

FACE IMAGE RETRIEVAL USING DEEP LEARNING MODELS

Lê Phú Trường Đào Xuân Tân Trần Lê Hữu Vinh

Khoa Toán - Tin học
Trường Đại học Khoa học Tự nhiên - ĐHQG TP.HCM

Ngày 23 tháng 1 năm 2025

Nội dung đề tài

- ① Giới thiệu đề tài
- ② Xử lý dữ liệu đầu vào
- ③ Mô hình và các kỹ thuật liên quan
- ④ Đánh giá mô hình
- ⑤ DEMO - Query Interface Design
- ⑥ Lời kết

Giới thiệu đề tài

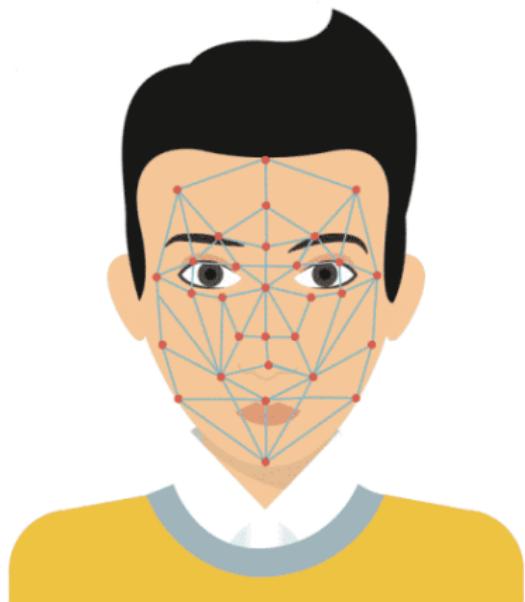
I. Giới thiệu đề tài

1. Mục tiêu của đề tài

1. **Xây dựng hệ thống CBIR (Content-Based Image Retrieval)**, cho phép người dùng tìm kiếm hình ảnh tương tự dựa trên ảnh đầu vào.
2. **Tối ưu hóa độ hiệu quả và chính xác** ứng dụng Deep Learning và các đặc trưng khuôn mặt để nâng cao khả năng nhận diện.
3. **Cải thiện trải nghiệm người dùng**, tạo ra một giải pháp thực tiễn cho các lĩnh vực **nhận diện khuôn mặt, quản lý truyền thông, và bảo mật**.

I. Giới thiệu đề tài

1. Mục tiêu của đề tài



CAPTURING

The foremost requirement is to capture the image and that can be done by scanning existing images or using cameras.



EXTRACTING

Unique facial data is then extracted from the sample.



COMPARING

The data is then Compared with the database.



MATCHING

The software then decides whether the sample matches any picture in the database or not.

Hình 1: CBIR System.

I. Giới thiệu đề tài

2. Dataset

CelebFaces Attributes Dataset (CelebA) là một bộ dữ liệu quy mô lớn với các thuộc tính khuôn mặt, bao gồm hơn 200.000 hình ảnh các nhân vật nổi tiếng.



Hình 2: *CelebA Dataset.*

I. Giới thiệu đề tài

2. Dataset

Các hình ảnh trong bộ dữ liệu này bao quát nhiều biến thể về tư thế và nền ảnh phức tạp. CelebA sở hữu sự đa dạng lớn, số lượng phong phú và các chú thích chi tiết, bao gồm:

- **202.599** hình ảnh khuôn mặt của các nhân vật nổi tiếng, nền ảnh dạng. Kích thước **128x128**, định dạng **JPEG**.
- **10.177** định danh (identities) (tên không được công bố).
- **5** landmark locations, **40** binary attributes trên mỗi khuôn mặt (màu tóc, có râu/không có râu...).

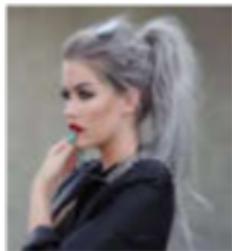
Trong project này nhóm chỉ sử dụng dữ liệu liên quan đến **identities**, **face images** và **attributes**.

I. Giới thiệu đề tài

2. Dataset



Smile



Gray Hair



Mustache



Eyeglasses



Attractive

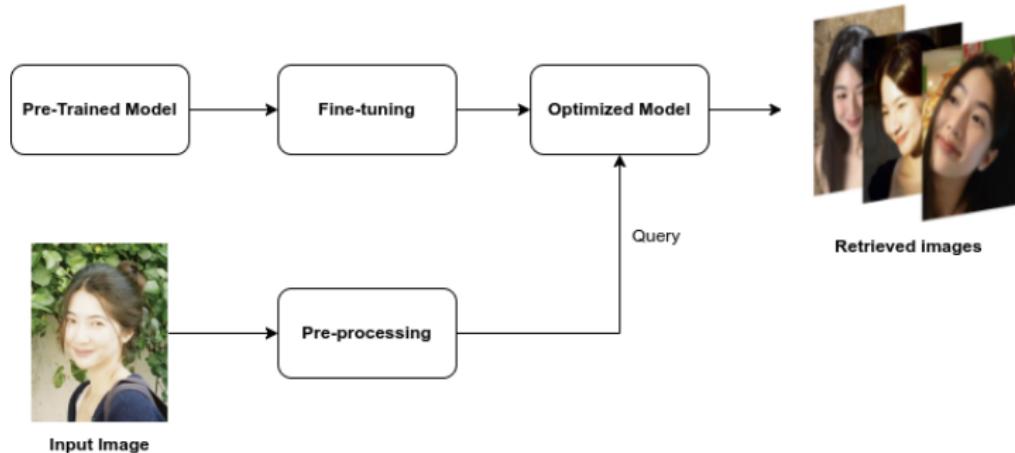


Wearing Hat

Hình 3: *Attributes in CelebA.*

I. Giới thiệu đề tài

3. Kiến trúc dự án



Hình 4: Project architecture.

Xử lý dữ liệu đầu vào

II. Xử lý dữ liệu đầu vào

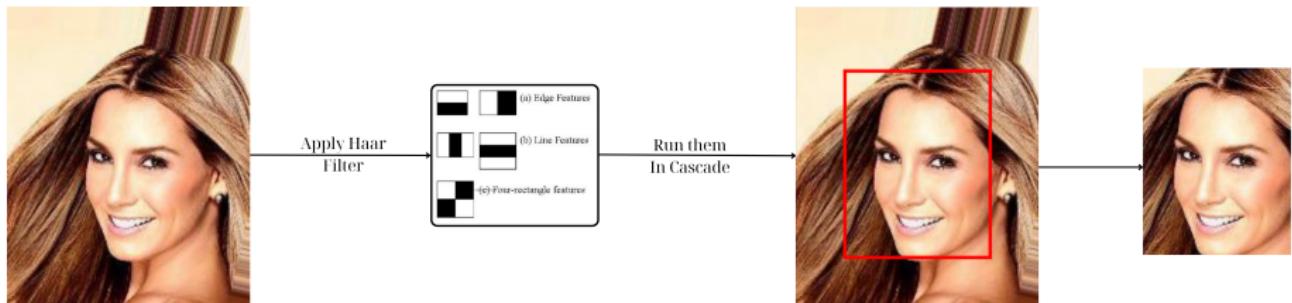
1. Tiền xử lý

Quy trình tiền xử lý:

- **Quét:** Ta quét từng phần của ảnh với kích thước vùng quét giảm dần, và xem vùng nào chứa mặt.
- **Bộ lọc Haar:** Việc quét dùng bộ lọc Haar, gồm nhiều bước lọc, mỗi bước gồm các bộ lọc phân biệt 1 vài đặc trưng của mặt, như mũi, miệng.
- **Cascade:** Nếu một bước thất bại giữa chừng, ta không xét nó tiếp.
- **Lấy trung bình:** Đến cuối, ta lấy trung bình của các vùng quét được nhận diện là có chứa mặt

II. Xử lý dữ liệu đầu vào

1. Tiền xử lý



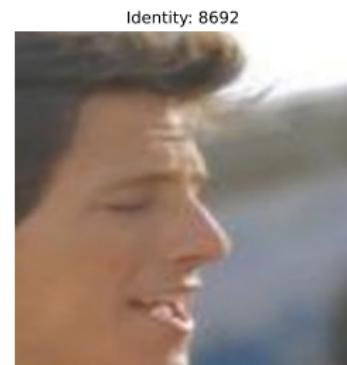
Hình 5: Bộ lọc Haar.

Sau bước tiền xử lý, chúng ta còn lại **196.791** ảnh dữ liệu.

II. Xử lý dữ liệu đầu vào

2. Xử lý dữ liệu cho mô hình

Đây là bước xử lý quan trọng vì đây sẽ là tập dữ liệu huấn luyện trong các mô hình, vì vậy việc xử lý phải được lên ý tưởng và thực hiện một cách cẩn thận. Mục tiêu chính của phần này là chuẩn bị dữ liệu để mô hình học cách phân biệt ảnh *cùng ID* (Anchor, Positive) và *khác ID* (Negative).



Hình 6: Các identity của ảnh.

II. Xử lý dữ liệu đầu vào

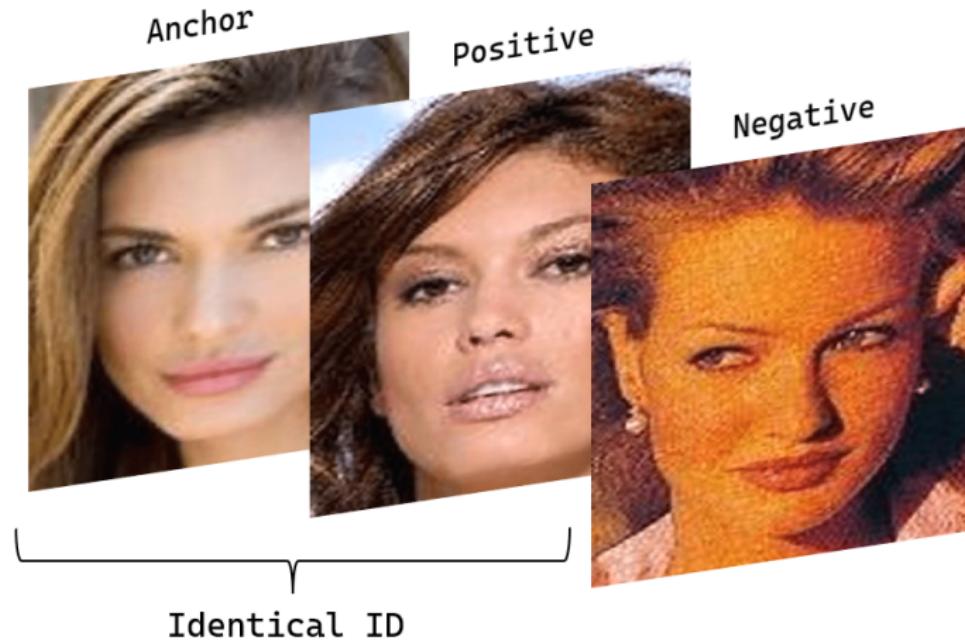
2. Xử lý dữ liệu cho mô hình

Quy trình xử lý:

1. **Lọc dữ liệu:** Loại bỏ ảnh hỏng và các ID có ít hơn 2 ảnh.
2. **Tổ chức theo ID:** Nhóm ảnh cùng ID, đảm bảo ảnh đủ cho cặp *Anchor - Positive*.
3. **Chia tập Train/Test:** Chia 80% ID dùng để huấn luyện, 20% để kiểm thử. Việc làm này đảm bảo không rò rỉ dữ liệu giữa hai tập.
4. **Sinh bộ ba (Triplet):**
 - Chọn ngẫu nhiên *Anchor, Positive* từ cùng ID.
 - Chọn *Negative* từ ID khác, đảm bảo mẫu *Negative* hợp lệ.
5. **Tạo Query/Gallery:**
 - **Query:** 1 ảnh từ mỗi ID trong tập Test.
 - **Gallery:** Các ảnh còn lại từ cả ID giống và khác Query.

II. Xử lý dữ liệu đầu vào

2. Xử lý dữ liệu cho mô hình



Hình 7: Tạo các bộ triplet

II. Xử lý dữ liệu đầu vào

3. Chuẩn bị dữ liệu và các phép biến đổi

Đây là bước xử lý tiếp theo nhằm Tiền tăng cường dữ liệu để tối ưu hiệu quả học với *Triplet Loss*. Quy trình xử lý:

1. **Resize:** Dưa toàn bộ ảnh về kích thước 218×218 , đảm bảo tính nhất quán.
2. **Normalize:** Chuẩn hóa RGB với các giá trị từ bộ ImageNet:

$$\mu = [0.485, 0.456, 0.406], \quad \sigma = [0.229, 0.224, 0.225].$$

3. Tăng cường dữ liệu (Augmentation):

- *Random Erasing*: Xóa ngẫu nhiên một vùng ảnh ($p = 0.5$) để mô phỏng che khuất.
- *Dropout*: Loại bỏ vùng ảnh nhỏ ($p = 0.3$) để giảm sự phụ thuộc vào một khu vực cụ thể.

4. Dữ liệu kiểm thử: Chỉ áp dụng *Resize* và *Normalize*, không dùng các phép biến đổi ngẫu nhiên.

II. Xử lý dữ liệu đầu vào

3. Chuẩn bị dữ liệu và các phép biến đổi

5. Batch size:

- Tập huấn luyện: $N = 64$ hoặc 128 .
- Tập kiểm thử: $N = 16$ hoặc 32 .

6. Dataset tổ chức triplet: Tạo các (a, p, n) trong mỗi batch, đảm bảo kích thước và định dạng thống nhất.



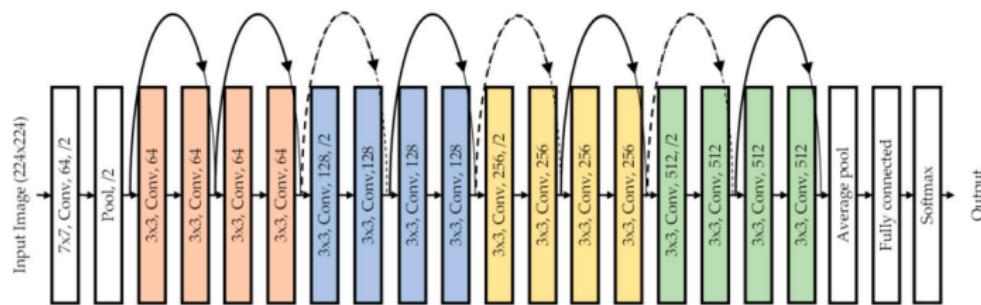
Hình 8: Thực hiện các phép biến đổi.

Mô hình và các kỹ thuật liên quan

III. Mô hình và các kỹ thuật liên quan

1. Giới thiệu mô hình

1. **ResNet:** Mô hình giúp giải quyết vấn đề Vanishing Gradient, Exploding Gradient và suy giảm hiệu suất khi mạng neural sâu nhờ cơ chế sử dụng hàm dư $F(x)$ với $H(x) = F(x) + x$ (skip connection) giúp gradient lan truyền hiệu quả và học ánh xạ dễ dàng hơn.



Hình 9: Cấu trúc ResNet-18.

III. Mô hình và các kỹ thuật liên quan

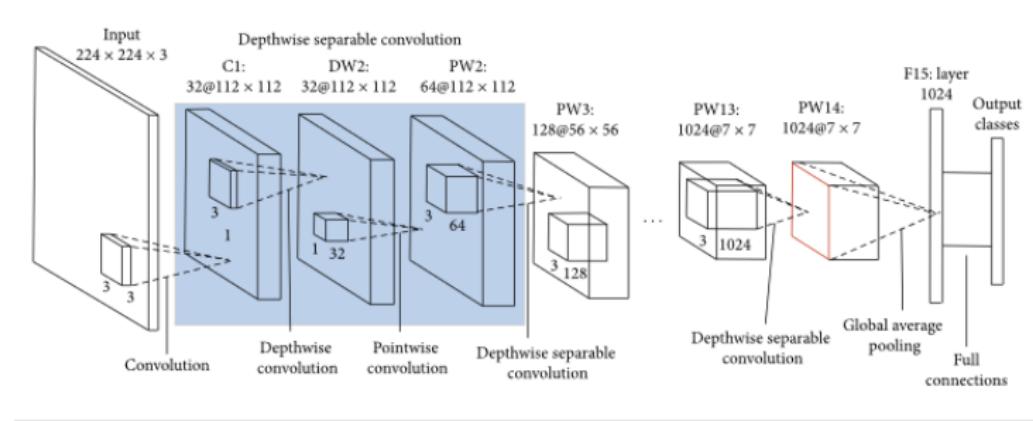
1. Giới thiệu mô hình

Trong bài toán này, chúng tôi sẽ sử dụng mô hình **ResNet-50** với khoảng **25.6 triệu** bộ tham số và số lượng đặc trưng (features) được trích xuất là **2048**.

2. **MobileNet:** Mô hình giúp tối ưu hiệu năng trên thiết bị di động với chi phí tính toán thấp hoặc tối ưu việc tính toán trên những thiết bị không mạnh mẽ về phần cứng nhờ vào cơ chế **Depthwise Convolution** và **Pointwise Convolution**.

III. Mô hình và các kỹ thuật liên quan

1. Giới thiệu mô hình



Hình 10: Cấu trúc MobileNet.

Trong bài toán này, chúng tôi sẽ sử dụng mô hình **MobileNet-V2** với khoảng **3.5 triệu** bộ tham số và số lượng đặc trưng (features) được trích xuất là **1280**.

III. Mô hình và các kỹ thuật liên quan

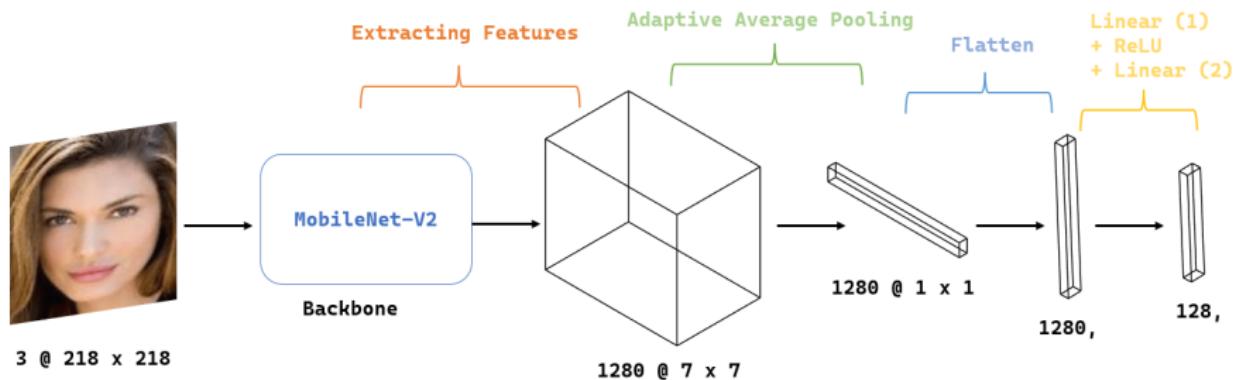
2. Thiết kế Embedding Layer

Embedding Layer đóng vai trò quan trọng trong việc tối ưu hóa biểu diễn đặc trưng của hình ảnh và đảm bảo rằng các đầu ra của mô hình phù hợp với không gian tìm kiếm ảnh với thiết kế cụ thể như sau:

1. **Adaptive Average Pooling**: Tổng hợp thông tin toàn cục, giảm kích thước đầu ra backbone xuống (1×1), giữ nguyên số kênh. Đầu ra: $(N, C, 1, 1)$.
2. **Flatten**: Chuyển đầu ra từ $(N, C, 1, 1)$ sang (N, C) để chuẩn bị cho các lớp Fully Connected.
3. **Linear Layer (1)**: Giảm chiều vector đặc trưng xuống kích thước embedding.
4. **ReLU**: Hàm kích hoạt phi tuyến loại bỏ giá trị âm, tăng khả năng học và biểu diễn của embedding.
5. **Linear Layer (2)**: Tinh chỉnh vector embedding, giữ nguyên kích thước không gian embedding.

III. Mô hình và các kỹ thuật liên quan

2. Thiết kế Embedding Layer



Hình 11: Minh họa Embedding Block cho mô hình MobileNet-V2.

III. Mô hình và các kỹ thuật liên quan

3. Phương pháp truy vấn: KD-Tree

KD-Tree (K-dimensional Tree) là một cấu trúc dữ liệu phân hoạch nhị phân trong không gian nhiều chiều, được thiết kế để tổ chức và truy vấn hiệu quả các điểm trong tập dữ liệu lớn.

III. Mô hình và các kỹ thuật liên quan

3.1. Xây dựng KD-Tree

1. Tập dữ liệu $D = \{f_1, f_2, \dots, f_n\}, f_i \in \mathbb{R}^k$.
2. Mỗi nút (*Node*) phân hoạch không gian bằng vector trọng số W_i và ngưỡng t phân hoạch dữ liệu thành hai nhánh:

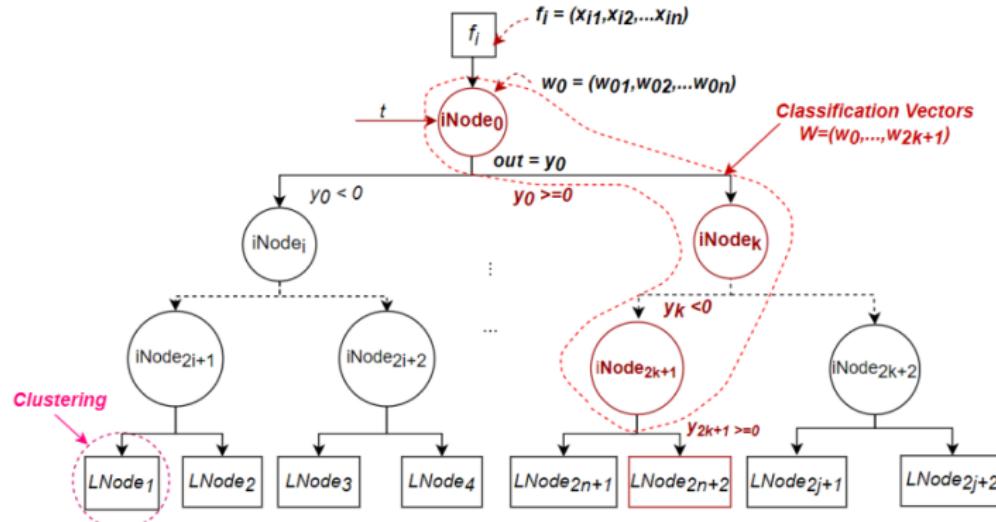
$$S = \text{Sign}(\text{Sigmoid}(W_i \cdot f_i - t) - t).$$

- Nếu $S \geq 0$, vector f_i thuộc nhánh phải.
- Nếu $S < 0$, vector f_i thuộc nhánh trái.

3. Tiếp tục phân hoạch đệ quy đến khi đạt điều kiện dừng: kích thước cụm nhỏ hoặc độ sâu tối đa.

III. Mô hình và các kỹ thuật liên quan

3.1. Xây dựng KD-Tree



Hình 12: Cấu trúc dữ liệu KD-Tree tổng quát.

III. Mô hình và các kỹ thuật liên quan

3.2. Thuật toán truy vấn KD-Tree

1. Với vector truy vấn q , duyệt từ gốc đến nút lá dựa trên trọng số W_i và t để tính giá trị phân hoạch:

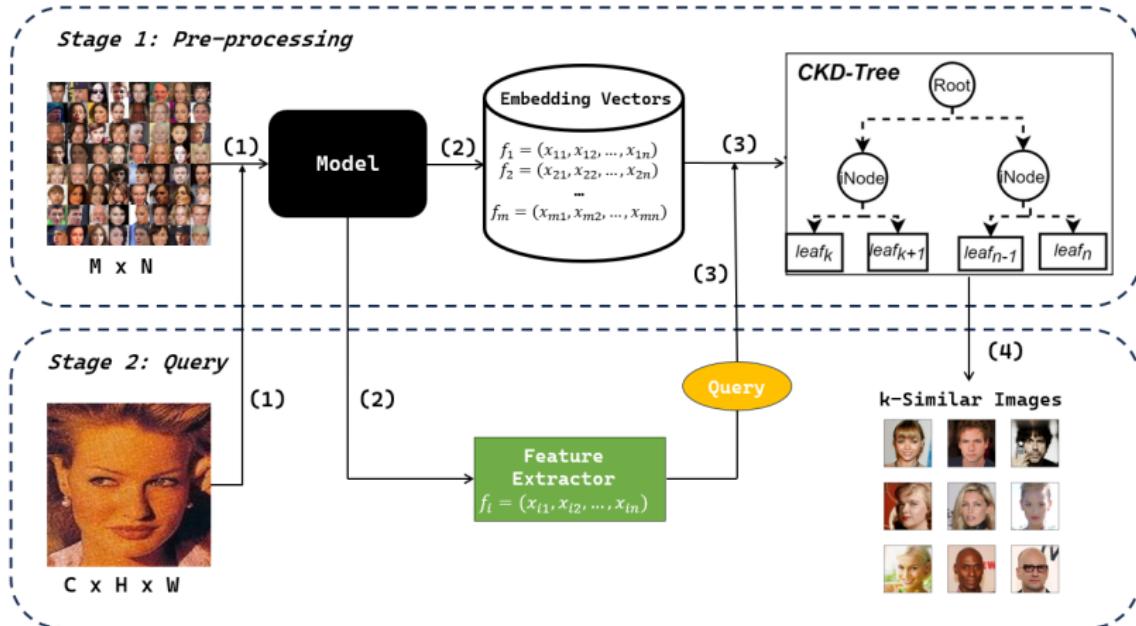
$$S = \text{Sign}(\text{Sigmoid}(W_i \cdot q - t) - t).$$

- Nếu $S \geq 0$, đi vào nhánh phải.
 - Nếu $S < 0$, đi vào nhánh trái.
2. Tại nút lá, tính khoảng cách $d(q, f_i) = \|q - f_i\|_2^2$ và lưu các k -vector gần nhất.
 3. Quay lui để kiểm tra các nhánh có thể chứa kết quả tốt hơn.
 4. Sau khi duyệt xong cây, tập kết quả *Result* chứa các vector f_i trong không gian biểu diễn gần nhất với q :

$$\text{Result} = \bigcup_{LNode} \{f_i \in LNode : f_i \text{ thỏa mãn tiêu chí gần nhất}\}.$$

III. Mô hình và các kỹ thuật liên quan

3.2. Thuật toán truy vấn KD-Tree



Hình 13: Quy trình truy vấn ảnh tổng quát.

III. Mô hình và các kỹ thuật liên quan

3.3. Phương pháp Fine-Tuning

Trong phần này, chúng tôi sẽ điều chỉnh mô hình để tối ưu hóa không gian biểu diễn (*embedding space*) cho bài toán *Face Image Retrieval* dựa trên **Triplet Loss**. Sau đó, việc huấn luyện sẽ được thực hiện trên **10 layers** cuối của từng mô hình.

III. Mô hình và các kỹ thuật liên quan

3.3.1. Triplet-Loss

Mục tiêu của Triplet-Loss là *học* một không gian biểu diễn (*embedding space*) sao cho:

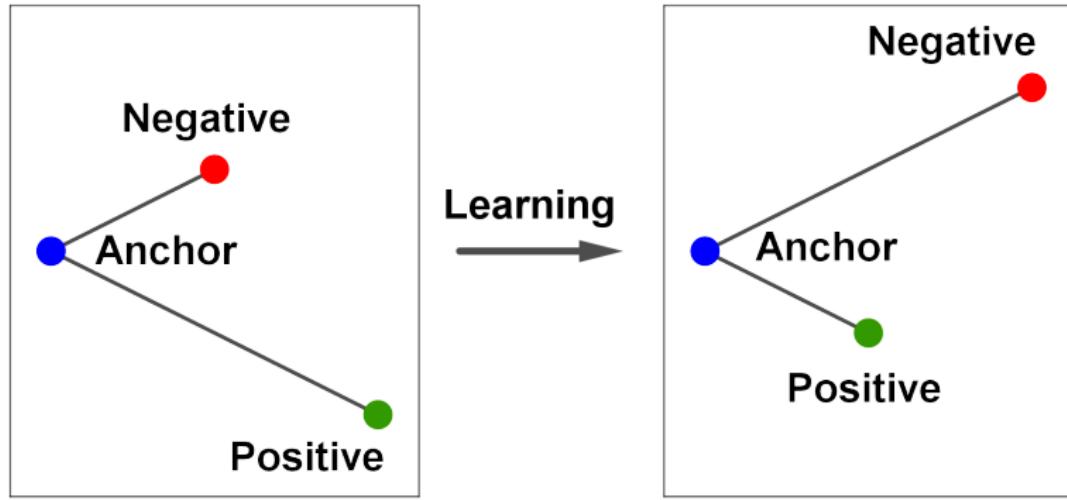
- Ảnh cùng danh tính (*Anchor* và *Positive*) nằm gần nhau.
- Ảnh khác danh tính (*Anchor* và *Negative*) nằm xa nhau ít nhất một khoảng α .

Công thức:

$$\mathcal{L} = \max\left(d(f(a), f(p)) - d(f(a), f(n)) + \alpha, 0\right)$$

III. Mô hình và các kỹ thuật liên quan

3.3.1. Triplet-Loss



Hình 14: Hàm mất mát triplet giảm khoảng cách giữa điểm gốc (anchor) và điểm tích cực (positive), cả hai có cùng danh tính, đồng thời tăng khoảng cách giữa điểm gốc và điểm tiêu cực (negative) có danh tính khác.

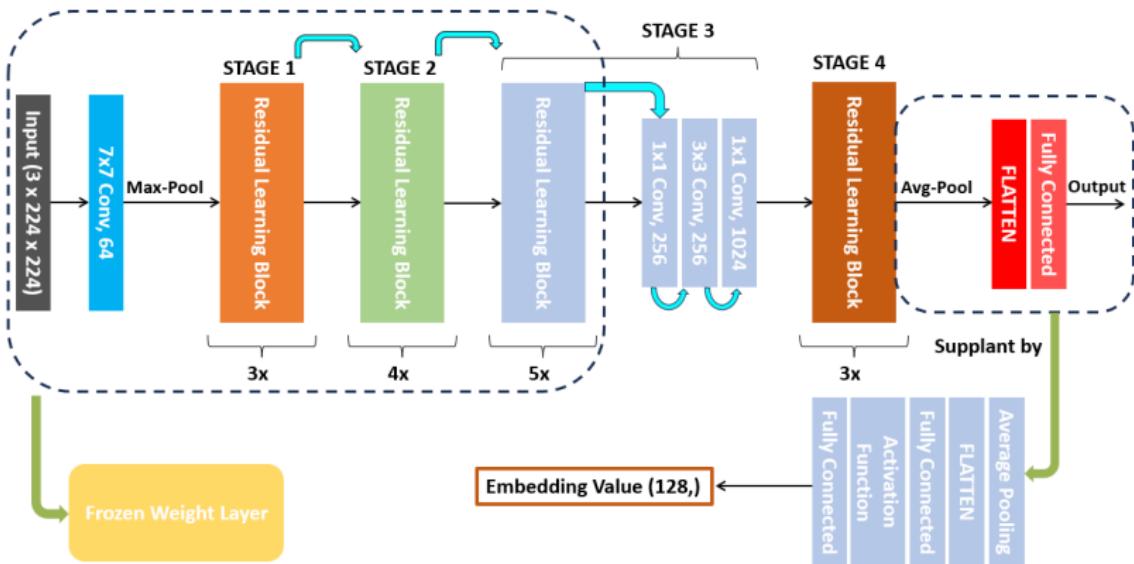
III. Mô hình và các kỹ thuật liên quan

3.3.2. Chiến lược Fine-Tuning

1. **Đóng băng (Freezing) các tầng đầu:** Tận dụng trọng số tiền huấn luyện từ ImageNet để giữ lại các đặc trưng phổ quát (như đường viền, kết cấu) và giảm số tham số cần tối ưu, tăng tốc độ huấn luyện.
2. **Huấn luyện các tầng cao:** Mở khóa (*unfreeze*) 10 layers cuối cùng của mô hình (ResNet-50 và MobileNet-V2) cho việc học các đặc trưng chuyên biệt liên quan đến khuôn mặt từ tập dữ liệu CelebA.
3. **Thay thế lớp cuối:** Thay lớp Fully Connected bằng *Embedding Layer* (128-chiều) để phù hợp với *Triplet Loss*.

III. Mô hình và các kỹ thuật liên quan

3.3.2. Chiến lược Fine-Tuning



Hình 15: Quy trình fine-tuning ResNet-50 cho bài toán Face Image Retrieval.

Đánh giá mô hình

IV. Đánh giá mô hình

1. Phương pháp đánh giá

Đánh giá khả năng truy xuất ảnh của mô hình thông qua các vector biểu diễn (embeddings) và thuộc tính (attributes) của ảnh. Quy trình đánh giá:

1. **Tính vector embeddings:** Trích xuất vector 128-chiều cho tập query và gallery.
2. **Xây dựng KDTree:** Sử dụng embeddings từ gallery để xây dựng KDTree cho truy vấn hiệu quả.
3. **Truy vấn top- k :** Tìm k vector gần nhất trong tập gallery dựa trên khoảng cách Euclidean.
4. **So sánh attributes:** Tính:
 - **Accuracy:** Tỷ lệ thuộc tính khớp giữa ảnh query và ảnh gallery.

$$\text{Accuracy} = \frac{\text{Number of matching attributes}}{\text{Total number of attributes}}.$$

IV. Đánh giá mô hình

1. Phương pháp đánh giá

- **Distance:** Trung bình chênh lệch tuyệt đối giữa các thuộc tính.

$$\text{Distance} = \frac{\sum_j |\text{attr}_{j,\text{query}} - \text{attr}_{j,\text{gallery}}|}{\text{Total number of attributes}}.$$

5. Tổng hợp kết quả:

$$\text{Top-}k \text{ Accuracy} = \frac{1}{k} \sum_{i=1}^k \text{Accuracy}_i,$$

$$\text{Mean Distance} = \frac{1}{k} \sum_{i=1}^k \text{Distance}_i.$$

IV. Đánh giá mô hình

2. Trước khi thực hiện Fine-Tuning

k	ResNet-50			MobileNet-V2		
	Accuracy (%)	Mean Distance	Query Time (s)	Accuracy (%)	Mean Distance	Query Time (s)
$k = 10$	80.19	0.3962	56	79.62	0.4076	54
$k = 20$	79.88	0.4023	56	79.28	0.4143	58
$k = 50$	79.52	0.4097	58	78.96	0.4207	55
$k = 100$	79.22	0.4156	64	78.68	0.4265	63
$k = 1000$	78.07	0.4386	235	77.46	0.4508	233
Trung bình	79.38	0.4125	94s¹ và 93s²	78.8	0.424	93s¹ và 49s²

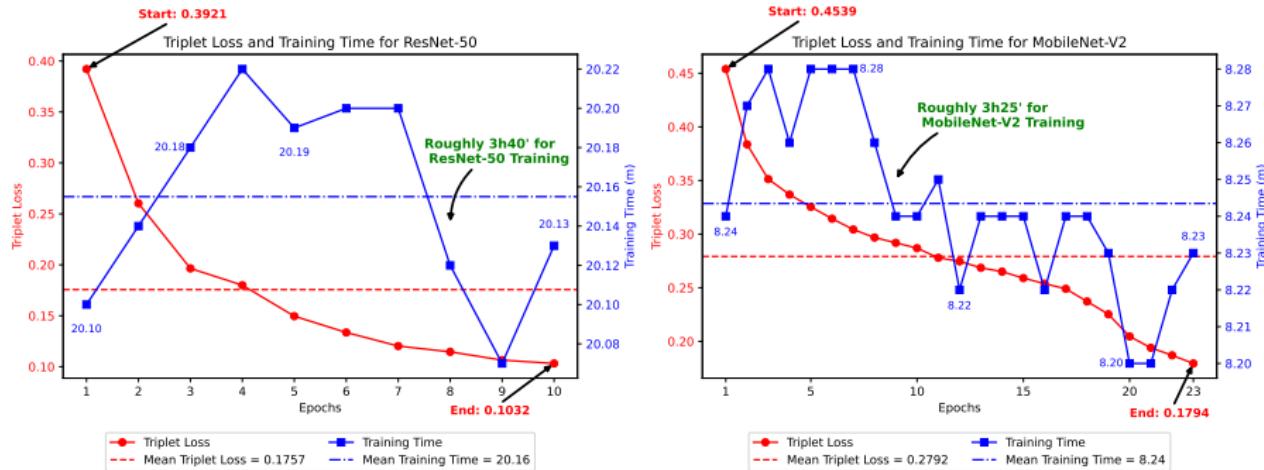
Bảng 1: So sánh hiệu suất truy xuất giữa ResNet-50 và MobileNet-V2 cho các giá trị k khác nhau.

¹Thời gian truy vấn trung bình của tất cả các giá trị k.

²Thời gian tính toán embedding cho toàn bộ tập dữ liệu.

IV. Đánh giá mô hình

3. Quá trình Fine-Tuning



Hình 16: Biểu đồ Triplet Loss và thời gian huấn luyện cho từng mô hình.

IV. Đánh giá mô hình

4. Sau khi thực hiện Fine-Tuning

k	ResNet-50			MobileNet-V2		
	Accuracy (%)	Mean Distance	Query Time (s)	Accuracy (%)	Mean Distance	Query Time (s)
$k = 10$	82.33	0.3534	41	81.70	0.3659	39
$k = 20$	82.11	0.3577	44	81.50	0.3700	41
$k = 50$	81.82	0.3635	43	81.25	0.3750	41
$k = 100$	81.57	0.3685	53	81.05	0.3790	51
$k = 1000$	80.47	0.3906	217	80.11	0.3979	215
Trung bình	81.66	0.3667	80s¹ và 102s²	81.12	0.3776	77s¹ và 49s²

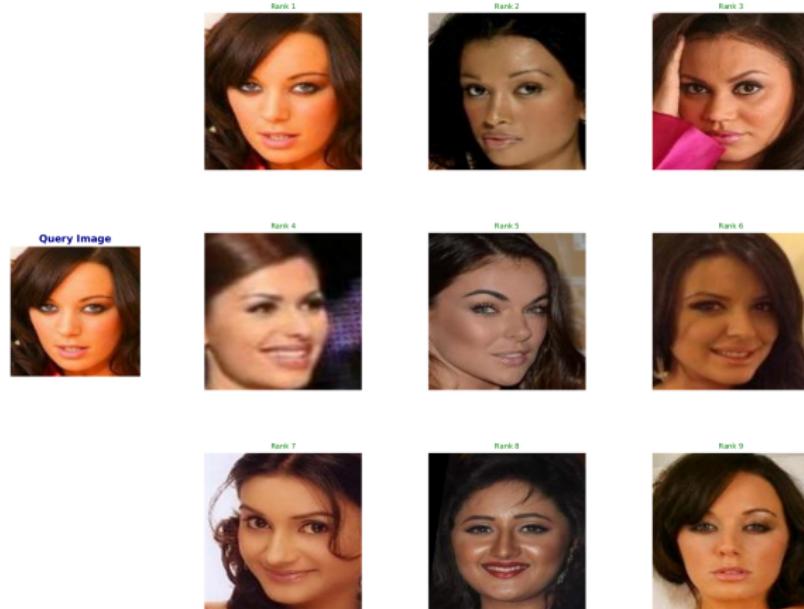
Bảng 2: So sánh hiệu suất truy xuất giữa ResNet-50 và MobileNet-V2 cho các giá trị k khác nhau sau khi thực hiện Fine-Tuning.

¹Thời gian truy vấn trung bình của tất cả các giá trị k.

²Thời gian tính toán embedding cho toàn bộ tập dữ liệu.

IV. Đánh giá mô hình

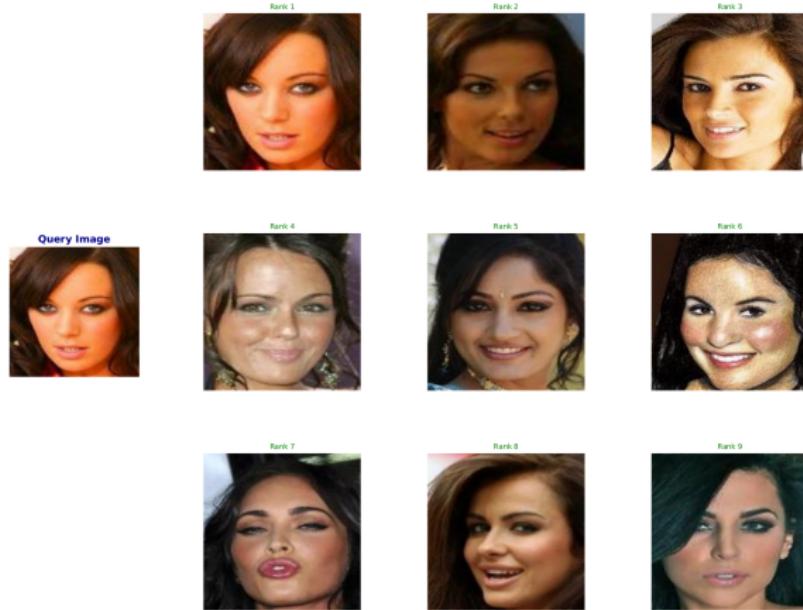
5. Kết quả truy vấn



Hình 17: Kết quả truy vấn hình ảnh dựa trên mô hình Fine-Tuned Resnet-50.

IV. Đánh giá mô hình

5. Kết quả truy vấn



Hình 18: Kết quả truy vấn hình ảnh dựa trên mô hình Fine-Tuned MobileNet-V2.

DEMO - Query Interface Design

Lời kết