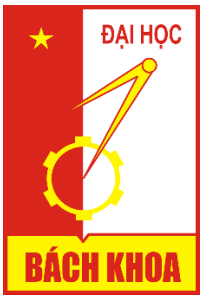
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG



ĐỒ ÁN MÔN HỌC

PROJECT I

Đề tài: Nhận diện khuôn mặt

GVHD: PGS.TS. Lê Thanh Hương

Sinh viên thực hiện : Ngô Thành Nam_20194127

Hà Nội – Tháng 1 năm 2022

MỤC LỤC

A. LỜI NÓI ĐẦU

Trong một thế giới công nghệ đang hằng ngày thay đổi, việc học tập kiến thức nền tảng đối với sinh viên kĩ thuật, đặc biệt là sinh viên công nghệ thông tin (CNTT) là vô cùng quan trọng. Bởi lẽ bất cứ công nghệ hiện đại nào cũng được xây dựng nên từ những kiến thức nền tảng cốt lõi, việc học tập những kiến thức này chính là chìa khoá giúp sinh viên CNTT nắm bắt bất cứ công nghệ mới nào một cách nhanh chóng, đầy đủ, rõ ràng về bản chất nhất.

Hiện nay trong thời kỳ đổi mới và phát triển. Xu thế 4.0 đã mang đến cho con người nhiều sự đột phá trong công nghệ đặc biệt phải nhắc tới sự hiện diện của "AI". Trí tuệ nhân tạo là một ngành thuộc lĩnh vực khoa học máy tính dần được mọi người quan tâm và chú ý hơn.

Ngày nay các hệ thống nhận diện khuôn mặt là một trong những hệ thống không thể thiếu ở các công ty, doanh nghiệp dù lớn hay nhỏ. Và nó đang phát triển rộng rãi kể cả ở các trường học. Em quyết định thực hiện một project nhỏ về đề tài nhận diện khuôn mặt sử dụng các thuật toán học máy đơn giản.

Vì em chỉ mới tiếp cận với các công nghệ trong học máy và học sâu nói riêng cũng như trí tuệ nhân tạo nói chung nên project này khó tránh khỏi những sai sót. Hy vọng nhận được những lời góp ý và nhận xét của cô để những dự án thực tế về sau của em đạt được kết quả tốt nhất.

B. NỘI DUNG

Chương I. TỔNG QUAN VỀ DỰ ÁN

1. Một số khái niệm

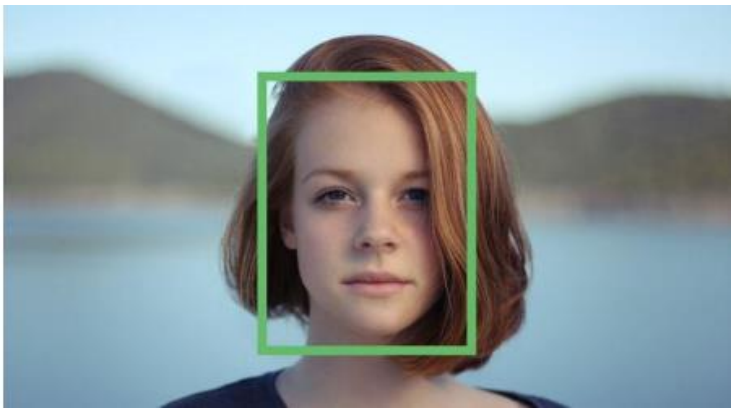
1.1. Thị giác máy tính

Thị giác máy tính (computer vision) là một trong những lĩnh vực rất quan trọng trong AI giúp giải quyết được nhiều bài toán liên quan tới phân loại ảnh, nhận diện vật thể, phân khúc hình ảnh, xác thực khuôn mặt, ... Thị giác máy tính có nhiều tiềm năng ứng dụng trong các ngành công nghiệp như sản xuất ô tô, thiết bị thông minh, robotics, ... Không những thế trong y sinh nó còn giúp bác sĩ chuẩn đoán vị trí khối u dựa trên mô hình nhận diện khối u. Trong quản lý giao thông chúng ta cũng có thể ứng dụng thị giác máy tính để đếm lưu lượng xe cộ. Thuật toán OCR tự động trích xuất nội dung chữ từ ảnh chụp văn bản, ...

1.2. Face detection (Phát hiện khuôn mặt)

Face detection là một tính năng đơn giản dùng để phát hiện có sự hiện diện trong khuôn mặt người trong một bức ảnh, một video, …

Hệ thống chỉ đánh giá xem trong khung hình đó liệu có khuôn mặt người không chứ không xác định rõ khuôn mặt đó là của ai.



Ảnh minh họa

1.3. Face Recognition (Nhận diện khuôn mặt)

Trong khi face detection chỉ phát hiện xem có khuôn mặt trong một khung hình hay không thì face recognition sẽ đánh giá xem khuôn mặt đó thuộc về người nào dựa vào một database mà người lập trình dựng sẵn.

Ảnh minh họa

2. Các phương pháp xác thực khuôn mặt

2.1. Phương pháp truyền thống

Các phương pháp truyền thống: Xác thực khuôn mặt bằng cách trích xuất ra một land mark cho face. Land mark như là một bản đồ xác định các vị trí cố định trên khuôn mặt của một người như mắt, mũi, miệng, lông mày,….

Như vậy thay land mark face đã loại bỏ những phần thông tin không cần thiết và giữ lại những thông tin chính. Khi đó mỗi khuôn mặt sẽ được nén thành một véc tơ n chiều. Thông thường là 68 chiều.

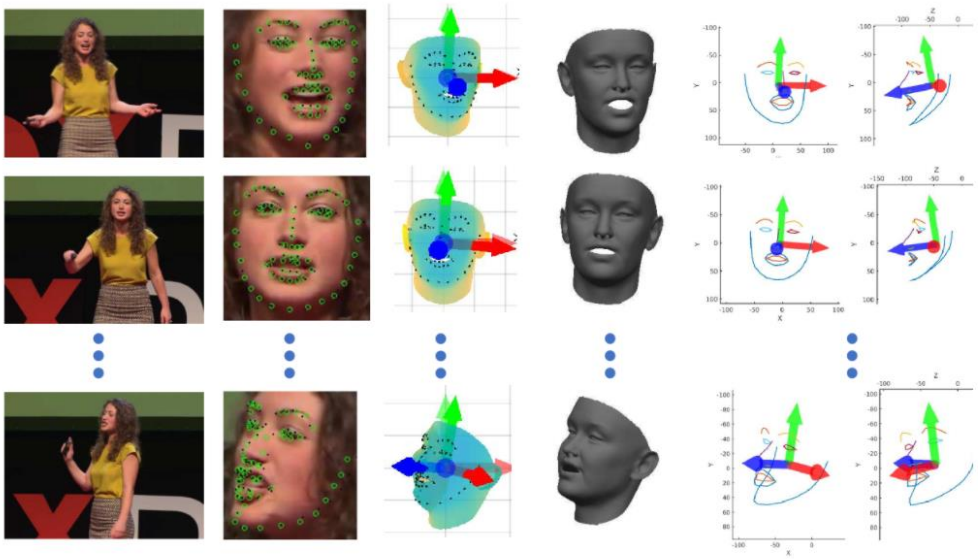
Sử dụng đầu vào là land mark face, áp dụng các thuật toán cổ điển như SVM, k-NN, Naive Bayes, Random Forest,, … để phân loại khuôn mặt cho một người.

2.2. Nhận diện 3D

Kĩ thuật nhận diện 3D sẽ sử dụng không gian 3 chiều để biểu diễn khuôn mặt. Từ thông tin này có thể xác định các đặc trưng khác nhau trên bề mặt khuôn mặt như các đường countour của mắt, mũi, cằm.

Một lợi thế của nhận diện khuôn mặt 3D là không bị ảnh hưởng bởi những thay đổi về ánh sáng như các phương pháp 2D. Dữ liệu 3D đã cải thiện đáng kể độ chính xác của nhận dạng khuôn mặt.

Để tạo ra một ảnh 3D, một cụm ba camera được áp dụng. Mỗi camera sẽ hướng vào một góc khác nhau. Tất cả các camera này phối hợp cùng nhau trong việc theo dõi khuôn mặt của một người trong thời gian thực và có thể nhận diện chúng.



3. Các thuật toán nhận diện khuôn mặt

3.1. One-shot learning

One-shot learning là thuật toán học có giám sát mà mỗi một người chỉ cần 1 vài, rất ít hoặc thậm chí chỉ 1 bức ảnh duy nhất (để khởi tạo ra nhiều biến thể).

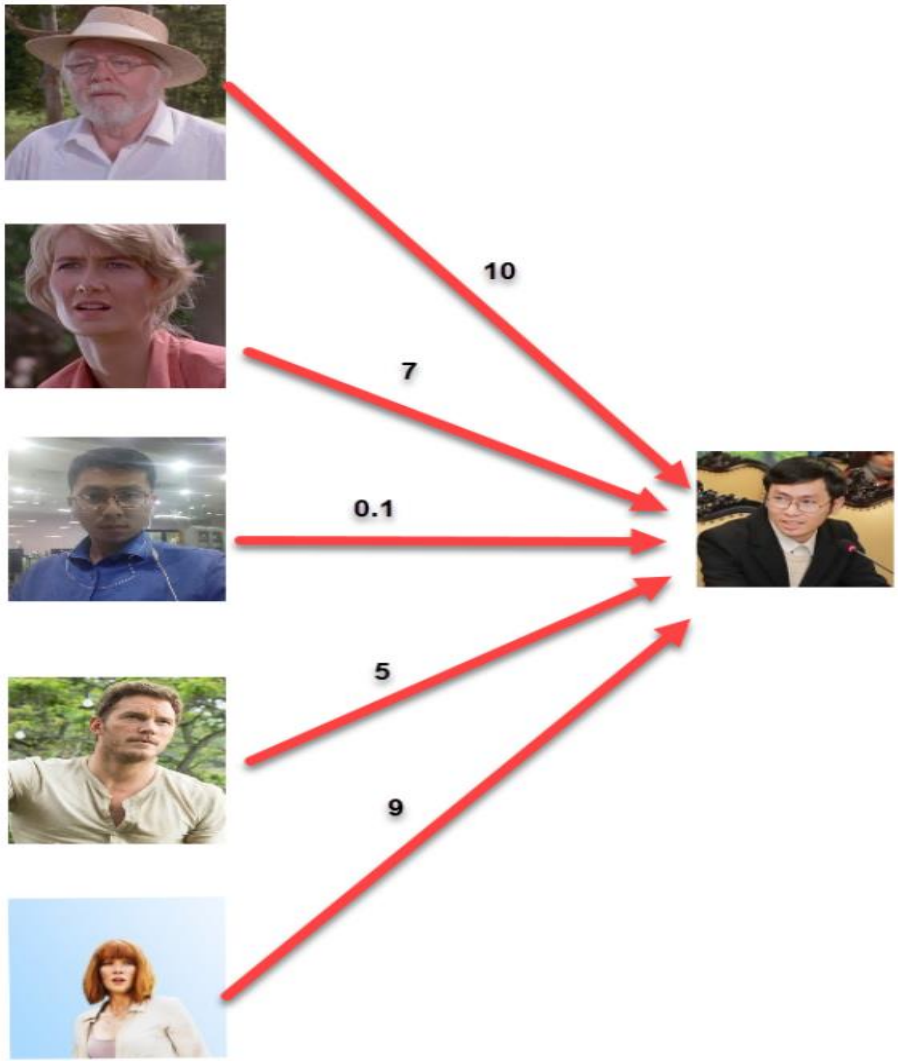
Từ đầu vào là bức ảnh của một người, chúng ta sử dụng một kiến trúc thuật toán CNN đơn giản để dự báo người đó là ai.

Tuy nhiên nhược điểm của phương pháp này là chúng ta phải huấn luyện lại thuật toán thường xuyên khi xuất hiện thêm một người mới vì shape của output thay đổi tăng lên 1. Rõ ràng là không tốt đối với các hệ thống nhận diện khuôn mặt của một công ty vì số lượng người luôn biến động theo thời gian.

3.2. Learning similarity

Phương pháp này dựa trên một phép đo khoảng cách giữa 2 bức ảnh, thông thường là các norm chuẩn l1 hoặc l2 sao cho nếu 2 bức ảnh thuộc cùng một người thì khoảng cách là nhỏ nhất và nếu không thuộc thì khoảng cách sẽ lớn hơn.

$$\begin{cases} d(\text{img1}, \text{img2}) \leq \tau & \rightarrow \quad \text{same} \\ d(\text{img1}, \text{img2}) > \tau & \rightarrow \quad \text{different} \end{cases}$$



Hình 1

Hình 1: Phương pháp learning similarity.

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 (bên phải) với toàn bộ các ảnh còn lại (bên trái). Ta cần chọn một ngưỡng threshold để quyết định ảnh là giống hoặc khác. Giả sử ngưỡng threshold là 0.5. Trong các bức ảnh bên trái thì bức ảnh ở giữa có khoảng cách với ảnh bên phải nhỏ hơn 0.5. Do đó nó được dự báo cùng một người với ảnh bên phải.

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.

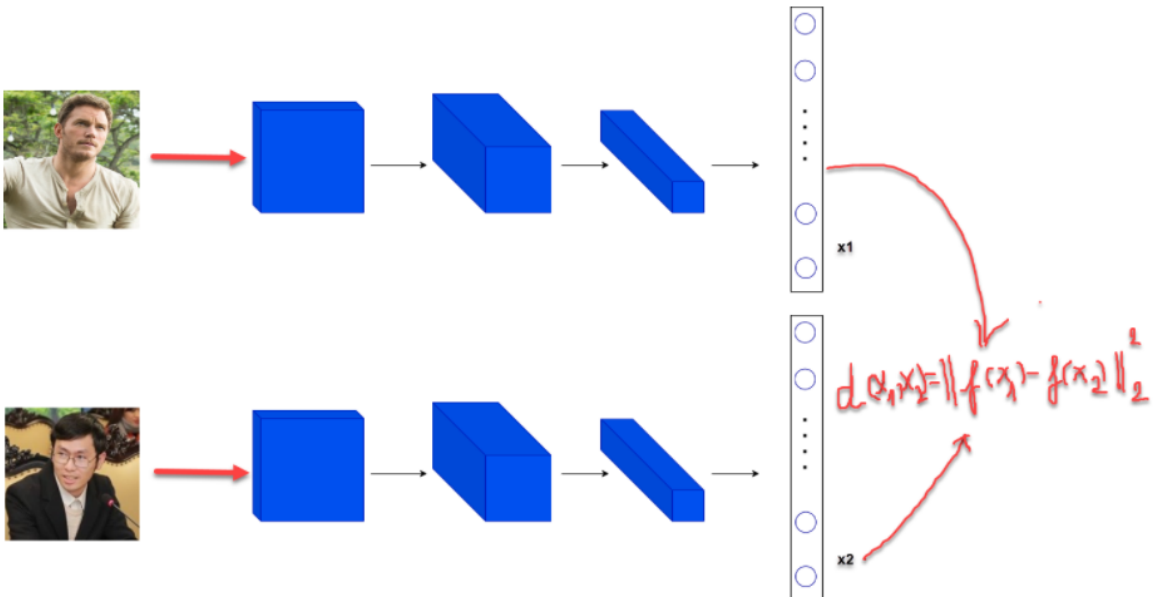
Điểm mấu chốt là cần xây dựng được một model encoding đủ tốt để chiếu các bức ảnh lên một không gian eucledean n chiều. Sau đó sử dụng khoảng cách để quyết định nhãn của chúng.

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.

3.3. Siam network

Những kiến trúc mạng mà khi ta đưa vào 2 bức ảnh và mô hình sẽ trả lời chúng thuộc về cùng 1 người hay không được gọi chung là Siam network

Kiến trúc của Siam network dựa trên base network là một Convolutional neural network **đã được loại bỏ output layer** có tác dụng encoding ảnh thành véc tơ embedding. Đầu vào của mạng siam network là 2 bức ảnh bất kì được lựa chọn ngẫu nhiên từ dữ liệu ảnh. Output của Siam network là 2 véc tơ tương ứng với biểu diễn của 2 ảnh input. Sau đó chúng ta đưa 2 véc tơ vào hàm loss function để đo lường sự khác biệt giữa chúng.



Hình 2: Từ mô hình Convolutional neural network, mô hình trả ra 2 véctơ encoding là x1 và x2 biểu diễn cho lần lượt ảnh 1 và 2. x1 và x2 có cùng số chiều. Hàm f(x) có tác dụng tương tự như một phép biến đổi qua layer fully connected trong mạng neural network để tạo tính phi tuyến và giảm chiều dữ liệu về các kích thước nhỏ. Thông thường là 128 đối đối với các mô hình pretrain.

+) Khi x1, x2 là cùng 1 người thì

$$||f(\mathbf{x_1}) - f(\mathbf{x_2})||_2^2$$

Phải là 1 giá trị nhỏ

+) Khi x1, x2 không phải là cùng 1 người thì

$$||f(\mathbf{x_1}) - f(\mathbf{x_2})||_2^2$$

Phải là 1 giá trị lớn

4. Thuật toán facenet

4.1. Khái quát thuật toán

Facenet đã giải quyết được vấn đề của các thuật toán trước đó là:

- Base network áp dụng một mạng convolutional neural network và giảm chiều dữ liệu xuống chỉ còn 128 chiều. Do đó quá trình suy diễn và dự báo nhanh hơn và đồng thời độ chính xác vẫn được đảm bảo.
- Sử dụng loss function là hàm triplot loss có khả năng học được **đồng thời** sự giống nhau giữa 2 bức ảnh cùng nhóm và phân biệt các bức ảnh không cùng nhóm. Do đó hiệu quả hơn rất nhiều so với các phương pháp trước đây

4.2. Triple loss

Trong facenet, quá trình encoding của mạng convolutional neural network đã giúp ta mã hóa bức ảnh về 128 chiều. Sau đó những véc tơ này sẽ làm đầu vào cho hàm loss function đánh giá khoảng cách giữa các véc tơ.

Để áp dụng triple 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. 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).

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 function luôn lấy 3 bức ảnh làm input và trong mọi trường hợp ta kì vọng:

$$d(\mathbf{A}, \mathbf{P}) < d(\mathbf{A}, \mathbf{N})$$

(bất đẳng thức (1))

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 biểu thức trở thành:

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

Như vậy hàm loss function sẽ là:

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

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

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 hợp mô hình nhận diện sai ảnh Negative thành Postive 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.

Tức là nếu:

$$||f(\mathbf{A}) - f(\mathbf{P})||_2^2 - ||f(\mathbf{A}) - f(\mathbf{N})||_2^2 + \alpha \leq 0$$

sẽ được điều chỉnh về 0. Khi đó hàm loss function trở thành:

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

Như vậy khi áp dụng Triple loss vào các mô hình convolutional neural network chúng ta có thể tạo ra các biểu diễn véc tơ tốt nhất cho mỗi một bức ảnh. Những biểu diễn véc tơ 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 chiếu euclidean.

4.3. Lựa chọn triple images input

Nếu lựa chọn triple input một cách ngẫu nhiên có thể ảnh khiến cho bất đẳng thức (1) dễ dàng xảy ra vì trong các ảnh ngẫu nhiên, khả năng giống nhau giữa 2 ảnh là rất khó. Hầu hết các trường hợp đều thỏa mãn bất đẳng thức (1) và không gây ảnh hưởng đến giá trị của loss function do giá

trị của chúng được set về 0. Như vậy việc học những bức ảnh Negative quá khác biệt với Anchor sẽ không có nhiều ý nghĩa.

Để mô hình khó học hơn và đồng thời cũng giúp mô hình phân biệt chuẩn xác hơn mức độ giống và khác nhau giữa các khuôn mặt, chúng ta cần lựa chọn các input theo bộ 3 khó học (hard triplets).

Ý tưởng là chúng ta cần tìm ra bộ ba (A,N,P) sao cho (1) là gần đạt được đẳng thức (xảy ra dấu =) nhất. Tức là d(A,P) lớn nhất và d(A,N) nhỏ nhất. Hay nói cách khác với mỗi Anchor A cần xác định:

- **Hard Positive:** Bức ảnh Positive có khoảng cách xa nhất với Anchor tương ứng với nghiệm:

$$\text{argmax}_{\mathbf{P}_i}(d(\mathbf{A}_i, \mathbf{P}_i))$$

- **Hard Negative:** Bức ảnh Negative có khoảng cách gần nhất với Anchor tương ứng với nghiệm:

$$\text{argmin}_{\mathbf{N}_j}(d(\mathbf{A}_i, \mathbf{N}_j))$$

Chương II. ÁP DỤNG VÀO BÀI TOÁN

1. Dataset

Dữ liệu mà em sử dụng là ảnh của 5 người, mỗi người có khoảng 70 tấm ảnh để đưa vào huấn luyện.

2. Sử dụng pretrain model

Đầu tiên em sử dụng 1 mô hình pretrain có sẵn . Model này có tác dụng embedding các khuôn mặt có trong bức ảnh thành những véc tơ embedding 128 chiều.

2.1. Tiền xử lý dữ liệu

Em xây dựng hàm blobImage để giảm nhiễu cho ảnh do chiếu sáng (illumination)

(Hình 1)

Hình 1: Ảnh gốc và ảnh đã được blob. Ta có thể nhận thấy ảnh đã được segment thành các vùng ảnh có chung cường độ màu sắc. Do đó ảnh hưởng của thay đổi màu sắc do ánh sáng đã được giảm thiểu.

2.2. Trích xuất các khuôn mặt

2.3. Embedding từ pretrain model

Sử dụng các khuôn mặt đã được trích xuất để tạo embedding vector. Đầu vào của model là các ảnh blob kích thước 96x96.

2.4. Most similarity

Sau khi lấy được dữ liệu các khuôn mặt, em sử dụng phương pháp learning similarity ở trên để tìm kiếm các ảnh tương đồng nhất làm nhãn cho ảnh dự báo.

Để tính toán similarity thì em dùng hàm cosine_similarity của sklearn

3. Training triplot loss

Để so sánh với kết quả của pretrain mode ở trên thì em huấn luyện lại model mới cho bộ dữ liệu của mình.

3.1. Base network model

Base_network model của em là VGG16, và có thể load kiến trúc mạng này từ keras.

3.2. Preprocessing data

Dữ liệu ảnh các khuôn mặt hiện tại đang không cùng shape, do đó cần phải resize lại các ảnh thông qua hàm resize của opencv.

3.3. Triplet-semi-hard loss

Em sử dụng hàm triplet-semi-hard loss của tensoflow.

3.4. Huấn luyện model

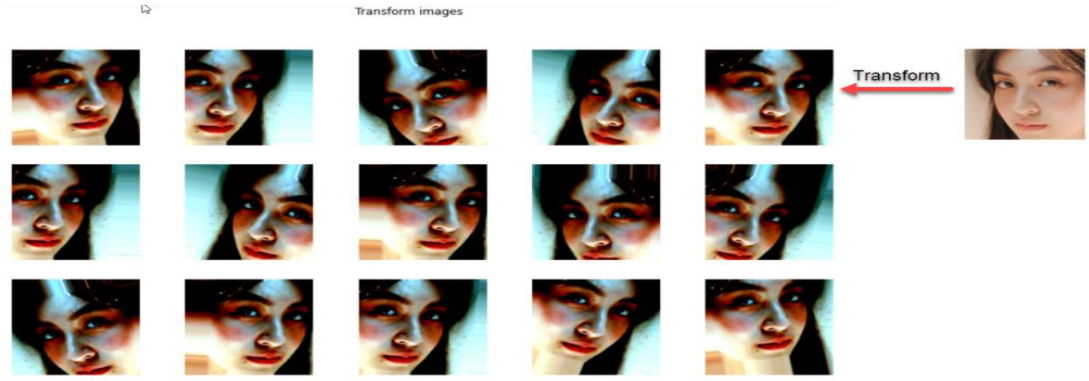
Để huấn luyện model, em khởi tạo một tensorflow dataset với batch_size = 32 và shuffle sau mỗi 1024 steps.

4. Data Augumentation

Tiếp theo em sẽ khởi tạo một ImageDataGenerator để thực hiện một loạt các bến đổi cho hình ảnh. Trong đó bao gồm:

- Chuẩn hóa theo phân phối chuẩn các pixels của ảnh: Trung bình các pixels bằng 0, phương sai bằng 1.

- Tạo các ảnh với các góc nghiêng là 20 độ.
- Dịch chuyển ảnh theo width, height.
- Lật ảnh theo chiều ngang.



4.1. Huấn luyện model

Em điều chỉnh tăng batch_size = 64 và shuffle sau mỗi 1024 steps.

Chương III. ĐÁNH GIÁ KẾT QUẢ

1. Pretrain model

Với mô hình pretrain model có sẵn thì độ chính xác đạt được của mô hình với bộ dữ liệu là :

```
0.7213114754098361
```

2. Model faceNet được huấn luyện lại

Độ chính xác đạt được với bộ dữ liệu là:

```
0.7540983606557377
```

3. Model faceNet sau khi sử dụng Augumentation

```
0.7704918032786885
```

Kết luận: Em nhận thấy rằng trong điều kiện bộ dữ liệu là không quá tốt thì đây là một kết quả chấp nhận được.

Chương IV.HƯỚNG PHÁT TRIỂN

• Tối ưu các tham số của các thuật toán kể trên .

• Tìm hiểu thêm các thuật toán thiên về hướng học sâu vừa có độ chính xác cao vừa có tốc độ real time.

• Tìm hiểu thêm các cách xử lý với dữ liệu lớn vì trong thực tế hệ thống có thể nhận diện lên tới hàng nghìn người.

C. TÀI LIỆU THAM KHẢO

https://arxiv.org/pdf/1503.03832.pdf

http://llcao.net/cu-deeplearning17/pp/class10_FaceNet.pdf

https://arxiv.org/abs/1406.4773.pdf

https://www.researchgate.net/publication/329893282_Face_Recognition_Based_on_Improved_FaceNet_Model