính nhanh, Viola và Jones giới thiệu khái niệm ảnh tích phân (Integral Image). Integral Image là một mảng 2 chiều với kích thước bằng kích thước của ảnh cần tính các đặc trưng Haar Like, với mỗi phần tử của mảng này được tính bằng cách tính tổng của điểm ảnh phía trên và bên trái của nó. Bắt đầu từ vị trí trên, bên trái đến vị trí dưới, phải của ảnh, việc tính toán này chỉ dựa trên phép cộng số nguyên đơn giản.



Giá trị của ảnh tích phân tại điểm P có tọa độ (x,y) được tính như sau:

$$P(x, y) = \sum_{\substack{x' \leq x \\ y \leq y'}} i(x', y')$$

Sau khi đã tính được ảnh tích phân, việc tính tổng điểm ảnh của một vùng bất kì nào đó trên ảnh được thực hiện như sau:



Ví dụ ta tính tổng điểm ảnh của vùng D:

Với: A, B, C, D là tổng giá trị các điểm ảnh trong từng vùng

P1, P2, P3, P4 là giá trị ảnh tích phân tại 4 đỉnh của D

Ta có: $P1 = A$

$$P2 = A + B$$

$$P3 = A + C$$

$$P4 = A + B + C + D$$

Vậy:

$$P1 + P4 - P2 - P3 = A + (A + B + C + D) - (A + B) - (A + C) = D$$

$$\Rightarrow \mathbf{D = P1 + P4 - P2 - P3}$$

Khi áp dụng vào tính toán các giá trị đặc trưng, ta thấy:

- Đặc trưng 2 hình chữ nhật (đặc trưng cạnh) được tính thông qua 6 giá trị điểm ảnh tích phân.
- Đặc trưng 3 hình chữ nhật (đặc trưng đường) và đặc trưng tâm – xung quanh được tính thông qua 8 giá trị điểm ảnh tích phân.
- Đặc trưng 4 hình chữ nhật (đặc trưng chéo) được tính thông qua 9 giá trị điểm ảnh tích phân.

Trong khi nếu tính dùng định nghĩa thì các giá trị cần tính toán lên tới hàng trăm.

Điều này làm tăng tốc độ xử lý một cách đáng kể.

Tiếp theo, ta sử dụng phương pháp học máy Adaboost để xây dựng bộ phân loại mạnh với độ chính xác cao.

2.1.2. Thuật toán Adaboost

Adaboost là một bộ phân loại phi tuyến phức dựa trên tiếp cận boosting được Freund và Schapzire đưa ra vào năm 1995. Adaboost hoạt động dựa trên nguyên tắc kết hợp tuyến tính các bộ phân loại yếu để tạo nên một bộ phân loại mạnh.

Là một cải tiến của tiếp cận boosting. Adaboost sử dụng khái niệm trọng số để đánh dấu các mẫu khó nhận dạng. Trong quá trình huấn luyện, cứ mỗi bộ phân loại yếu được xây dựng, thuật toán sẽ tiến hành cập nhật lại trọng số để chuẩn bị cho việc xây dựng bộ phân loại kế tiếp: tăng trọng số của các mẫu bị nhận dạng sai và giảm trọng số của các mẫu được nhận dạng đúng bởi các bộ phân loại yếu vừa xây dựng. Bằng cách này bộ phân loại sau có thể tập trung vào các mẫu mà các bộ phân loại trước nó làm chưa tốt. Sau cùng, các bộ phân loại yếu sẽ được kết hợp tùy theo mức độ tốt của chúng để tạo nên một bộ phân loại mạnh.

Biểu diễn bộ phân loại yếu:

$$h_k(x) = \begin{cases} 1: & p_k f_k < p_k \theta_k \\ 0: & p_k f_k \geq p_k \theta_k \end{cases}$$

Trong đó:

x : cửa sổ con cần xét

θ_k : ngưỡng

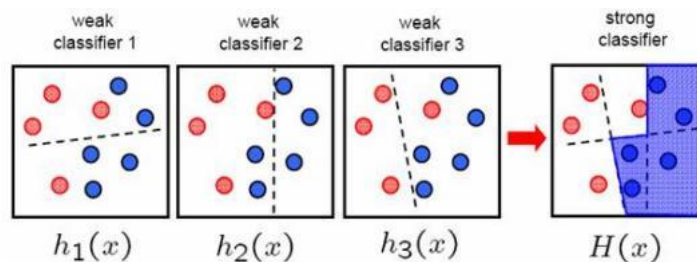
f_k : giá trị của đặc trưng Haar Like

p_k : hệ số quyết định chiều của phương trình

Adaboost sẽ kết hợp các bộ phân loại yếu thành bộ phân loại mạnh như sau:

$$H(x) = a_1 * h_1 + a_2 * h_2 + \dots + a_n * h_n = \sum_{i=1}^n a_i * h_i$$

Với $a_i \geq 0$: hệ số chuẩn hoá cho các bộ phân loại yếu



Thuật toán Adaboost:

- Cho một tập n các mẫu: $(x_1, y_1), \dots, (x_n, y_n)$ trong đó y_i là nhãn của mẫu x_i .
Trong bài toán của chúng em, x_i là các ảnh và $y_i \in \{0,1\}$ tương ứng cho ảnh x_i có chứa khuôn mặt người hay không.
- Khởi tạo các giá trị trọng số $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ tương ứng với $y_i = 0, 1$
trong đó: m là số ảnh không chứa mặt người (trường hợp âm - negative)
 l là số trường ảnh chứa mặt người (trường hợp dương - positive)
- Xây dựng T bộ phân loại yếu:

Lặp $t = 1, \dots, T$:

- Chuẩn hoá các trọng số để cho w_t là một phân phối xác suất:

$$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^n w_{t,j}}$$

- Với mỗi đặc trưng j , huấn luyện bộ phân loại h_j chỉ được hạn chế để sử dụng cho một đặc trưng đơn.

Sai số được đánh giá cho $w_{t,i}$:

$$\epsilon_t = \sum_i w_{t,i} |h_j(x_i) - y_i|$$

- Chọn bộ phân loại h_t với sai số ϵ_t nhỏ nhất.
- Cập nhật lại các trọng số:

$$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$

Trong đó: $e_i = \begin{cases} 0: & x_i \text{ được phân loại chính xác} \\ 1: & \text{ngược lại} \end{cases}$

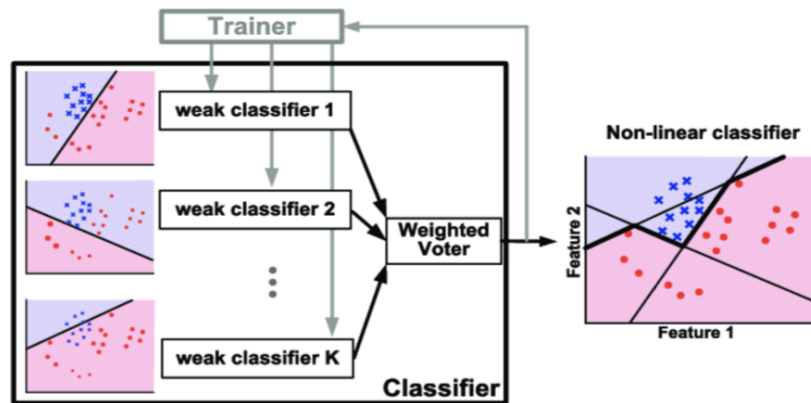
$$\beta_t = \frac{\epsilon}{1-\epsilon_t}$$

- Bộ phân loại mạnh cuối cùng là:

$$h_j(x) = \begin{cases} 1: & \sum_{t=1}^T \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \alpha_t \\ 0: & \text{ngược lại} \end{cases}$$

với $\alpha_t = \log \frac{1}{\beta_t}$

Minh hoạ thuật toán Adaboost:



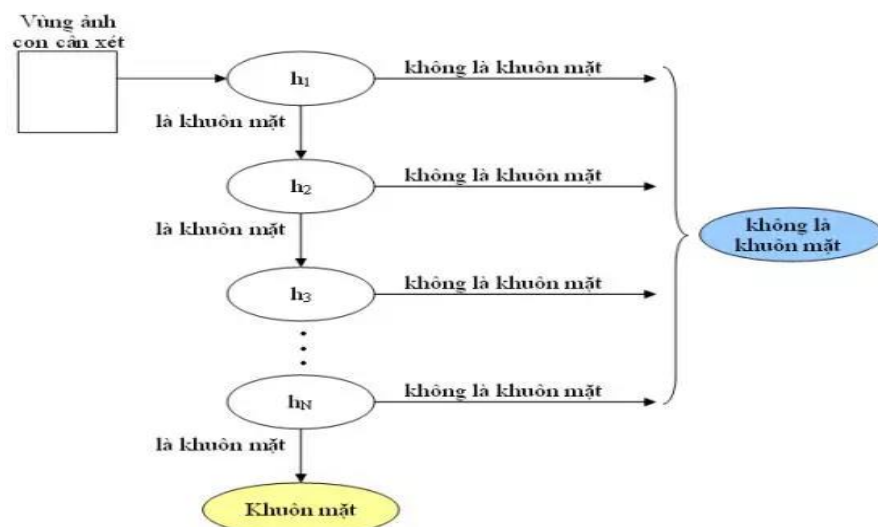
Các bộ lọc Haar Like kể cả sau Adaboost như trên vẫn chỉ bắt được những đặc trưng rất cơ bản, và để nhận ra một khuôn mặt thì chúng ta cần tầm 6000 các đặc trưng như vậy. Vậy chúng ta cần có một cách để xem cửa sổ đó có chứa mặt không, mà vẫn phải xử lý đủ nhanh cho cả 6000 đặc trưng đó, và để giải quyết vấn đề đó chúng em đã sử dụng mô hình phân tầng Cascade.

2.1.3. Mô hình phân tầng Cascade

Mô hình Cascade là mô hình phân tầng với mỗi tầng là một phân lớp được xây dựng bằng thuật toán Adaboost sử dụng bộ phân lớp yếu là cây quyết định với các đặc trưng Haar Like.

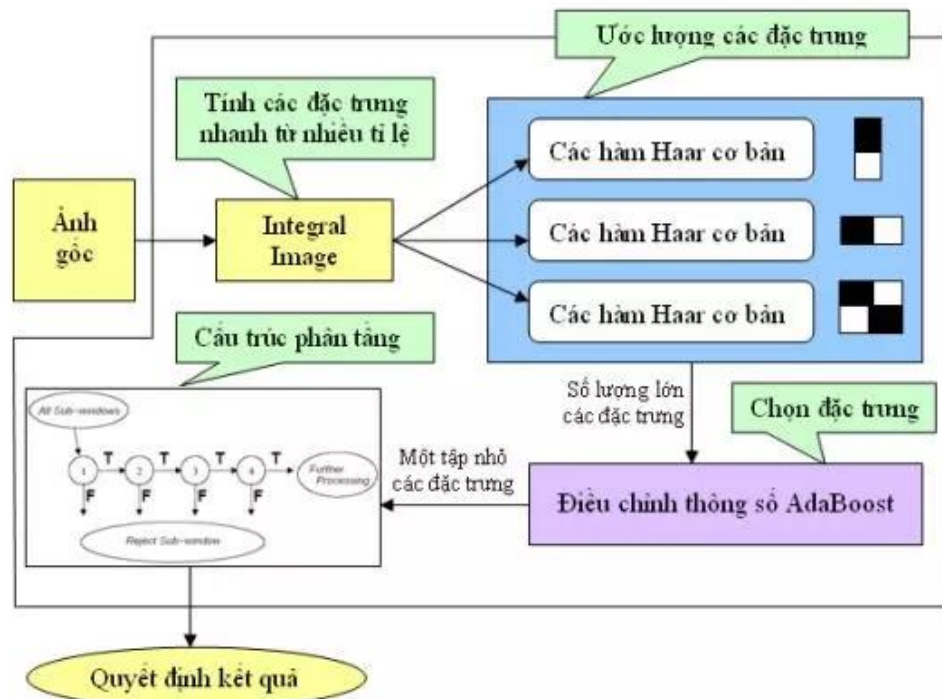
Bây giờ, ta đưa tất cả các cửa sổ con đi qua chuỗi các bộ phân lớp này:

- Bộ phân lớp đầu tiên sẽ loại bỏ phần lớn các ảnh không phải khuôn mặt và cho đi qua các ảnh được cho là khuôn mặt. Ở đây, bộ phân lớp này rất đơn giản và do đó, độ phức tạp tính toán cũng rất thấp. Tất nhiên, vì nó đơn giản nên trong số các ảnh được nhận dạng là khuôn mặt sẽ có một số lượng lớn ảnh bị nhận dạng sai (không phải khuôn mặt).
- Những ảnh 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. Ta gọi những ảnh 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 ảnh đi qua được tất cả các bộ phân lớp thì ta mới quyết định đó là khuôn mặt.



Tóm lại, chuỗi các bộ phân lớp sẽ xử lý các mẫu (ảnh) đi vào theo nguyên tắc sau: Nếu có một bộ phân lớp nào đó cho rằng đó không phải mặt người thì ta loại bỏ ngay, còn nếu bộ phân lớp đó cho rằng đó là khuôn mặt thì ta chuyển đến bộ phân lớp sau. Nếu một mẫu đi qua được hết tất cả các bộ phân lớp thì ta mới quyết định đó là khuôn mặt.

Sơ đồ nhận diện khuôn mặt:



2.2. Bài toán 2: Dự đoán độ tuổi trên khuôn mặt

Convolutional Neural Network (CNNs – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến. Nó giúp cho chúng ta xây dựng được những hệ thống thông minh với độ chính xác cao như hiện nay.

2.2.1. Convolutional

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.

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

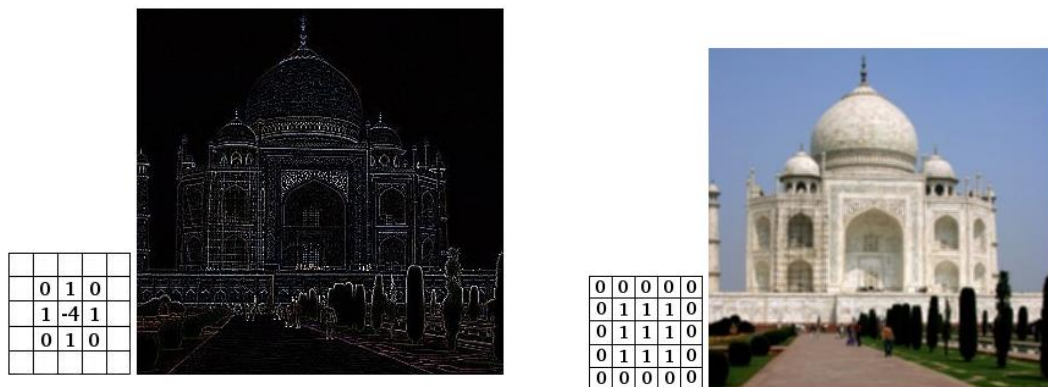
Image

4	3	4

Convolved Feature

Trong hình ảnh ví dụ trên, ma trận bên trái là một hình ảnh trắng đen được số hoá. Ma trận có kích thước 5x5 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 kernel, filter hoặc feature detect là một ma trận có kích thước nhỏ như trong ví dụ trên là 3x3.

Convolution hay tích chập là nhân từng phần tử bên trong ma trận 3x3 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 5x5 bên trái.



2.2.2. Cấu trúc mạng CNN

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.

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.

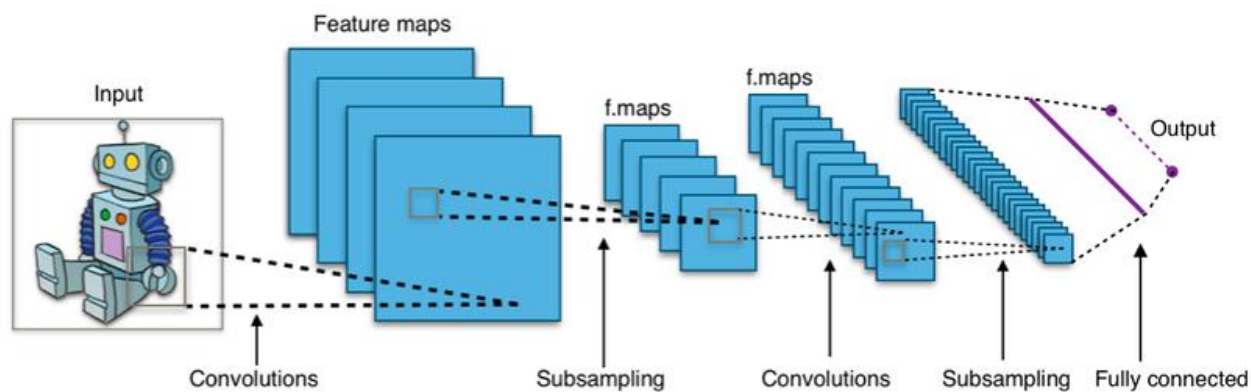
Mô hình này gọi là mạng kết nối đầy đủ (fully connected layer) hay mạng toàn vẹn (affine layer). Còn trong mô hình CNNs thì ngược lại. Các layer liên kết được với nhau thông qua cơ chế convolution.

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.



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.

Đó là lý do tại sao CNNs cho ra mô hình với độ chính xác rất cao. Cũng giống như cách con người nhận biết các vật thể trong tự nhiên.

CHƯƠNG 3. CÔNG NGHỆ SỬ DỤNG VÀ BỘ DỮ LIỆU

3.1. Công nghệ sử dụng

Môi trường: Anaconda.

Ngôn ngữ lập trình: Python 3+.

Các thư viện sử dụng:

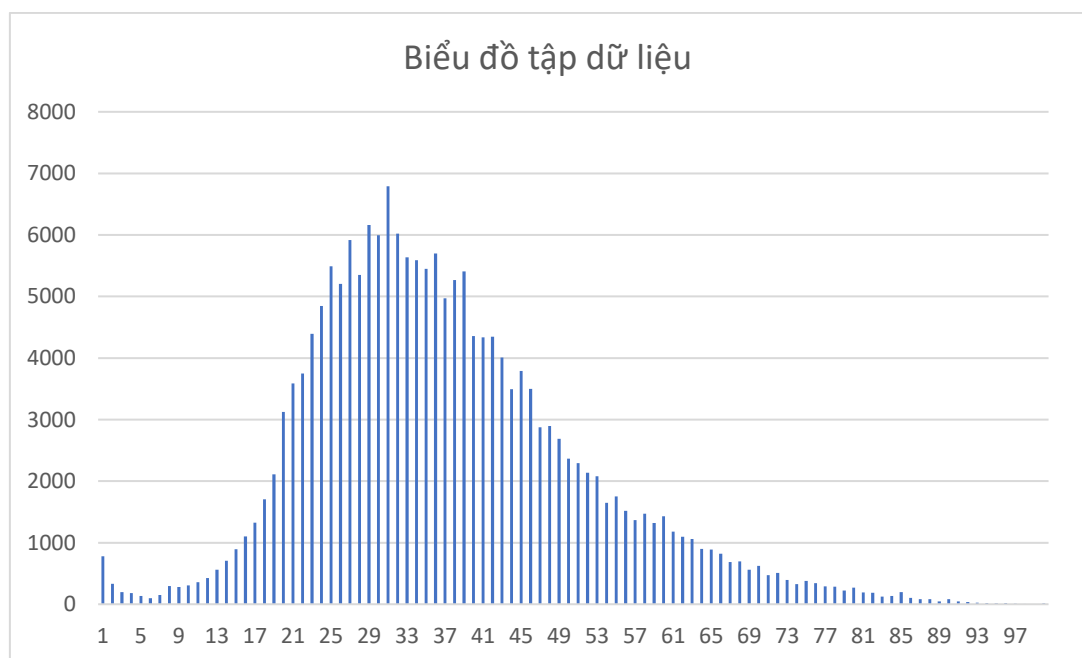
- OpenCV
- Tkinter
- Anaconda
- Numpy
- Pandas
- Sklearn
- TensorFlow
- Matplotlib

3.2. Bộ dữ liệu

Bộ dữ liệu sử dụng: [Age prediction | Kaggle](#)

Mô tả bộ dữ liệu:

- Tập dữ liệu sử dụng gồm hơn 185.000 bức ảnh chụp khuôn mặt người ở các độ tuổi khác nhau từ 0 – 100 tuổi. Kích thước các ảnh là 128x128 px.
- Tập dữ liệu được chia thành 2 phần “train” và “test” với tỉ lệ là 7:3.



CHƯƠNG 4. XÂY DỰNG CHƯƠNG TRÌNH VÀ TRIỂN KHAI CÀI ĐẶT

3.1. Xây dựng chương trình Training data

3.1.1. Các thư viện sử dụng

```
import numpy as np
import pandas as pd
from pathlib import Path
import os.path
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import tensorflow as tf
from sklearn.metrics import r2_score
```

Thư viện numpy: là một thư viện toán học phổ biến và mạnh mẽ của Python. Cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ sử dụng “core Python” đơn thuần.

Thư viện pandas: Thư viện pandas trong python là một thư viện mã nguồn mở, hỗ trợ đặc lực trong thao tác dữ liệu. Đây cũng là bộ công cụ phân tích và xử lý dữ liệu mạnh mẽ của ngôn ngữ lập trình python.

Thư viện pathlib: Tạo đường dẫn.

Thư viện matplotlib: Tạo biểu đồ từ dữ liệu train để có thể đánh giá một cách tốt hơn.

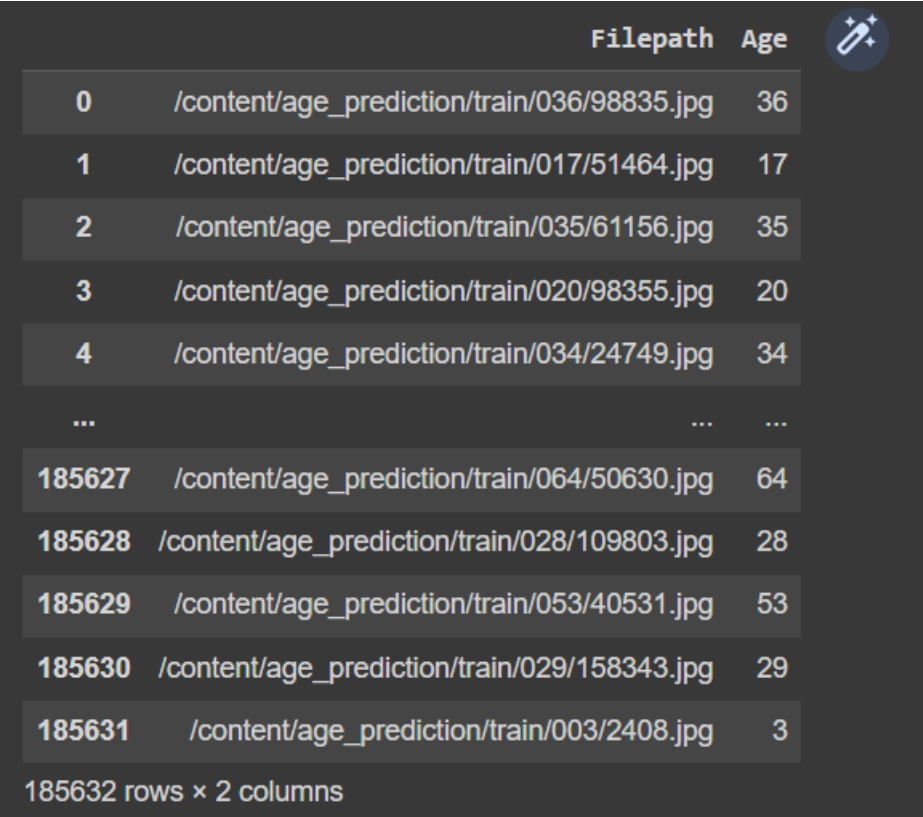
Thư viện sklearn (Scikit-learn): là thư viện mạnh mẽ nhất dành cho các thuật toán học máy được viết trên ngôn ngữ Python. Sử dụng để chia tập dữ liệu và đánh giá hiệu năng bằng điểm r^2 score.

Thư viện tensorflow: là thư viện mã nguồn mở cho machine learning, hỗ trợ mạnh mẽ các phép toán học để tính toán trong machine learning và deep learning giúp việc tiếp cận các bài toán trở nên đơn giản, nhanh chóng và tiện lợi hơn nhiều. Trong bài toán này, ta sử dụng keras trong tensorflow để xây dựng model và train.

3.1.2. Xây dựng dữ liệu

```
image_dir = Path('D:/chuyen_nganh/20221/machine_learning/project/Nhap_mom_ML-hao/Nhap_mom_ML-hao/age_prediction/train')
filepaths = pd.Series(list(image_dir.glob(r'*/*.jpg')), name='Filepath').astype(str)
ages = pd.Series(filepaths.apply(lambda x: os.path.split(os.path.split(x)[0])[1]), name='Age').astype(np.int)
images = pd.concat([filepaths, ages], axis=1).sample(frac=1.0, random_state=1).reset_index(drop=True)
train_df, test_df = train_test_split(images, train_size=0.7, shuffle=True, random_state=1)
```

Tập dữ liệu sẽ được xây dựng thành 2 cột chính là Filepath và Age.



	Filepath	Age
0	/content/age_prediction/train/036/98835.jpg	36
1	/content/age_prediction/train/017/51464.jpg	17
2	/content/age_prediction/train/035/61156.jpg	35
3	/content/age_prediction/train/020/98355.jpg	20
4	/content/age_prediction/train/034/24749.jpg	34
...
185627	/content/age_prediction/train/064/50630.jpg	64
185628	/content/age_prediction/train/028/109803.jpg	28
185629	/content/age_prediction/train/053/40531.jpg	53
185630	/content/age_prediction/train/029/158343.jpg	29
185631	/content/age_prediction/train/003/2408.jpg	3

185632 rows x 2 columns

Từ tập dữ liệu này sử dụng hàm `train_test_split` để chia dữ liệu thành 2 phần Train và Test tỉ lệ 7:3

3.1.3. Loading Images

Sử dụng `ImageDataGenerator` để tạo hàng loạt dữ liệu hình ảnh tensor với tính năng tăng cường dữ liệu theo thời gian thực.

Rescale $1./255$ để đưa dữ liệu về khoảng $[0:1]$

```
train_generator = tf.keras.preprocessing.image.ImageDataGenerator(  
    rescale=1./255,  
    validation_split=0.2  
)  
test_generator = tf.keras.preprocessing.image.ImageDataGenerator(  
    rescale=1./255  
)
```

Truyền luồng dữ liệu qua trình tạo ở trên:

```
train_images = train_generator.flow_from_dataframe(  
    dataframe=train_df,  
    x_col='Filepath',  
    y_col='Age',  
    target_size=(120, 120),  
    color_mode='rgb',  
    class_mode='raw',  
    batch_size=64,  
    shuffle=True,  
    seed=42,  
    subset='training'  
)
```

```
val_images = train_generator.flow_from_dataframe(  
    dataframe=train_df,  
    x_col='Filepath',  
    y_col='Age',  
    target_size=(120, 120),  
    color_mode='rgb',  
    class_mode='raw',  
    batch_size=64,  
    shuffle=True,  
    seed=42,  
    subset='validation'  
)
```

```
test_images = test_generator.flow_from_dataframe(  
    dataframe=test_df,  
    x_col='Filepath',  
    y_col='Age',  
    target_size=(120, 120),  
    color_mode='rgb',  
    class_mode='raw',  
    batch_size=64,  
    shuffle=False  
)
```

3.1.4. Tạo Model

```
inputs = tf.keras.Input(shape=(120, 120, 3))
x = tf.keras.layers.Conv2D(filters=16, kernel_size=(3, 3), activation='relu')(inputs)
x = tf.keras.layers.MaxPool2D()(x)
x = tf.keras.layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu')(x)
x = tf.keras.layers.MaxPool2D()(x)
x = tf.keras.layers.GlobalAveragePooling2D()(x)
x = tf.keras.layers.Dense(64, activation='relu')(x)
x = tf.keras.layers.Dense(64, activation='relu')(x)
outputs = tf.keras.layers.Dense(1, activation='linear')(x)

model = tf.keras.Model(inputs=inputs, outputs=outputs)
```

Đầu vào input là khối (120, 120, 3)

Lớp đầu tiên là lớp tích chập 2 chiều Conv2D với filter = 16 (16 lần chuyển toàn bộ hình ảnh), kích thước hạt nhân kernel_size = 3x3.

Sau đó, sử dụng Maxpool2D để giảm kích thước ảnh xuống ta được 16 tính năng 59x59.

Tương tự ta lại sử dụng Conv2D với filter = 32 và Maxpool2D để có được 32 tính năng 28x28.

Tiếp theo sử dụng GlobalAveragePooling2D để tính toán trung bình trên 2 chiều để đưa ra duy nhất 32 tính năng cuối cùng.

Cuối cùng là tạo ra mạng lưới thần kinh 2 lớp với 64 tế bào.

Đầu ra chỉ xuất ra 1 giá trị với activation= 'linear' (kích hoạt tuyến tính) vì đây là tác vụ hồi quy

Mô hình Model:

Layer (type)	Output Shape	Param #
input_2 (InputLayer)	[(None, 120, 120, 3)]	0
conv2d_2 (Conv2D)	(None, 118, 118, 16)	448
max_pooling2d_2 (MaxPooling 2D)	(None, 59, 59, 16)	0
conv2d_3 (Conv2D)	(None, 57, 57, 32)	4640
max_pooling2d_3 (MaxPooling 2D)	(None, 28, 28, 32)	0
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 32)	0
dense_3 (Dense)	(None, 64)	2112
dense_4 (Dense)	(None, 64)	4160
dense_5 (Dense)	(None, 1)	65

Cấu hình Model:

```
model.compile(  
    optimizer='adam',  
    loss='mse'  
)
```

3.1.5. Train Model

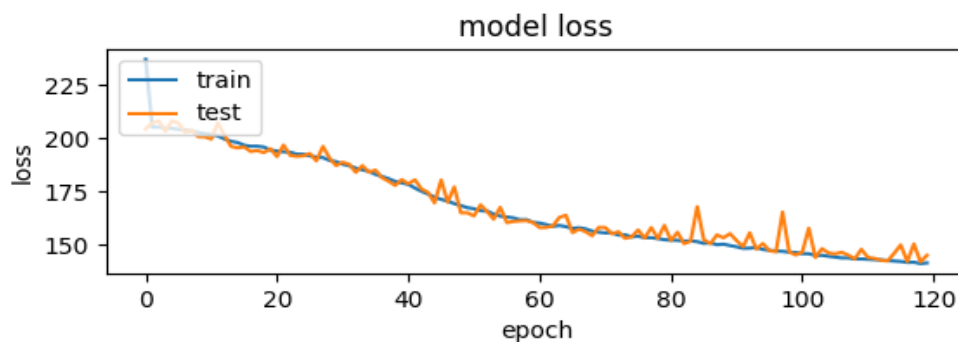
```
history = model.fit(  
    train_images,  
    validation_data=val_images,  
    epochs=100  
)
```

Đầu tiên ta chọn epochs = 100 để đánh giá kết quả train.

Sử dụng matplotlib để vẽ biểu đồ:

```
plt.plot(history.history['loss'])  
plt.plot(history.history['val_loss'])  
plt.title('model loss')  
plt.ylabel('loss')  
plt.xlabel('epoch')  
plt.legend(['train', 'test'], loc='upper left')  
plt.show()
```

Kết quả:



Dựa trên biểu đồ ta thấy train và test đã xấp xỉ nhau và có xu hướng lệch nhau tại epoch = 74. Do đó ta sẽ train lại với epoch = 74.

Lưu lại model sau khi train:

```
model.save('models')
```

3.2. Face_detection.py

Load model đã train và tại ImageDataGenerator.

```
model = keras.models.load_model('D:/chuyen_nganh/20221/machine_learning/project/Nhap_mom_ML-hao/Nhap_mom_ML-hao/models5')
test_generator = keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255
)
```

Với mỗi khung hình được haadcascade nhận diện là img_cat, ta sẽ resize theo shape (3, 120, 120, 3) rồi dùng trình tạo ở trên để đưa ảnh về khoảng [0, 1]. Sau đó dự đoán tuổi và lưu vào biến output_predict. Cuối cùng dùng cv2.putText để hiển thị dự đoán lên màn hình.

```
for (x, y, w, h) in faces: # For each detected face: (faces is the tuple of x,y--point of upper left corner,w-width,h-height)
    cv2.rectangle(frame, (x,y), (x+w,y+h), (255,0,0), 3) #frame on which rectangle will be there,top-left,bottom-right,color,thickness
    img_cat = frame[x:y, x+w:y+h]
    img_age = np.resize(img_cat, (3, 120, 120, 3))
    img_age = img_age.astype('float32')
    # model.predict(img_age)
    img_predict = test_generator.flow(img_age, batch_size=32, shuffle=True)
    output_predict = int(np.squeeze(model.predict(img_predict)).item(0))
    col = (0, 255, 0)
    cv2.putText(frame, str(output_predict), (x, y-10), cv2.FONT_HERSHEY_SIMPLEX, 1.2, col, 2)
```

3.3. Triển khai cài đặt

1. Clone repo link từ github: https://github.com/nghiaxir/Nhap_mom_ML.git
2. Download bộ dataset từ Kaggle và lưu ở project: [Age prediction | Kaggle](#)
3. Cài đặt môi trường Anaconda
 - Cài Anaconda tại link: <https://www.anaconda.com/products/distribution>
 - Mở Anaconda Prompt (Run as Administrator)
 - o Tạo môi trường mới: conda create --name env_machine
 - o Cấu hình môi trường: activate env_machine
 - o Cài đặt tensorflow: conda install -c conda-forge tensorflow
 - o Cài đặt pandas: conda install -c anaconda pandas
 - o Cài đặt matplotlib: conda install -c conda-forge matplotlib
 - o Cài đặt scikit-learn: conda install -c anaconda scikit-learn
 - o Cài đặt OpenCV: conda install -c conda-forge opencv
4. Cài đặt các thư viện còn thiếu
5. Mở project bằng pycharm
 - Chọn File -> Settings (Ctrl + Alt + S) -> Project -> Python Interpreter

Bài tập lớn: Nhập môn Học máy và Khai phá dữ liệu

- Đổi Python Interpreter sang đường dẫn file python.exe của anaconda

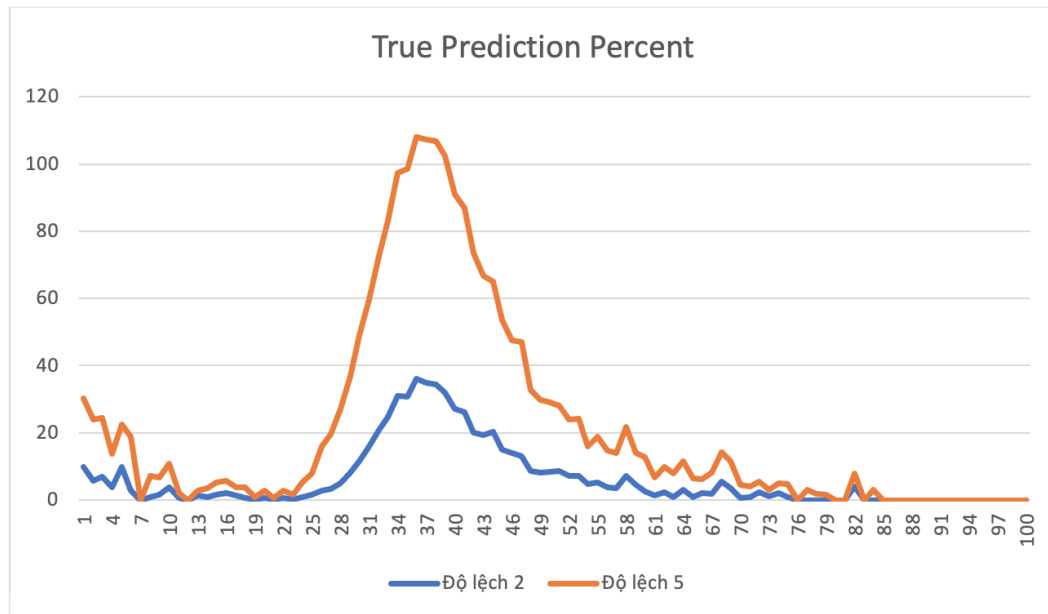
(Ví dụ: D:\Program Files\Anaconda\envs\env_machine\python.exe)

5. Sửa đường dẫn Model trong file face_detection.py
6. Chạy camera.py để mở app
7. Để test tập dữ liệu sử dụng file test.py (Sửa link Model và link dữ liệu image_dir)

CHƯƠNG 4. KẾT QUẢ TỔNG QUAN

4.1. Đánh giá mô hình

Sau khi đã huấn luyện và chạy thử với tập test thì kết quả thu được biểu đồ sau:



Biểu đồ gồm 2 đường “Độ lệch 2” và “Độ lệch 5” ứng với khoảng chấp nhận sai số do module dự đoán. Kết quả cho ta thấy module dự đoán tốt ở khoảng 23 – 50 tuổi.

Nguyên nhân do dữ liệu huấn luyện được tập trung chủ yếu ở khoảng tuổi này. Chấm điểm theo hệ số R^2 cũng cho kết quả tương tự.

```
1741/1741 [=====] - 316s 182ms/step
Test RMSE: 11.80705
Test R^2 Score: 0.32078
```

4.2. Sản phẩm demo

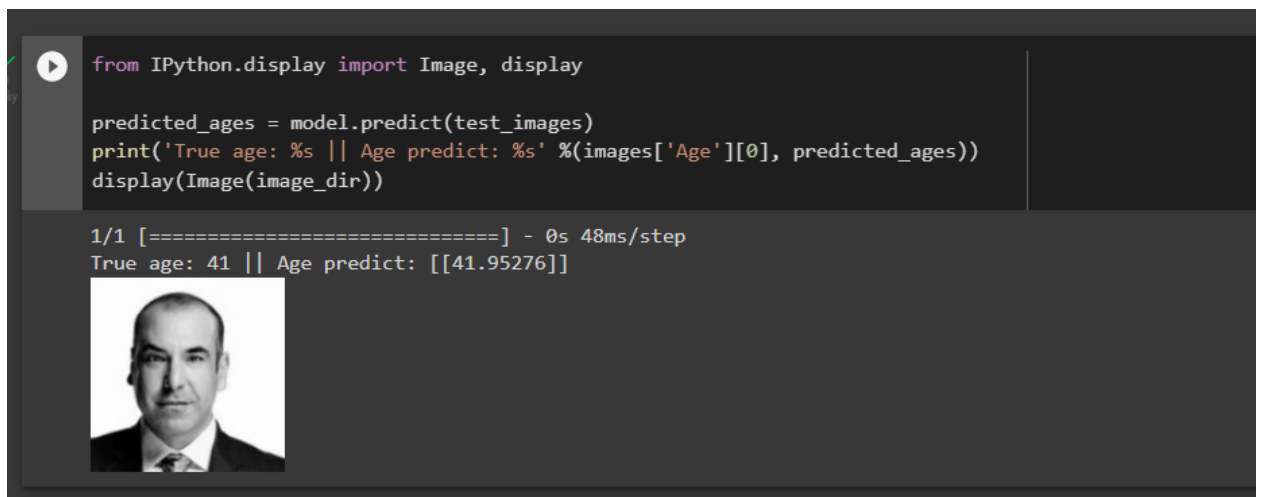
4.2.1. Kiểm thử tập dữ liệu test

Sử dụng file test.py để test các dữ liệu ảnh từ folder test.

Ví dụ, khi test file 035 (35 tuổi) ta được mảng đầu ra là các dự đoán của từng ảnh. Giá trị trung bình là 37,734962.

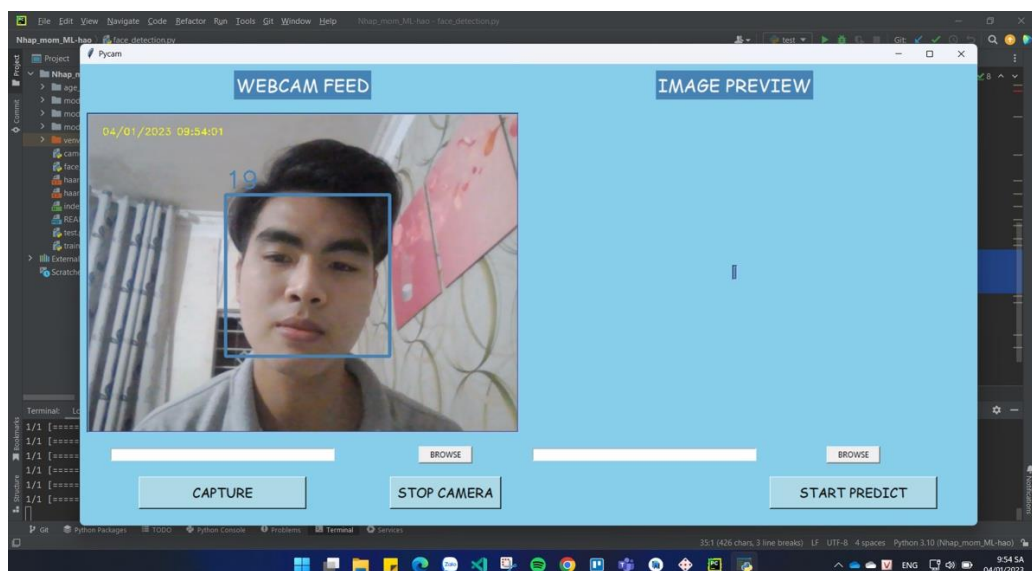

```
PS D:\chuyen_nganh\20221\machine_learning\project\Nhap_mom_ML-hao\Nhap_mom_ML-hao> py test.py
ok
D:\chuyen_nganh\20221\machine_learning\project\Nhap_mom_ML-hao\Nhap_mom_ML-hao\test.py:15: DeprecationWarning: use `int` by itself. Doing this will not modify any behavior and is safe. When replacing you wish to review your current use, check the release note link for additional information.
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/release/1.20
    ages = pd.Series(filepaths.apply(lambda x: os.path.split(os.path.split(x)[0])[1]), name='Age')
Found 1428 validated image filenames.
23/23 [=====] - 16s 710ms/step
[36.064457 41.11317 34.71931 ... 37.49979 39.381077 40.020935]
37.734062
```

Dự đoán tuổi của 1 ảnh:



4.2.2. Demo app Age Prediction real-time

Sau khi đã có mô hình hiệu quả, nhóm chúng em tiến hành kiểm nghiệm thực tế. Chúng em đã xây dựng một app để kiểm nghiệm hoạt động của mô hình. Chức năng cơ bản của app là cho người dùng có thể kiểm nghiệm chứng năng dự đoán độ tuổi khuôn mặt trên ảnh đầu vào hoặc trải nghiệm nhận diện real-time trên webcam.



Link video demo app Age Prediction real-time: https://drive.google.com/file/d/1-gh0hQR6WPkuy8olrRO2JxAmRFZod_0j/view?usp=sharing

4.3. Nhận xét, đánh giá

Trong khuôn khổ đồ án môn học, nhóm chúng em đã thiết kế được mô hình đạt được các yêu cầu đề ra:

- Phát hiện đúng khuôn mặt.
- Dự đoán được độ tuổi tương đối chính xác.
- Đạt được yêu cầu thời gian thực.
- Đạt được độ chính xác và ổn định cần thiết.

Các vấn đề còn thiếu sót:

- Bộ phân loại Haar Cascade chưa phát hiện khuôn mặt chính xác tuyệt đối. Nguyên nhân có thể do ảnh chưa được xử lý sâu như góc quay khuôn mặt, màu ảnh, kích thước khuôn mặt... Nhóm chúng em sẽ cố gắng khắc phục và thử nghiệm các phương pháp phát hiện khuôn mặt hiện đại hơn.
- Do sử dụng camera từ máy tính cá nhân nên bức ảnh chưa rõ nét, mô hình chưa thật sự phân biệt quá rõ đặc trưng sâu giữa các khuôn mặt, mặc dù các bài toán nhận diện ở thời điểm hiện tại yêu cầu độ chính xác cần $> 95-97\%$.
- Cải thiện kiến trúc mô hình để đạt được mô hình hiệu quả hơn.

4.4. Hướng phát triển

- Nâng cấp hệ thống camera để cải thiện chất lượng ảnh của người dùng.
- Thiết kế mô hình với các chỉ số như đầu vào bức ảnh, số bước lặp, kiến trúc mô hình sao cho phù hợp cùng hệ thống phần cứng xử lý như GPU để đáp ứng xử lý thời gian thực.

TÀI LIỆU THAM KHẢO

1. <https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf>
2. https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html
3. <https://viblo.asia/p/haar-cascade-la-gi-luan-ve-mot-ky-thuat-chuyen-dung-de-nhan-biet-cac-khuon-mat-trong-anh-E375zamdIGW>
4. <https://github.com/opencv/opencv/tree/master/data/haarcascades>
5. <https://topdev.vn/blog/thuat-toan-cnn-convolutional-neural-network/#cnn-convolutional-neural-network-la-gi>
6. <https://sefiks.com/2019/02/13/apparent-age-and-gender-prediction-in-keras/?fbclid=IwAR1yHp6J9Mtnf0Jky6lbPDg58-KNomtZFejoIMf-PhNsG3-aNWNG4Cigyq4>
7. <https://www.youtube.com/watch?v=rwiPcSrPPQk>
8. <https://www.youtube.com/watch?v=G0iYWNRuDcM&t=977s>
9. Bài giảng Nhập môn Học máy và Khai phá dữ liệu (IT3190) – TS. Nguyễn Nhật Quang
10. Giáo trình Deep Learning cơ bản, Tái bản lần thứ 2 – Nguyễn Thanh Tuấn