

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO ĐỒ ÁN  
Face Recognition Using Near Infrared  
Images

MÔN: NHẬN DẠNG

Giảng viên hướng dẫn: PGS.TS. Lê Hoàng Thái  
Thầy Dương Thái Bảo  
Nhóm sinh viên thực hiện: Nhóm 07:  
Trần Gia Hào - 22120099  
Nguyễn Minh Hưng - 22120123  
Nguyễn Tấn Hưng - 22120126  
Hà Đức Huy - 22120133

Thành phố Hồ Chí Minh, năm 2025

# MỤC LỤC

<b>1</b>	<b>Phân công công việc</b>	<b>3</b>
<b>2</b>	<b>Nội dung chương sách</b>	<b>3</b>
2.1	Introduction	3
2.2	Active NIR Imaging System	4
2.3	Illumination Invariant Face Representation	7
2.3.1	Modeling of Active NIR Images	7
2.3.2	Compensation for Monotonic Transform	8
2.4	NIR Face Classification	10
2.4.1	AdaBoost Based Feature Selection	10
2.4.2	LDA Classifier	11
2.5	Experiments	12
2.5.1	Basic Evaluation	12
2.5.2	Weak Illumination	14
2.5.3	Eyeglasses	15
2.5.4	Time Lapse	16
2.5.5	Outdoor environment	17
2.6	Conclusions	19
<b>3</b>	<b>Phương pháp tiên tiến</b>	<b>20</b>
3.1	Giới thiệu	20
3.2	Phương pháp thực hiện	20
3.3	Phát hiện khuôn mặt	21
3.4	Nhận diện khuôn mặt	22
3.4.1	Mô hình VGG-Face:	22
3.4.2	Bộ phân loại tùy chỉnh:	23
3.5	Kết quả:	23
<b>4</b>	<b>Thực nghiệm:</b>	<b>24</b>
4.1	Chuẩn bị dữ liệu	24
4.1.1	Chia tập dữ liệu	24
4.1.2	Tăng cường dữ liệu (Data Augmentation)	25
4.1.3	Thông kê sau khi tăng cường dữ liệu	25
4.2	Face Detection	25
4.2.1	Importing phương pháp face_detection	25

4.2.2	Xây dựng kiến trúc mô hình VGG-Face . . . . .	27
4.2.3	Nạp trọng số mô hình VGG-Face và loại bỏ lớp Softmax . . . . .	30
4.2.4	Thực hiện crop images bằng YOLOFace . . . . .	31
4.3	Face Recognition . . . . .	32
4.3.1	Chia tập huấn luyện, kiểm tra và kiểm thử . . . . .	32
4.3.2	Chuẩn bị dữ liệu huấn luyện (train) . . . . .	33
4.3.3	Chuẩn bị dữ liệu kiểm thử (validation) và dữ liệu kiểm tra (test)	34
4.3.4	Lưu trữ và tải lại dữ liệu đã xử lý . . . . .	35
4.3.5	Xây dựng mô hình phân loại Softmax dựa trên đặc trưng khuôn mặt đã trích xuất từ mô hình VGG-Face. . . . .	36
4.3.6	Huấn luyện mô hình dựa trên tập train và đánh giá mô hình trên tập valid và test . . . . .	37
4.4	Thử nghiệm mô hình trên ảnh khuôn mặt bất kỳ . . . . .	39
	Tài liệu tham khảo . . . . .	41

# 1 Phân công công việc

STT	Nhiệm vụ	Người thực hiện	MSSV	Đánh giá
1	Đọc phần 15.3 và 15.4 của chương sách Thực hiện báo cáo phần 15.3 và 15.4 Thực hiện phần Face Detection	Trần Gia Hào	22120099	100%
2	Tìm kiếm phương pháp tiên tiến Thực hiện báo cáo phần phương pháp tiên tiến Thực hiện phần Face Recognition	Nguyễn Minh Hưng	22120123	100%
3	Đọc phần 15.5 và 15.6 của chương sách Thực hiện báo cáo phần 15.5 và 15.6 Thực hiện huấn luyện và kiểm tra mô hình	Nguyễn Tấn Hưng	22120126	100%
4	Đọc phần 15.1 và 15.2 của chương sách Thực hiện báo cáo phần 15.1 và 15.2 Thực hiện tìm kiếm và chuẩn bị dữ liệu	Hà Đức Huy	22120133	100%

Bảng 1: Bảng phân công công việc

## 2 Nội dung chương sách

### 2.1 Introduction

Nhận dạng khuôn mặt nên dựa trên các yếu tố nội tại của khuôn mặt, chẳng hạn như hình dạng 3D và độ phản xạ (albedo) của bề mặt khuôn mặt. Các yếu tố ngoại tại bao gồm điều kiện chiếu sáng, kính đeo mắt và kiểu tóc không liên quan đến đặc điểm sinh trắc học định danh và do đó ảnh hưởng của chúng nên được giảm thiểu. Trong số tất cả các yếu tố này, sự thay đổi về chiếu sáng là một thách thức lớn và cần được giải quyết trước tiên.

Các hệ thống nhận dạng khuôn mặt dựa trên hình ảnh ánh sáng nhìn thấy (VIS) thông thường, thường bị ảnh hưởng về độ chính xác do thay đổi ánh sáng môi trường, dù là trong các tình huống có người dùng hợp tác và ánh sáng trong nhà ổn định. Các nghiên cứu đã đưa ra một số kết luận:

- Điều kiện chiếu sáng, đặc biệt là góc chiếu sáng, làm thay đổi đáng kể diện mạo của khuôn mặt
- Khi so sánh các ảnh chưa qua xử lý, sự khác biệt giữa ảnh của cùng một người dưới điều kiện ánh sáng khác nhau còn lớn hơn cả sự khác biệt giữa hai người khác nhau dưới cùng điều kiện ánh sáng
- Tất cả các bộ lọc cục bộ được nghiên cứu đều không thể vượt qua được biến đổi do thay đổi hướng ánh sáng gây ra. Ảnh hưởng của ánh sáng cũng được thể hiện rõ trong các đánh giá như Bài kiểm tra nhà cung cấp hệ thống nhận dạng khuôn mặt (Face Recognition Vendor Test).

Nhận dạng khuôn mặt dựa trên ánh sáng hồng ngoại gần (Near Infrared – NIR), trái ngược với phương pháp truyền thống dựa trên ánh sáng nhìn thấy (visible light - VIS), là một cách tiếp cận hiệu quả nhằm vượt qua ảnh hưởng của thay đổi chiếu sáng đối với nhận dạng khuôn mặt. Phương pháp này sử dụng thiết bị chụp ảnh chuyên dụng để thu nhận ảnh khuôn mặt hồng ngoại gần được chiếu sáng từ phía trước, giúp chuẩn hóa hướng chiếu sáng. Bằng cách sử dụng phương pháp biểu diễn đặc trưng khuôn mặt phù hợp, chẳng hạn như mẫu nhị phân cục bộ (Local Binary Pattern - LBP), sự biến đổi về cường độ ánh sáng cũng được khắc phục. Những điều này dẫn đến việc biểu diễn khuôn mặt hoàn toàn không phụ thuộc vào độ chiếu sáng. Vì thế, các vấn đề gây ra bởi điều kiện, môi trường chiếu sáng không kiểm soát đã được giảm thiểu, và những khó khăn trong việc xây dựng hệ thống đối sánh khuôn mặt cũng được giảm xuống. **Phương pháp NIR thường đạt được hiệu suất cao hơn đáng kể so với phương pháp VIS trong các ứng dụng có người dùng hợp tác nhưng điều kiện chiếu sáng không được kiểm soát.** Các sản phẩm nhận dạng khuôn mặt NIR và hệ thống tích hợp đã có mặt trên thị trường và được sử dụng trong nhiều ứng dụng khác nhau.

## 2.2 Active NIR Imaging System

**Yếu tố then chốt trong phương pháp nhận dạng khuôn mặt bằng ánh sáng hồng ngoại gần (NIR) là một hệ thống phần cứng chụp ảnh NIR chuyên dụng.** Mục tiêu của hệ thống này là khắc phục vấn đề gây ra bởi ánh sáng môi trường không kiểm soát được gây ra và tạo ra các ảnh khuôn mặt có điều kiện chiếu sáng tốt, phù hợp cho việc nhận dạng khuôn mặt. Để đạt được điều kiện chiếu sáng tốt phải bao gồm 2 yếu tố là:

- Ánh sáng chiếu vào khuôn mặt đến từ hướng chính diện
- Ảnh khuôn mặt thu được có cường độ điểm ảnh (pixel) phù hợp.

Các yêu cầu chính đối với hệ thống chụp ảnh NIR bao gồm:

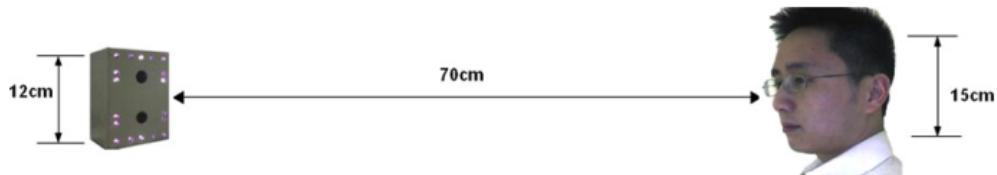
- Đèn chiếu sáng NIR chủ động không được gây khó chịu cho mắt người.
- Hướng chiếu của ánh sáng NIR vào khuôn mặt cần phải được cố định.
- Tín hiệu ánh sáng NIR chủ động đến cảm biến camera phải lấn át tín hiệu từ các nguồn sáng môi trường khác.

Ở đây, ánh sáng NIR được đề cập là ánh sáng NIR phát ra từ hệ thống chụp ảnh NIR chủ động, không bao gồm các thành phần NIR tự nhiên như ánh sáng mặt trời hay bóng đèn.

Để đạt được các yêu cầu ở trên cho hệ thống chụp ảnh NIR, có thể sử dụng các phương pháp sau:

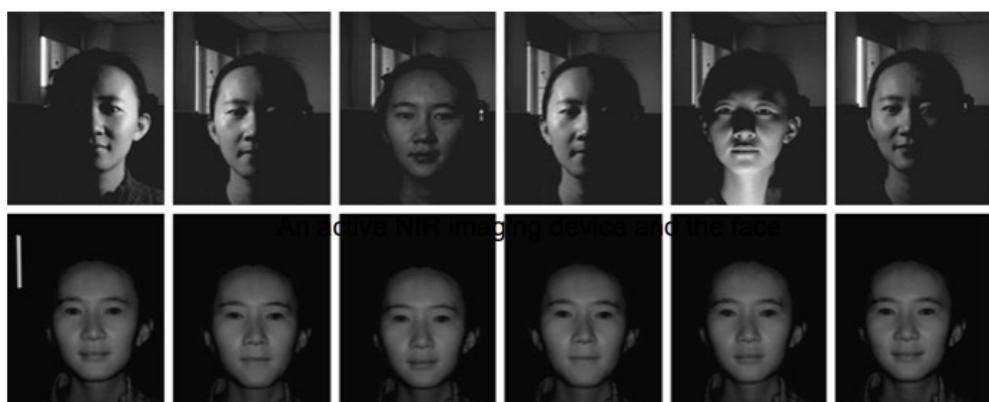
- Chọn các nguồn sáng như LED hoạt động ở phổ ánh sáng không nhìn thấy được. Ví dụ, LED NIR ở bước sóng 850 nm sẽ phát ra ánh sáng đỏ nhạt mờ, còn 940 nm thì hoàn toàn vô hình.

- Lắp đặt các đèn LED NIR xung quanh ống kính máy ảnh để đảm bảo ánh sáng chiếu vào khuôn mặt từ hướng chính diện.
- Chọn đèn LED NIR có công suất đủ mạnh để lấn át các nguồn sáng môi trường có thể ảnh hưởng đến hệ thống. Môi trường thuận lợi nhất là trong bóng tối hoàn toàn, còn môi trường khó khăn nhất là ánh sáng mặt trời vào mùa hè. Cần sử dụng thời gian phơi sáng ngắn để tránh bị lóa khi đèn LED có công suất cao.
- Sử dụng bộ lọc quang học để giảm thiểu ánh sáng từ môi trường. Một lựa chọn là bộ lọc thông dài (long pass filter) để loại bỏ các thành phần ánh sáng nhìn thấy. Lựa chọn tốt hơn nhưng đắt hơn là bộ lọc thông dải hẹp (narrow band pass filter) phù hợp với bước sóng của đèn LED chủ động đã chọn.



Hình 1: Một thiết bị chụp ảnh NIR chủ động và quan hệ giữa nó với khuôn mặt

**Hình 1** minh họa một hệ thống chụp ảnh NIR dựa vào ý tưởng ở trên và mối quan hệ vị trí của nó với khuôn mặt.



Hình 2: Sự khác biệt giữa ảnh màu và ảnh NIR

**Hình 2** cho thấy các ảnh khuôn mặt được chiếu sáng bằng đèn LED NIR từ phía trước, bằng đèn bàn bên cạnh, và bằng ánh sáng môi trường. Có thể thấy rằng:

- Các điều kiện chiếu sáng như vậy có thể gây vấn đề cho việc nhận dạng khuôn mặt từ ảnh màu.
- Các ảnh NIR, sau khi đã loại bỏ thành phần ánh sáng nhìn thấy nhờ bộ lọc, hầu hết đều được chiếu sáng từ phía trước bằng ánh sáng NIR, và gần như không bị ảnh hưởng bởi ánh sáng từ bên hông.

Dựa vào **Hình 1** và **Hình 2**, ta có thể đưa ra nhận xét sau: Trong môi trường ngoài trời, ánh sáng mặt trời chứa thành phần NIR rất mạnh, mạnh hơn nhiều so với khả năng lấn át của hệ thống chụp ảnh NIR được mô tả ở trên. Do đó, hệ thống chụp ảnh NIR cần được nâng cấp để khắc phục ảnh hưởng từ ánh sáng mặt trời.

Từ những nhận xét trên, một giải pháp cho hệ thống chụp ảnh NIR nâng cấp (ENIR) là sử dụng đèn flash NIR công suất lớn và đồng bộ thời gian chiếu sáng của đèn NIR với thời điểm phơi sáng của cảm biến ảnh. Những điểm chính của ENIR bao gồm:

- Sử dụng nguồn chiếu sáng NIR công suất lớn như máy phát laser NIR dải hẹp.
- Thiết lập thời gian phơi sáng cực ngắn, khoảng 50 micro giây ( $\mu\text{s}$ ).
- Đồng bộ thời gian chiếu sáng của đèn NIR với cửa sổ phơi sáng của cảm biến ảnh.
- Sử dụng bộ lọc quang học thông dải hẹp phù hợp với bước sóng của đèn flash NIR chủ động đã chọn.

Hệ thống chụp ảnh NIR nâng cao (ENIR) cũng sẽ **bao gồm một bước xử lý bổ sung** để giảm thiểu thêm ảnh hưởng từ các nguồn sáng mạnh và không kiểm soát được trong môi trường. Điều này được thực hiện bằng cách **lấy hiệu của hai khung hình liên tiếp**:

- Khung hình đầu tiên được chụp khi chiếu sáng khuôn mặt bằng nguồn sáng NIR chủ động
- Khung hình thứ hai được chụp sau khi tắt nguồn sáng đó, phản ánh hình ảnh khuôn mặt dưới điều kiện ánh sáng tĩnh của môi trường.

Thao tác lấy hiệu của hai khung hình liên tiếp **giúp giảm hoặc loại bỏ thành phần NIR mạnh có trong ánh sáng môi trường hoặc ánh sáng mặt trời, từ đó cho ra ảnh khuôn mặt được chiếu sáng bởi nguồn NIR chủ động từ phía trước**. Kết quả là, hệ thống ENIR không chỉ cung cấp ánh sáng chiếu chính diện thích hợp mà còn giảm thiểu ánh sáng môi trường cũng như ánh sáng mặt trời ngoài trời.



Hình 3: Từ trái sang phải: a. Ảnh khuôn mặt dưới sự kết hợp của ánh sáng mặt trời và ánh sáng hồng ngoại gần chủ động; b. Ảnh khuôn mặt chỉ có ánh sáng mặt trời; c. Ảnh hiệu số (a) – (b)

**Hình 3** trình bày một số ảnh được chụp khi bật/tắt nguồn sáng chủ động, cùng với ảnh đầu ra cuối cùng của camera ENIR dưới ánh sáng mặt trời. Từ **Hình 3(c)**, ta có thể thấy rằng hệ thống chụp ảnh ENIR có thể hoạt động hiệu quả dưới ánh sáng mặt trời.

## 2.3 Illumination Invariant Face Representation

### 2.3.1 Modeling of Active NIR Images

Theo mô hình Lambertian, một ảnh  $I(x, y)$  dưới nguồn sáng điểm được mô tả như sau:

$$I(x, y) = \rho(x, y) \cdot n(x, y) \cdot s \quad (1)$$

Trong đó:

- $\rho(x, y)$  là hệ số phản xạ (albedo) của vật liệu bề mặt khuôn mặt tại điểm  $(x, y)$ ,
- $n(x, y) = (n_x, n_y, n_z)$  là vector đơn vị pháp tuyến của bề mặt tại điểm  $(x, y)$ ,
- $s = (s_x, s_y, s_z)$  là vector cột biểu diễn hướng chiếu sáng, có độ lớn.

**Giải thích:**

- $\rho(x, y)$  thể hiện tính chất quang học (photometric) của da và tóc trên khuôn mặt,
- $n(x, y)$  thể hiện hình dạng hình học (3D) của khuôn mặt,
- Tích  $\rho(x, y) \cdot n(x, y)$  là thuộc tính *nội tại* của khuôn mặt tại tư thế cố định, và là yếu tố cần thiết cho nhận dạng,
- $s$  là thuộc tính *ngoại tai* (extrinsic) cần được loại bỏ.

Giả sử hướng chiếu sáng có thể được viết dưới dạng:

$$s = \kappa s^0 \quad (2)$$

với:

- $\kappa$  là một hằng số nhân đại diện cho độ mạnh yếu của nguồn sáng (do khoảng cách thay đổi),
- $s^0 = (s_x^0, s_y^0, s_z^0)$  là vector đơn vị hướng chiếu sáng.

Gọi  $\theta(x, y)$  là góc giữa ánh sáng tới và pháp tuyến bề mặt tại điểm  $(x, y)$ , ta có:

$$\cos \theta(x, y) = n(x, y) \cdot s^0 \quad (3)$$

Như vậy, biểu thức có thể viết lại dưới dạng:

$$I(x, y) = \kappa \rho(x, y) \cos \theta(x, y)$$

**Mục đích của các phép biến đổi trên:**

**1. Cải thiện hiệu suất nhận diện khuôn mặt:**

Sử dụng ánh sáng NIR chủ động và mô hình Lambertian (phương trình 1) để kiểm soát hướng ánh sáng  $s$ , giảm ảnh hưởng của ánh sáng môi trường (yếu tố ngoại tại). Điều này giúp bảo toàn các đặc tính nội tại như  $\rho(x, y)$  và  $\mathbf{n}(x, y)$ , tăng độ chính xác nhận diện khuôn mặt trong điều kiện ánh sáng khác nhau.

**2. Tách biệt đặc tính nội tại của khuôn mặt:**

Tách  $\rho(x, y) \cdot \mathbf{n}(x, y)$  (đặc tính nội tại) khỏi  $s$  (đặc tính ngoại tại) bằng cách điều chỉnh  $\kappa$  và  $s^0$ , hỗ trợ nhận diện khuôn mặt hiệu quả hơn.

Tóm lại, việc áp dụng mô hình và các phép biến đổi này giúp tăng độ chính xác và ổn định nhận diện khuôn mặt bằng cách giảm tác động của ánh sáng môi trường và bảo toàn đặc tính nội tại.

### 2.3.2 Compensation for Monotonic Transform

Đối với một khuôn mặt, hằng số  $\kappa$  là yếu tố duy nhất ảnh hưởng đến cường độ của ảnh khuôn mặt. Biến đổi đơn điệu này về cường độ có thể được hiệu chỉnh bằng các phương pháp như cân bằng histogram (histogram equalization), đặc tả histogram (histogram specification), hoặc các đặc trưng bất biến với biến đổi đơn điệu.

Mức độ tự do trong  $\kappa$ , hay trong một biến đổi đơn điệu, có thể được bù trừ bằng cách sử dụng các đặc trưng LBP để đạt được biểu diễn khuôn mặt bất biến với ánh sáng. Các bit nhị phân mô tả một vùng con  $3 \times 3$  được tạo ra bằng cách so sánh giá trị xám của 8 điểm ảnh xung quanh với giá trị xám tại điểm ảnh trung tâm; vector đặc trưng được tạo ra bằng cách nối các bit nhị phân này theo chiều ngược kim đồng hồ. Có tổng cộng 256 giá trị có thể xảy ra, tương ứng với 256 mẫu LBP, được gọi là mã LBP; mỗi giá trị đại diện cho một kiểu mẫu LBP cục bộ. Dạng cơ bản này có thể mở rộng thành LBP đa tỉ lệ, gọi là LBP( $P, R$ ), trong đó  $R$  là bán kính của vòng tròn bao quanh điểm trung tâm và  $P$  là số điểm ảnh nằm trên vòng tròn. Một chuỗi LBP( $P, R$ ) được gọi là "đồng nhất" (uniform), ký hiệu là  $LBP_{(P,R)}^{u2}$ , nếu các bit lân cận (theo chiều vòng tròn) chỉ chứa tối đa hai lần chuyển trạng thái từ 0 sang 1 hoặc ngược lại.

Local Window	Thresholded	Weights																											
<table border="1"> <tr><td>18</td><td>15</td><td>8</td></tr> <tr><td>21</td><td>18</td><td>6</td></tr> <tr><td>27</td><td>23</td><td>22</td></tr> </table>	18	15	8	21	18	6	27	23	22	<table border="1"> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td></tr> </table>	1	0	0	1	0	0	1	1	1	<table border="1"> <tr><td>8</td><td>4</td><td>2</td></tr> <tr><td>16</td><td>0</td><td>1</td></tr> <tr><td>32</td><td>64</td><td>128</td></tr> </table>	8	4	2	16	0	1	32	64	128
18	15	8																											
21	18	6																											
27	23	22																											
1	0	0																											
1	0	0																											
1	1	1																											
8	4	2																											
16	0	1																											
32	64	128																											

**LBP String = (0001111)**

**LBP Code = 0+0+0+8+16+32+64+128=248**

Hình 4: LBP code cho cửa sổ 3x3 window

Việc kết hợp giữa ảnh NIR và các đặc trưng LBP sẽ tạo ra một biểu diễn khuôn mặt bất biến với ánh sáng. Nói cách khác, khi áp dụng toán tử LBP lên ảnh NIR chủ động, ta thu được các đặc trưng khuôn mặt không bị ảnh hưởng bởi sự thay đổi của ánh sáng. Biểu diễn khuôn mặt bất biến với ánh sáng này mang lại những lợi thế lớn trong việc nhận dạng khuôn mặt dưới điều kiện ánh sáng thay đổi.

**Quy trình trích xuất đặc trưng histogram LBP được mô tả như sau:**

1. Tính toán đặc trưng LBP cơ bản

- Tính giá trị  $LBP_{8,1}^{u2}$  cho mỗi giá trị pixel trong ảnh

2. Lập biểu đồ histogram cho LBP

- Một histogram của các mã LBP cơ bản được tính trong một vùng lân cận tại mỗi điểm ảnh, mỗi cột của histogram thể hiện số lần xuất hiện của mã LBP tương ứng trong vùng lân cận đó. Có tổng cộng 59 cột cho  $LBP_{8,1}^{u2}$ .
- Một histogram LBP được coi như là tập hợp của 59 đặc trưng riêng biệt.

3. Thu thập các histogram LBP

- Với ảnh khuôn mặt có kích thước  $W \times H$  và vùng bên trong có kích thước  $W' \times H'$ , tổng số đặc trưng histogram LBP là  $D = W' \times H' \times 59$  (tức là số vị trí điểm ảnh hợp lệ nhân với số lượng cột histogram LBP)

**Mục đích của việc làm này:**

1. **Bù trừ biến đổi đơn điệu về cường độ:**

Sử dụng đặc trưng LBP để bù cho biến đổi đơn điệu do hằng số  $\kappa$ , tạo biểu diễn khuôn mặt bất biến với ánh sáng, giảm ảnh hưởng của thay đổi cường độ ánh sáng.

2. **Cải thiện nhận diện khuôn mặt trong điều kiện ánh sáng thay đổi:**

Kết hợp ảnh NIR với đặc trưng LBP (ví dụ:  $LBP_{8,1}^{u2}$ ) để trích xuất đặc trưng không phụ thuộc ánh sáng, tăng độ chính xác và ổn định của nhận diện khuôn mặt.

Tóm lại, nhờ áp dụng bù cho biến đổi đơn điệu với đặc trưng LBP, mô hình hóa hình ảnh NIR chủ động được cải thiện, giúp nhận diện khuôn mặt chính xác và ổn định hơn trong điều kiện ánh sáng thay đổi.

## 2.4 NIR Face Classification

Trong số lượng lớn các đặc trưng histogram LBP có trong tập đặc trưng, một số thì hữu ích cho nhận diện khuôn mặt, một số khác thì không hữu ích lầm, và thậm chí một số có thể gây mâu thuẫn. Do đó, chúng cần được lựa chọn hoặc gán trọng số để đạt được hiệu suất tốt nhất

### 2.4.1 AdaBoost Based Feature Selection

Vì quá trình AdaBoost thực chất học một bộ phân loại hai lớp, nên nên chuyển bài toán phân lớp nhiều lớp thành bài toán hai lớp bằng cách sử dụng khái niệm về sự khác biệt trong-lớp (intra-class - đặc trưng của hai khuôn mặt thuộc cùng một người) và ngoài-lớp (extra-class - đặc trưng của hai khuôn mặt thuộc hai người khác nhau).

Giả sử ta có một tập huấn luyện gồm N ví dụ thuộc hai lớp dương và âm:

$$S = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

Trong đó:

- $x_i$  là một ví dụ huấn luyện (sự khác biệt giữa hai vector đặc trưng)
- $y_i \in \{+1, -1\}$  là nhãn lớp

Thuật toán AdaBoost có thể được sử dụng để học một tập hợp T đặc trưng tốt nhất theo từng giai đoạn, từ đó xây dựng một chuỗi các bộ phân loại yếu  $h_t(x) \in \{+1, -1\}$ , và kết hợp tuyến tính các bộ phân loại yếu này thành một bộ phân loại mạnh hơn:

$$H(x) = \text{sign}\left(\sum_{t=1}^T \alpha_t h_t(x)\right)$$

Trong đó:

- $\alpha_t \in R$  là trọng số kết hợp

AdaBoost giả định rằng có một phương pháp để học một bộ phân loại yếu  $h_t(x)$  từ các ví dụ huấn luyện được gán trọng số theo phân phối hiện tại  $w_t$ . Trong trường hợp này, ta sử dụng một bộ phân loại yếu dựa trên giá trị của một đặc trưng đơn lẻ, tức là một giá trị histogram của LBP. Do đó, khi AdaBoost xây dựng  $h_t(x)$ , nó cần chọn một đặc trưng tốt để sử dụng. Theo cách này, AdaBoost có thể cung cấp một tập hợp con các đặc trưng tốt nhất.

### Giai đoạn kiểm tra (test phase)

Trong giai đoạn kiểm tra, hàm phân loại  $H(x)$  đã được học có thể được dùng để phân loại ảnh khuôn mặt. Sự khác biệt được tính trên các đặc trưng đã được chọn giữa hai ảnh khuôn mặt. Mỗi đặc trưng sẽ đưa ra một quyết định yếu  $h_t(x)$ . Các quyết định yếu này được kết hợp tuyến tính với trọng số  $\alpha_t$  để tạo ra giá trị dự đoán  $H(x)$ . Quyết định cuối cùng được đưa ra bằng cách so sánh  $H(x)$  với một ngưỡng. Nếu lớn hơn ngưỡng, hai ảnh được coi là của cùng một người (intra-class), nếu không thì là người khác nhau (inter-class).

### Lợi ích:

- Giảm số lượng đặc trưng cần dùng, từ đó tăng tốc độ tính toán và giảm độ phức tạp mô hình.
- Tăng độ chính xác bằng cách tập trung vào những đặc trưng có khả năng phân biệt cao nhất.

#### 2.4.2 LDA Classifier

LDA giảm chiều dữ liệu bằng cách chiếu tuyến tính vector đặc trưng gốc trong không gian nhiều chiều xuống một không gian con có chiều thấp hơn sao cho tỷ lệ giữa phân tán trong-lớp và phân tán ngoài-lớp được tối thiểu hóa. Việc tối thiểu hóa tỷ lệ này giúp giảm lỗi phân loại, đặc biệt khi phân phối dữ liệu của các lớp tuân theo phân phối chuẩn (Gaussian)

Dữ liệu có chiều cao thường được tiền xử lý bằng PCA trước khi áp dụng LDA, để đảm bảo ma trận phân tán trong-lớp là khả nghịch (không suy biến). Các ảnh cơ sở (basis images) thu được từ phép chiếu kết hợp giữa PCA và LDA được gọi là Fisherfaces.

Các biến thể nâng cao của LDA như LDA trực tiếp (Direct LDA) hoặc LDA điều chỉnh (Regularized LDA) cũng có thể được sử dụng để tìm ra ma trận chiếu P. Trong bối cảnh này, đầu vào của LDA là không gian đặc trưng đã được chọn lọc (ví dụ: thông qua AdaBoost).

Giả sử có hai vector đặc trưng đầu vào  $x_1$  và  $x_2$  trong không gian đã chọn, phép chiếu LDA được tính như sau:

$$v_1 = Px_1, v_2 = Px_2$$

Sau đó, độ tương đồng cosine (cosine score) giữa hai vector chiếu  $v_1$  và  $v_2$  được tính bằng:

$$H(v_1, v_2) = \frac{v_1 \cdot v_2}{\|v_1\| \cdot \|v_2\|}$$

Trong giai đoạn kiểm tra (test phase), hai vector chiếu  $v_1$  và  $v_2$  được tính từ hai ảnh khuôn mặt đầu vào  $x_1$  và  $x_2$ , một cái là ảnh mới, một cái là ảnh đã đăng ký sẵn (enrolled). Bằng cách so sánh giá trị  $H(v_1, v_2)$  với một ngưỡng, ta có thể quyết định liệu  $x_1$  và  $x_2$  có phải là cùng một người hay không.

Những lợi ích khi áp dụng LDA trong phân loại khuôn mặt hồng ngoại (NIR):

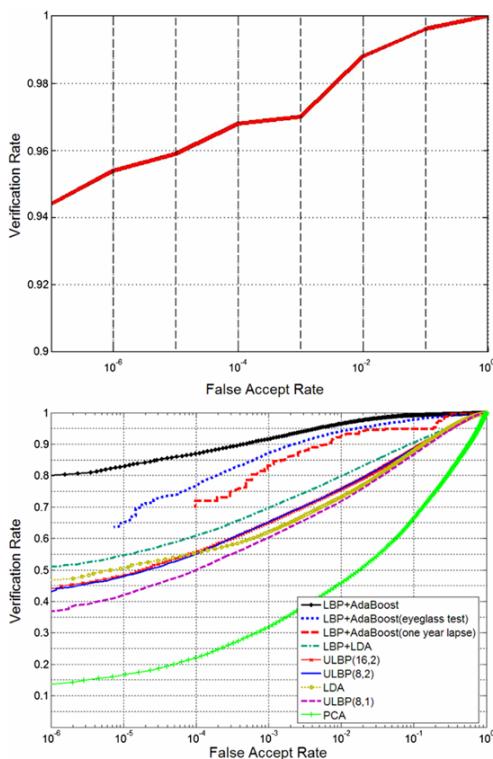
- **Tăng độ phân biệt giữa các đối tượng:** LDA tối ưu không gian chiết để làm nổi bật sự khác biệt giữa các khuôn mặt khác nhau trong điều kiện ánh sáng hồng ngoại.
- **Giảm chiều dữ liệu:** Giúp tăng tốc độ xử lý và giảm nhiễu trong dữ liệu đầu vào NIR, vốn thường có độ phân giải thấp và nhiều biến thiên.
- **Tăng độ chính xác nhận diện:** Không gian đặc trưng sau chiết giúp so sánh hai ảnh NIR hiệu quả hơn bằng các chỉ số như cosine similarity.

Do đó, việc áp dụng LDA giúp tăng cường hiệu suất phân loại, đặc biệt trong môi trường nhận diện khuôn mặt dựa trên ảnh hồng ngoại có điều kiện ánh sáng phức tạp.

## 2.5 Experiments

Phần này trình bày các thí nghiệm được thực hiện để đánh giá hiệu suất của hệ thống nhận diện khuôn mặt dựa trên NIR, với các điều kiện thử nghiệm khác nhau. Các thí nghiệm tập trung vào các yếu tố như ánh sáng yếu, đeo kính, khoảng cách thời gian, và môi trường ngoài trời. Các phương pháp được so sánh bao gồm **LBP + AdaBoost**, **LBP + LDA**, và các phương pháp cơ sở như PCA và LDA trên hình ảnh NIR.

### 2.5.1 Basic Evaluation



Hình 5: **Trên:** Đường cong ROC của phương pháp LBP kết hợp với AdaBoost trong bài toán xác thực khuôn mặt trên tập huấn luyện. **Dưới:** Các đường cong ROC của các phương pháp được so sánh.

- **Tập huấn luyện:**

- Gồm  $10^4$  hình ảnh khuôn mặt của khoảng 1000 người (mỗi người 10 ảnh), tất cả đều là người Trung Quốc.
- Đặc trưng LBP được trích xuất, tạo ra các vector đặc trưng chiều cao (748592 chiều).
- Tập huấn luyện bao gồm khoảng  $45 \times 10^3$  cặp trong lớp (*positive*, cùng một người) và  $5 \times 10^7$  cặp ngoài lớp (*negative*, khác người).
- Một chuỗi 5 bộ phân loại mạnh (*strong classifiers*) được huấn luyện với khoảng 1500 bộ phân loại yếu (*weak classifiers*) bằng thuật toán AdaBoost.
- **Kết quả:** Đường cong ROC (*Receiver Operating Characteristic*) cho thấy tỷ lệ sai chấp nhận (*False Acceptance Rate - FAR*) giảm xuống dưới  $10^{-7}$  với độ chính xác 94.4% trên tập huấn luyện.

- **Tập kiểm tra:**

- Gồm 3237 hình ảnh của 35 người, mỗi người có 80-100 ảnh.
- Không có ảnh nào trong tập kiểm tra xuất hiện trong tập huấn luyện.
- Tạo ra 149217 cặp trong lớp và 5088249 cặp ngoài lớp.
- Các phương pháp được so sánh:
  1. PCA trên hình ảnh NIR (khoảng cách Mahalanobis).
  2. LDA trên hình ảnh NIR (khoảng cách cosine).
  3. LBP + LDA.
  4. LBP gốc (theo phương pháp của Ahonen et al. và Hadid et al.) với ba toán tử:  $LBP_{(8,1)}^{u2}$ ,  $LBP_{(8,2)}^{u2}$ , và  $LBP_{(16,2)}^{u2}$ , sử dụng khoảng cách  $\chi^2$  giữa các biểu đồ LBP.
  5. LBP + AdaBoost.

- **Kết quả:**

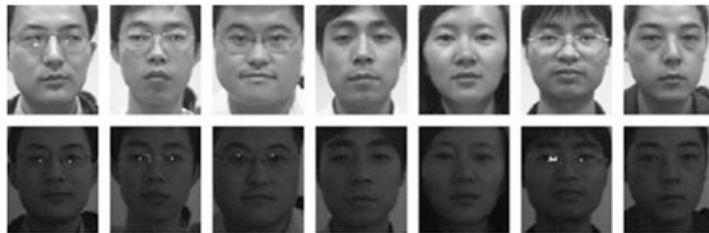
- Đường cong ROC trên tập kiểm tra (Hình 5) cho thấy hiệu suất của các phương pháp, được xếp hạng theo tỷ lệ xác minh (*Verification Rate - VR*) tại  $FAR = 0.1\%$ :
  - \* LBP + AdaBoost: VR = 91.8% (cao nhất).
  - \* LBP + LDA: VR = 69.9%.
  - \*  $LBP_{(8,2)}^{u2}$ : VR = 65.29%.
  - \*  $LBP_{(16,2)}^{u2}$ : VR = 65.29%.
  - \* Image + LDA: VR = 62.4%.
  - \*  $LBP_{(8,1)}^{u2}$ : VR = 60.7%.
  - \* Image + PCA: VR = 32.0% (thấp nhất).

**Ý nghĩa:** Phương pháp LBP + AdaBoost vượt trội so với các phương pháp khác, đặc biệt trong việc giảm thiểu lỗi sai chấp nhận và đạt độ chính xác cao.

### 2.5.2 Weak Illumination

Phần này so sánh hiệu suất của hệ thống nhận diện khuôn mặt dựa trên ánh sáng khả kiến (*Visible Light - VIS*) và NIR trong điều kiện ánh sáng yếu.

Visible Light Face Images under Controlled and Weak Light Condition

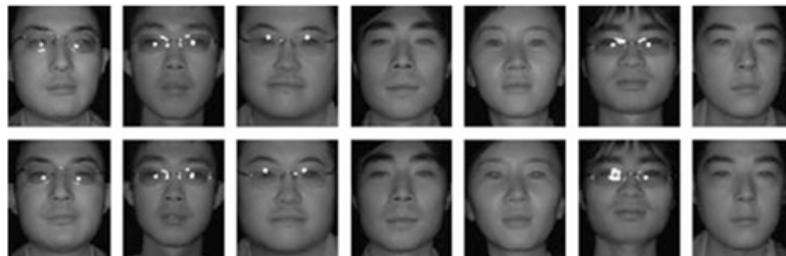


LBP+AdaBoost Matching Scores for intra- and extra-class pairs

	Ctrl 1	Ctrl 2	Ctrl 3	Ctrl 4	Ctrl 5	Ctrl 6	Ctrl 7
Weak 1	<b>0.4888</b>	0.4831	0.4751	0.4689	0.4764	0.4745	0.4658
Weak 2	0.4831	<b>0.5131</b>	0.4578	0.5202	0.4926	0.5014	0.4709
Weak 3	0.4808	0.4588	<b>0.4804</b>	0.4724	0.4814	0.4653	0.4619
Weak 4	0.4531	0.4563	0.4415	<b>0.4688</b>	0.4570	0.4582	0.4570
Weak 5	0.4892	0.4904	0.4757	0.4893	<b>0.5275</b>	0.4780	0.4824
Weak 6	0.4648	0.4835	0.4768	0.5085	0.4864	<b>0.4935</b>	0.4691
Weak 7	0.4686	0.4683	0.4636	0.4708	0.4887	0.4687	<b>0.5068</b>

Hình 6: **Trên:** Các ảnh được chụp trong điều kiện ánh sáng được kiểm soát (hàng 1) và trong điều kiện ánh sáng yếu (hàng 2), mỗi cột tương ứng với cùng một người. **Dưới:** Điểm khớp của phương pháp LBP kết hợp với AdaBoost.

Active NIR Light Face Images under Controlled and Weak Light Conditions



LBP+AdaBoost Matching Scores for intra- and extra-class pairs

	Ctrl 1	Ctrl 2	Ctrl 3	Ctrl 4	Ctrl 5	Ctrl 6	Ctrl 7
Weak 1	<b>0.6202</b>	0.3262	0.3963	0.3354	0.3492	0.3747	0.3558
Weak 2	0.3551	<b>0.6573</b>	0.3226	0.3213	0.3637	0.4054	0.3442
Weak 3	0.3886	0.3489	<b>0.7144</b>	0.3244	0.3759	0.3249	0.3139
Weak 4	0.3902	0.3062	0.3545	<b>0.6812</b>	0.4123	0.3069	0.3144
Weak 5	0.4135	0.3289	0.4507	0.3851	<b>0.6882</b>	0.3780	0.3510
Weak 6	0.3459	0.4163	0.3520	0.3292	0.2996	<b>0.6948</b>	0.2849
Weak 7	0.3697	0.2847	0.2831	0.3374	0.3656	0.2778	<b>0.6162</b>

Hình 7: **Trên:** Các ảnh được chụp trong điều kiện ánh sáng được kiểm soát (hàng 1) và trong điều kiện ánh sáng yếu (hàng 2), mỗi cột tương ứng với cùng một người. **Dưới:** Điểm khớp của phương pháp LBP kết hợp với AdaBoost.

- **Thử nghiệm:**

- Hình ảnh được chụp trong hai điều kiện: ánh sáng kiểm soát (*controlled illumination*) và ánh sáng yếu (*weak illumination*).
- Bộ phân loại LBP + AdaBoost cho VIS được huấn luyện trên hình ảnh VIS, trong khi bộ phân loại cho NIR được sử dụng từ các thử nghiệm trước.
- Bảng điểm số LBP + AdaBoost (Hình 15.6 và 15.7) cho thấy:

- \* **VIS:**

- Điểm số trung bình và phương sai của cặp trong lớp: 0.4970 và 0.0201.
- Điểm số trung bình và phương sai của cặp ngoài lớp: 0.4747 và 0.0154.
- Kết quả cho thấy có nhiều trường hợp nhầm lẫn vì điểm số trong lớp không cao hơn đáng kể so với điểm số ngoài lớp.

- \* **NIR:**

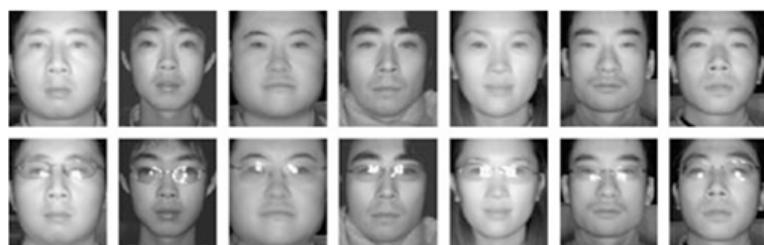
- Điểm số trung bình và phương sai của cặp trong lớp: 0.6675 và 0.0377.
- Điểm số trung bình và phương sai của cặp ngoài lớp: 0.3492 và 0.0403.
- NIR tách biệt rõ ràng giữa cặp trong lớp và ngoài lớp, với tất cả các cặp được khớp chính xác.

**Ý nghĩa:** Hệ thống NIR hoạt động tốt hơn nhiều so với VIS trong điều kiện ánh sáng yếu, nhờ khả năng bắt biến ánh sáng của hình ảnh NIR.

### 2.5.3 Eyeglasses

Phần này phân tích ảnh hưởng của việc đeo kính đến hiệu suất nhận diện khuôn mặt.

Active NIR Light Face Images With and Without Glasses



LBP+AdaBoost Matching Scores for intra- and extra-class pairs

	With 1	With 2	With 3	With 4	With 5	With 6	With 7	
W/O 1	<b>0.6537</b>	0.2880	0.3276	0.3251	0.4056	0.3510	0.3422	With 1
W/O 2	0.2205	<b>0.6468</b>	0.1730	0.2859	0.2545	0.2740	0.3856	With 2
W/O 3	0.3487	0.1838	<b>0.6565</b>	0.3209	0.3261	0.2666	0.3231	With 3
W/O 4	0.2464	0.2744	0.2654	<b>0.7092</b>	0.3331	0.2986	0.3680	With 4
W/O 5	0.3425	0.2945	0.3352	0.2896	<b>0.6120</b>	0.3191	0.3704	With 5
W/O 6	0.2896	0.2100	0.2205	0.2871	0.2699	<b>0.6366</b>	0.2346	With 6
W/O 7	0.2915	0.3708	0.3197	0.3958	0.3397	0.3056	<b>0.6541</b>	With 7
	W/O 1	W/O 2	W/O 3	W/O 4	W/O 5	W/O 6	W/O 7	

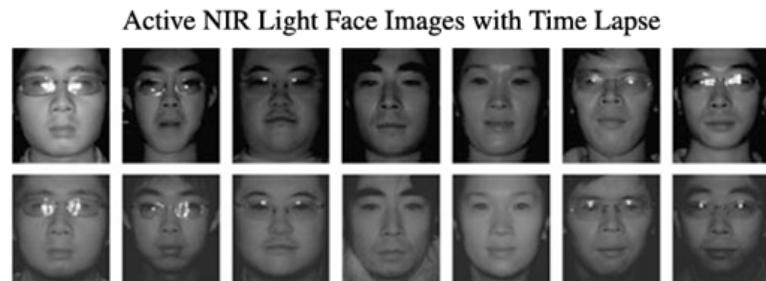
Hình 8: **Trên:** Các ảnh không đeo kính (hàng 1) và có đeo kính (hàng 2), mỗi cột tương ứng với cùng một người. **Dưới:** Điểm khớp của phương pháp LBP kết hợp với AdaBoost.

- **Thử nghiệm:**

- Hình ảnh được chụp với và không đeo kính (Hình 15.8).
- Bảng điểm số LBP + AdaBoost cho thấy:
  - \* Điểm số trung bình và phương sai của cặp trong lớp (có và không đeo kính): 0.9306 và 0.0419.
  - \* Điểm số trung bình và phương sai của cặp ngoài lớp (có hoặc không đeo kính): 0.7985 và 0.0761.
  - \* LBP + AdaBoost tách biệt tốt giữa cặp trong lớp và ngoài lớp, với điểm số trong lớp luôn cao hơn.
- Thống kê trên 1500 ảnh của 30 người (mỗi người 50 ảnh, 25 ảnh có kính, 25 ảnh không kính):
  - \* Tập *gallery*: ảnh không đeo kính.
  - \* Tập *probe*: ảnh đeo kính.
  - \* Đường cong ROC (Hình 15.5, nhãn “LBP + AdaBoost (eyeglass test)”) cho thấy tại FAR = 0.1%, VR = 87.1%, thấp hơn so với 91.8% của trường hợp không đeo kính.

**Ý nghĩa:** Việc đeo kính làm giảm nhẹ hiệu suất nhận diện, nhưng LBP + AdaBoost vẫn duy trì khả năng tách biệt tốt giữa các lớp.

#### 2.5.4 Time Lapse



**LBP+AdaBoost Matching Scores for intra- and extra-class pairs**

	Y06 1	Y06 2	Y06 3	Y06 4	Y06 5	Y06 6	Y06 7
Y05 1	<b>0.6040</b>	0.3364	0.2809	0.2946	0.3829	0.3038	0.2439
Y05 2	0.3069	<b>0.6667</b>	0.2070	0.2530	0.3142	0.2804	0.2709
Y05 3	0.3895	0.2689	<b>0.6372</b>	0.3398	0.3745	0.3022	0.2754
Y05 4	0.3010	0.2763	0.2212	<b>0.6542</b>	0.2990	0.2950	0.3265
Y05 5	0.3156	0.2789	0.2437	0.2492	<b>0.6441</b>	0.3012	0.2504
Y05 6	0.3532	0.3514	0.2958	0.2912	0.4219	<b>0.6371</b>	0.3193
Y05 7	0.3626	0.3553	0.2849	0.3530	0.3063	0.3117	<b>0.6513</b>

Hình 9: Phân tích ảnh hưởng của khoảng thời gian trôi qua. **Trên:** Ảnh khuôn mặt NIR chụp vào mùa xuân năm 2005 (hàng 1) và mùa xuân năm 2006 (hàng 2), mỗi cột tương ứng với cùng một người. **Dưới:** Điểm khớp của phương pháp LBP kết hợp với AdaBoost, trong đó Y05 và Y06 lần lượt biểu thị cho mùa xuân năm 2005 và mùa xuân năm 2006.

Phần này đánh giá tác động của khoảng cách thời gian (*time lapse*) đến nhận diện khuôn mặt.

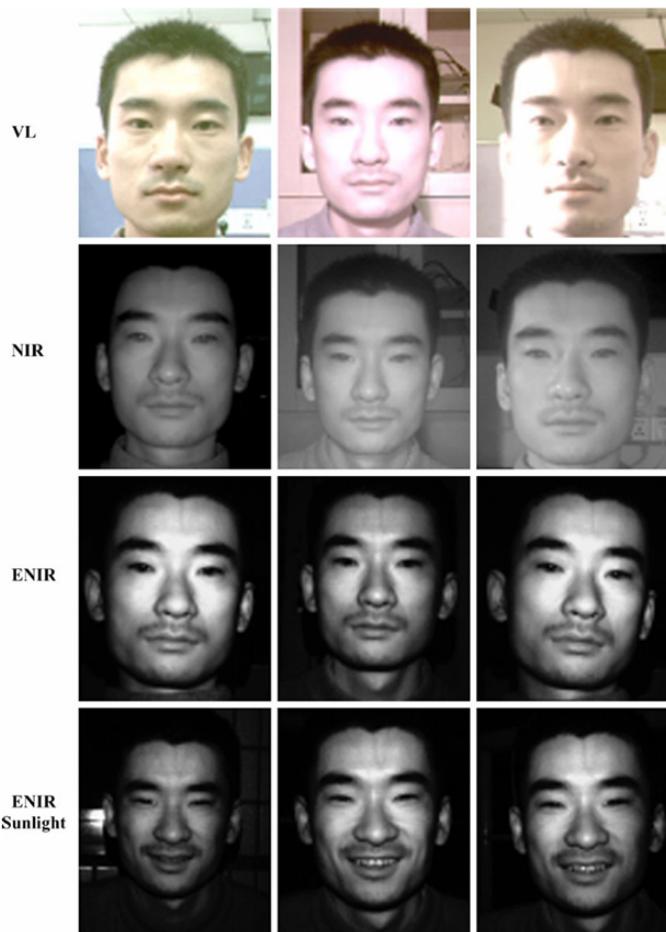
- **Thử nghiệm:**

- Hình ảnh NIR của 7 cá nhân được chụp vào mùa xuân 2005 và mùa xuân 2006 (Hình 15.9).
- Bảng điểm số LBP + AdaBoost cho thấy:
  - \* Điểm số trung bình và phương sai của cặp trong lớp: 0.6421 và 0.0198.
  - \* Điểm số trung bình và phương sai của cặp ngoài lớp: 0.3045 và 0.0462.
  - \* LBP + AdaBoost tách biệt tốt giữa các lớp, với điểm số trong lớp luôn cao hơn.
- Thống kê trên 750 ảnh của 30 người (mỗi người 25 ảnh, 10 ảnh cách đây 1 năm, 15 ảnh hiện tại):
  - \* Tập *gallery*: ảnh cách đây 1 năm.
  - \* Tập *probe*: ảnh hiện tại.
  - \* Đường cong ROC (Hình 15.5, nhãn “LBP + AdaBoost (one year lapse)”) cho thấy tại FAR = 0.1%, VR = 83.24%, thấp hơn so với 91.8% của trường hợp không có khoảng cách thời gian.

**Ý nghĩa:** Khoảng cách thời gian 1 năm làm giảm hiệu suất nhận diện, nhưng hệ thống NIR vẫn duy trì khả năng nhận diện tốt.

#### 2.5.5 Outdoor environment

Phần này đánh giá hiệu suất của hệ thống **ENIR** (Enhanced NIR) trong môi trường ngoài trời, so sánh với hệ thống VIS và NIR.



Hình 10: Ảnh khuôn mặt trong tập kiểm tra ngoài trời. **Hàng 1–3:** Ảnh VIS, NIR và ENIR dưới 3 điều kiện ánh sáng trong nhà. **Hàng 4:** Ảnh ENIR được chụp ngoài trời.

- **Tập kiểm tra:**

- Gồm 20 người, chụp trong 6 điều kiện ánh sáng:
  1. Ánh sáng trong nhà bình thường (*frontal*).
  2. Ánh sáng trong nhà mạnh (*frontal*).
  3. Ánh sáng trong nhà không trực diện (*non-frontal*).
  4. Ánh sáng mặt trời ngoài trời trực diện (*frontal*).
  5. Ánh sáng mặt trời ngoài trời bên hông (*side*).
  6. Ánh sáng mặt trời ngoài trời từ phía sau (*back*).
- Số lượng ảnh:
  - \* VIS và NIR: 800 ảnh (4 điều kiện trong nhà  $\times$  20 người  $\times$  10 ảnh/người).
  - \* ENIR: 1400 ảnh (800 ảnh trong nhà + 3 điều kiện ngoài trời  $\times$  20 người  $\times$  10 ảnh/người).
- Độ phân giải:  $640 \times 480$  cho tất cả các ảnh.
- Lưu ý: VIS và NIR không hoạt động tốt trong ánh sáng mặt trời ngoài trời do bị bão hòa.

- **Giao thức thử nghiệm:**

1. Đối với ảnh trong nhà: So sánh đường cong ROC của VIS, NIR và ENIR.
2. Đối với ảnh ngoài trời: Chỉ đánh giá ROC của ENIR (do VIS và NIR không hoạt động tốt).

- **Kết quả:**

- **Trong nhà:**

- \* Hình 15.11(a): Dưới ánh sáng mạnh trực diện (*strong frontal halogen lamp*), VIS và NIR giảm hiệu suất đáng kể, trong khi ENIR duy trì VR = 69% tại FAR = 0.001 và VR = 85% tại FAR = 0.01.
    - \* Hình 15.11(b): Tương tự, ENIR hoạt động tốt hơn trong ánh sáng không trực diện.

- **Ngoài trời** (Hình 15.11(c)):

- \* ENIR đạt VR cao nhất khi ánh sáng mặt trời từ phía sau: VR = 50% tại FAR = 0.001 và VR = 69% tại FAR = 0.01.
    - \* Hiệu suất thấp hơn khi ánh sáng mặt trời trực diện, có thể do ánh sáng mạnh gây thay đổi biểu cảm khuôn mặt (do mắt bị ảnh hưởng).

**Ý nghĩa:** Hệ thống ENIR vượt trội trong môi trường ánh sáng phức tạp, đặc biệt là ngoài trời, nơi VIS và NIR không hoạt động hiệu quả.

## 2.6 Conclusions

- **Giải pháp đề xuất:**

- Sử dụng phần cứng hình ảnh NIR chủ động, kết hợp với thuật toán hiệu quả (LBP, AdaBoost, LDA) và thiết kế hệ thống mới.
  - Đặc trưng LBP được trích xuất từ hình ảnh NIR tạo ra biểu diễn bất biến ánh sáng.
  - AdaBoost học một bộ nhận diện khuôn mặt mạnh mẽ dựa trên biểu diễn này.

- **Kết quả:**

- Hệ thống đạt độ chính xác cao và tốc độ nhanh.
  - Thí nghiệm cho thấy sự mạnh mẽ của hệ thống trong các điều kiện ánh sáng thay đổi, các nhóm dân tộc khác nhau, và vượt trội so với các phương pháp hiện có.
  - Hệ thống ENIR cải tiến hoạt động tốt trong môi trường ánh sáng mạnh như ánh sáng mặt trời.

- **Ứng dụng:**

- Hệ thống phù hợp cho các ứng dụng nhận diện 1-đối-nhiều (*1-to-many identification*) trong các tình huống người dùng hợp tác.

### 3 Phương pháp tiên tiến

#### 3.1 Giới thiệu

Trong bối cảnh các yêu cầu về an ninh và xác thực danh tính ngày càng cao, nhận diện khuôn mặt đang trở thành một công nghệ quan trọng trong nhiều lĩnh vực như giám sát và xác minh không tiếp xúc. Tuy nhiên, các hệ thống truyền thống dựa trên ảnh ánh sáng thường (visible light) gặp nhiều hạn chế trong điều kiện thiếu sáng, làm giảm độ chính xác.

Để khắc phục hạn chế của các hệ thống nhận diện khuôn mặt trong điều kiện thiếu sáng, các hệ thống sử dụng hình ảnh hồng ngoại gần (Near Infrared – NIR) đã được phát triển. Hình ảnh NIR lưu lại hình dạng khuôn mặt rõ nét mà không gây khó chịu cho người dùng, nhờ hoạt động ngoài dải ánh sáng nhìn thấy. Ngoài ra, hình ảnh NIR cũng giữ được đặc trưng khuôn mặt ổn định và có kết cấu, ít bị ảnh hưởng bởi ánh sáng nền, từ đó cải thiện độ chính xác nhận diện, đặc biệt trong các môi trường cần sự giám sát liên tục như nhà ga, bãi đỗ xe hay khu vực an ninh cao.

Vì vậy, xây dựng hệ thống nhận diện khuôn mặt sử dụng ảnh NIR là một hướng đi tiềm năng, phù hợp với các ứng dụng an ninh yêu cầu hoạt động liên tục, ngay cả trong điều kiện ánh sáng yếu.

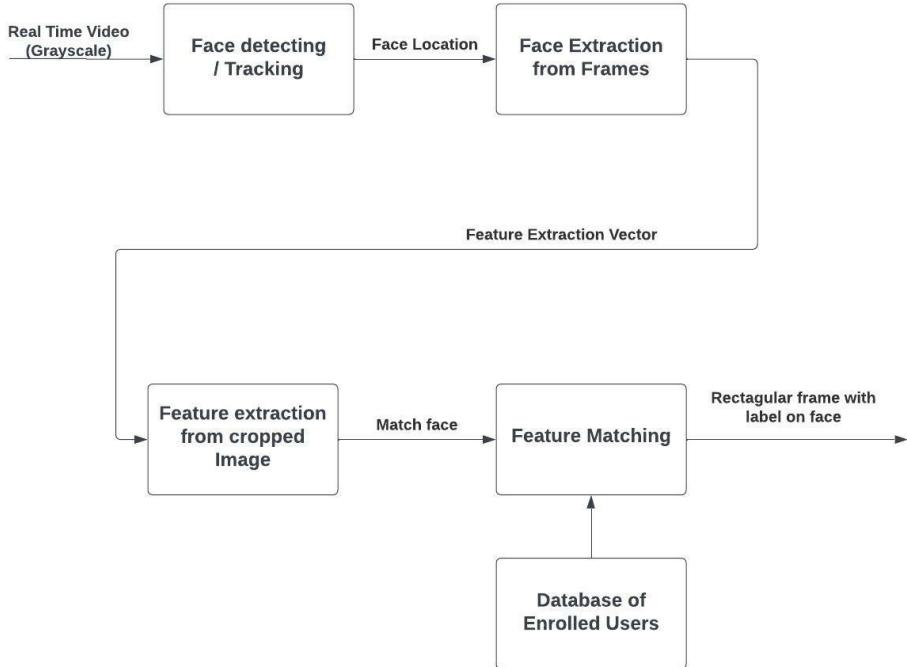
Một giải pháp về hệ thống nhận diện khuôn mặt sử dụng ảnh NIR đã được đề xuất bởi **Athreya V. Shet et al. [2]** tại hội nghị *CSITSS 2022*.

#### 3.2 Phương pháp thực hiện

**Thu thập dữ liệu:** Tập dữ liệu huấn luyện được tạo từ camera giám sát (CCTV) có tích hợp hồng ngoại (IR), với mỗi người trong tập dữ liệu có từ 40-50 bức ảnh NIR chụp xoay quanh họ dưới điều kiện ánh sáng yếu.

**Phát hiện khuôn mặt:** Việc phát hiện khuôn mặt được thực hiện bằng ba kỹ thuật khác nhau. Sau khi phát hiện, hộp giới hạn (bounding box) được vẽ quanh khuôn mặt, ảnh trong vùng giới hạn đó sẽ được cắt ra và dùng làm đầu vào cho bước nhận diện.

**Nhận diện khuôn mặt:** Trong bước nhận diện, mỗi khuôn mặt sau khi được cắt được đưa qua mô hình VGG-Face để trích xuất vector đặc trưng gồm 2.622 giá trị. Các vector này phản ánh đặc điểm nhận dạng của khuôn mặt và được sử dụng làm đầu vào cho bộ phân loại Softmax. Bộ phân loại này được huấn luyện riêng và so sánh vector của ảnh mới với các vector đã biết trong tập dữ liệu, đồng thời gán nhãn chính xác cho khuôn mặt tương ứng.



Hình 11: Quy trình thực hiện

### 3.3 Phát hiện khuôn mặt

Trong hệ thống phát hiện và nhận dạng khuôn mặt này, nhóm tác giả sử dụng đồng thời ba kỹ thuật phát hiện khuôn mặt khác nhau, bao gồm Haar-Cascade (dựa trên đặc trưng), YOLO-face, và MTCNN (cả hai đều dựa trên hình ảnh), để tối ưu hóa độ chính xác và hiệu suất phát hiện khuôn mặt trong các tình huống khác nhau.

- **Haar-Cascade:** Là một mô-đun có sẵn trong thư viện OpenCV. Quá trình phát hiện sử dụng file *haarcascade\_frontalface.xml*, tạo ra hộp giới hạn hình vuông quanh khuôn mặt sau khi đã xác định khuôn mặt bằng cách sử dụng các đặc trưng Haar, và lựa chọn đặc trưng tốt nhất thông qua thuật toán Adaboost.
- **YOLO-face:** Đây là một mô hình cải tiến dựa trên mô hình YOLOv3, được phát triển nhằm cải thiện hiệu suất trong việc phát hiện khuôn mặt. Phương pháp này sử dụng các anchor box tối ưu cho khuôn mặt cùng với hàm mất mát hồi quy chính xác hơn. Mô hình phát hiện khuôn mặt cải tiến này đã nâng cao độ chính xác đáng kể trong khi vẫn duy trì tốc độ phát hiện nhanh.
- **MTCNN (Multi-Task Cascaded Convolutional Neural Network):** Đây là một mô hình học sâu gồm ba mạng nơ-ron tích chập (P-Net, R-Net, O-Net), có khả năng phát hiện khuôn mặt với độ chính xác cao và hoạt động theo thời gian thực. Nhờ cấu trúc nhiều tầng của ba mạng nơ-ron, MTCNN cho phép nhận diện chính xác hơn trong nhiều tình huống phức tạp.

Face Detection Techniques Comparison

Dataset No	Haar-Cascade		YOLO-Face		MTCNN	
	Total frames detected	Actual frames with faces	Total frames detected	Actual frames with faces	Total frames detected	Actual frames with faces
1	259	259	282	282	280	280
2	62	62	57	57	63	63
3	226	226	187	185	240	240
4	193	174	197	196	234	234
5	302	284	297	297	300	300
6	225	220	232	232	233	233
7	270	239	280	280	282	282
8	292	288	292	292	300	300
9	88	83	111	111	135	135
10	110	90	157	157	160	160
11	264	263	270	268	273	273
12	281	277	279	279	296	296
13	249	208	254	254	278	266
14	66	46	94	94	101	99
15	187	132	289	289	300	297

Table 2: Detection algorithms result on the dataset created

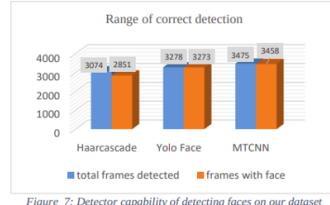


Figure 7: Detector capability of detecting faces on our dataset

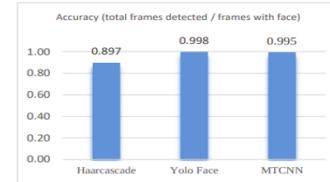


Figure 6: Accuracy of different detection techniques on our dataset

Hình 12: Kết quả so sánh giữa các kỹ thuật phát hiện khuôn mặt

Có thể thấy rằng, MTCNN phát hiện được nhiều khuôn mặt hơn, YOLO-Face có tốc độ nhanh hơn, và cả hai đều vượt trội hơn Haar-Cascade.

### 3.4 Nhận diện khuôn mặt

Mục tiêu của bước này là nhằm huấn luyện một mô hình phân loại nhằm gán nhãn chính xác cho các khuôn mặt mới, dựa trên các khuôn mặt đã biết được lưu trữ trong tập dữ liệu. Quá trình nhận diện được thực hiện sau khi khuôn mặt đã được cắt từ ảnh bởi các thuật toán phát hiện khuôn mặt.

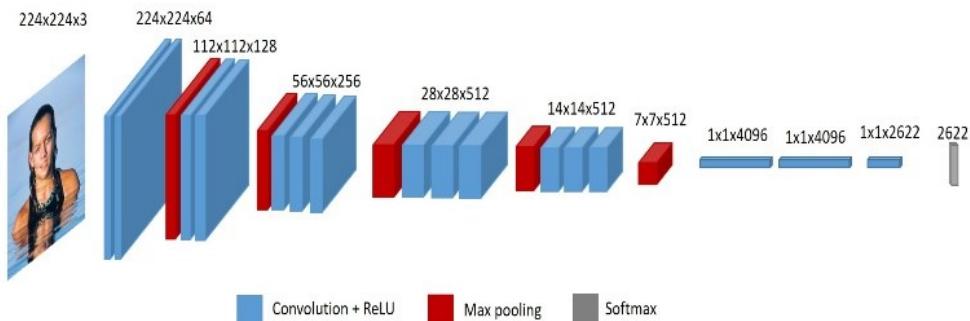
Trong giai đoạn đầu, nhóm tác giả tiến hành thử nghiệm một số phương pháp học sâu nổi tiếng cho tác vụ nhận diện khuôn mặt, điển hình là FaceNet và DeepFace. FaceNet (Google, 2015) là một kiến trúc mạnh mẽ dựa trên Inception Module, vốn là một khối xử lý đặc trưng có khả năng tối ưu hóa cấu trúc thừa cục bộ trong mạng CNN. Trong khi đó, DeepFace là một framework nhẹ, hỗ trợ nhiều mô hình nhận diện hiện đại như VGG-Face, FaceNet, OpenFace, DeepID, ArcFace, Dlib và SFace, đồng thời hỗ trợ cả các thuộc tính khuôn mặt như độ tuổi, giới tính, cảm xúc.

Tuy nhiên, khi áp dụng lên ảnh hồng ngoại gần (NIR) thu được từ hệ thống camera giám sát, các mô hình này không đạt độ chính xác như kỳ vọng do đặc tính dữ liệu khác biệt (ánh sáng yếu, không màu, nhiều nhiễu...). Do đó, nhóm tác giả quyết định xây dựng một mô hình nhận diện mới, sử dụng đặc trưng **VGG-Face** kết hợp với một **bộ phân loại Softmax** do nhóm tự thiết kế.

#### 3.4.1 Mô hình VGG-Face:

VGG-Face là một mô hình mạng nơ-ron tích chập sâu gồm 22 lớp, được huấn luyện trên tập dữ liệu gồm 2,6 triệu ảnh khuôn mặt của 2.622 người. Mô hình này được thiết kế để trích xuất đặc trưng khuôn mặt phục vụ cho các bài toán nhận diện khuôn mặt.

Khác với các kiến trúc phổ biến như **Inception** (sử dụng trong FaceNet) hay **locally connected layers**, VGG-Face sử dụng kiến trúc "rất sâu" để học các đặc trưng khuôn mặt ở nhiều cấp độ. Sau khi bỏ lớp softmax, mô hình trả về một vector đặc trưng 2.622 chiều cho mỗi ảnh, phản ánh cấu trúc và đặc điểm nhận dạng khuôn mặt. Các vector này được dùng làm đầu vào cho bộ phân loại ở các bước sau.



Hình 13: Kiến trúc của VGG-Face

#### 3.4.2 Bộ phân loại tùy chỉnh:

Dựa trên các vector đặc trưng (embeddings) thu được từ VGG-Face, nhóm xây dựng một mô hình phân loại dựa trên hồi quy Softmax. Mô hình gồm 3 lớp:

- **Lớp đầu tiên** có 100 đơn vị, sử dụng hàm kích hoạt **tanh**, giúp học đặc trưng phi tuyến mạnh.
- **Lớp thứ hai** có 10 đơn vị và cũng sử dụng **tanh**, đóng vai trò thu gọn và tổng hợp thông tin.
- **Lớp thứ ba (đầu ra)** có số lượng đơn vị phụ thuộc vào số người (labels) trong tập dữ liệu — thường từ 6 đến 15. Lớp này được nối với một hàm kích hoạt **softmax** để chuyển đổi đầu ra thành phân phối xác suất trên các lớp.

Việc sử dụng đặc trưng VGG-Face giúp tận dụng khả năng trích xuất đặc trưng mạnh mẽ của mô hình gốc, đồng thời mô hình phân loại nhẹ giúp huấn luyện nhanh, dễ kiểm soát và phù hợp với quy mô dữ liệu hiện có. Cách tiếp cận này cũng cho phép tùy biến linh hoạt khi số lượng người dùng thay đổi hoặc khi mô hình cần cập nhật thêm lớp mới.

### 3.5 Kết quả:

- Mô hình cuối cùng tích hợp cả hai bước phát hiện và nhận diện khuôn mặt trong một lần xử lý duy nhất, có thể hoạt động thực tế trên ảnh và video NIR theo thời gian thực.
- Sử dụng mô hình nhận diện tùy chỉnh dựa trên VGG-Face embeddings, các khuôn mặt trong khung hình được gắn nhãn chính xác theo tập dữ liệu đã huấn luyện.

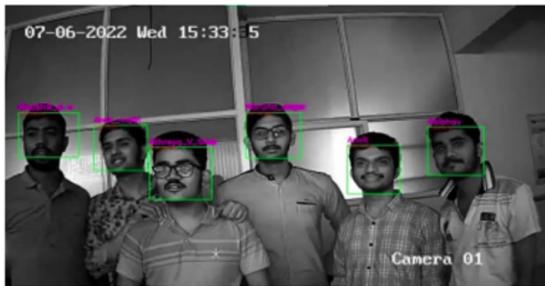


Figure 11: Real time testing of the model (sample image)

Frame rate of detection techniques along with vggface recognition. (fps, performed in 1.80Ghz cpu and Nvidia MX250)		
Detection Technique	Without recognition	With recognition
Haar-cascade	9.46	5.03
Yolo-face	1.16	0.80
Mtcnn	0.79	0.59

Table 2: Speed comparison of different approaches.

Hình 14: Kết quả sau khi thử nghiệm mô hình

## 4 Thực nghiệm:

### 4.1 Chuẩn bị dữ liệu

Bộ dữ liệu được sử dụng trong thực nghiệm này là **TD-NIR-A-Set1** được tải về từ đường dẫn: [https://tdface.ece.tufts.edu/downloads/TD\\_NIR\\_A/](https://tdface.ece.tufts.edu/downloads/TD_NIR_A/). Đây là một phần của bộ cơ sở dữ liệu TD-NIR-A (Tufts Database – Near Infrared), chứa các hình ảnh khuôn mặt người được chụp trong dải ánh sáng gần hồng ngoại (Near-Infrared - NIR). Các hình ảnh trong tập dữ liệu này được ghi nhận dưới nhiều góc nhìn khác nhau nhằm phục vụ các nghiên cứu về nhận diện khuôn mặt trong điều kiện ánh sáng đặc biệt.

- **Số lượng đối tượng:** 25 người tham gia, mỗi người có nhiều ảnh khuôn mặt chụp ở các góc độ khác nhau (chính diện, nghiêng trái, nghiêng phải, từ trên xuống, từ dưới lên, v.v.).
- **Tổng số ảnh:** 815 ảnh NIR.

#### 4.1.1 Chia tập dữ liệu

Trước khi đưa vào huấn luyện mô hình, toàn bộ tập dữ liệu gốc được chia thành ba tập riêng biệt: tập huấn luyện (training set), tập xác thực (validation set), và tập kiểm tra (test set). Việc phân chia được thực hiện theo tỷ lệ:

$$\text{Train : Validation : Test} = 7 : 2 : 1$$

Mục đích của việc chia tập như trên nhằm đảm bảo:

- Tập huấn luyện chiếm phần lớn dữ liệu (70%) để mô hình có thể học đầy đủ các đặc trưng trong ảnh.
- Tập xác thực (20%) được sử dụng trong quá trình huấn luyện để theo dõi hiệu năng mô hình trên dữ liệu chưa từng thấy.
- Tập kiểm tra (10%) được giữ lại để đánh giá khách quan mô hình sau khi quá trình huấn luyện hoàn tất.

#### 4.1.2 Tăng cường dữ liệu (Data Augmentation)

Do số lượng ảnh ban đầu tương đối hạn chế, đặc biệt là trong bối cảnh deep learning yêu cầu một lượng lớn dữ liệu để đạt hiệu quả cao, kỹ thuật tăng cường dữ liệu được áp dụng trên **tập huấn luyện** nhằm:

- Tăng tính đa dạng cho dữ liệu đầu vào.
- Mô phỏng các điều kiện biến thiên trong thực tế (góc chụp, ánh sáng, nhiễu...).
- Giảm thiểu hiện tượng overfitting trong quá trình huấn luyện.

Các kỹ thuật tăng cường dữ liệu phổ biến bao gồm:

- Xoay ảnh (rotation).
- Lật ngang (horizontal flipping).
- Dịch chuyển ảnh (translation).
- Thay đổi độ sáng hoặc tương phản.
- Thêm nhiễu Gaussian.

**Lưu ý:** Việc tăng cường dữ liệu chỉ được thực hiện trên tập huấn luyện. Các tập xác thực và kiểm tra được giữ nguyên để đảm bảo tính khách quan trong quá trình đánh giá mô hình.

#### 4.1.3 Thông kê sau khi tăng cường dữ liệu

Sau khi thực hiện tăng cường dữ liệu, kích thước cuối cùng của từng tập dữ liệu như sau:

Tập dữ liệu	Số lượng ảnh
Tập huấn luyện (Train)	2,794 ảnh
Tập xác thực (Validation)	165 ảnh
Tập kiểm tra (Test)	91 ảnh

Bảng 2: Số lượng ảnh sau khi tăng cường dữ liệu

## 4.2 Face Detection

### 4.2.1 Importing phương pháp face\_detection

Trong bước đầu tiên của quy trình xử lý ảnh, nhóm tiến hành tích hợp các phương pháp phát hiện khuôn mặt (face detection) từ thư mục `face_detection`. Trong số đó, phương pháp được sử dụng chính là **YOLOFace** — một biến thể của kiến trúc YOLO (You Only Look Once), được huấn luyện đặc biệt cho nhiệm vụ phát hiện khuôn mặt người với hiệu suất cao cả về độ chính xác lẫn tốc độ.

YOLOFace được huấn luyện trên tập dữ liệu khuôn mặt lớn như *WIDER FACE*, có khả năng nhận diện khuôn mặt ở nhiều điều kiện khác nhau như góc nhìn, kích thước khuôn

mặt và ánh sáng phức tạp. Phương pháp này phù hợp để xử lý hình ảnh trong thời gian thực, ngay cả khi có nhiều khuôn mặt trong một khung hình.

Trong hệ thống của nhóm, thư viện `yoloface` được tích hợp để đảm nhận các chức năng cốt lõi sau trong quy trình phát hiện và xử lý khuôn mặt tự động:

- **Tự động xác định vị trí khuôn mặt:** Mô hình học sâu sẽ tự động xác định vị trí khuôn mặt trong ảnh đầu vào mà không cần gán nhãn thủ công. YOLOFace cho phép phát hiện đa khuôn mặt trong một ảnh chỉ với một lần chạy mô hình duy nhất.
- **Vẽ khung (bounding box)** bao quanh từng khuôn mặt được phát hiện: Dựa trên tọa độ nhận diện, thư viện sẽ vẽ các hình chữ nhật bao quanh khuôn mặt, giúp trực quan hóa kết quả và thường hiển thị kèm theo điểm tin cậy.
- **Cắt ảnh (crop)** khuôn mặt ra khỏi ảnh gốc: Chức năng này trích xuất vùng ảnh chỉ chứa khuôn mặt, loại bỏ phần nền không liên quan và thường chuẩn hóa kích thước khuôn mặt để phục vụ các bước xử lý sau.
- **Lưu trữ ảnh khuôn mặt đã cắt** để phục vụ cho các bước xử lý tiếp theo: Các khuôn mặt đã cắt được lưu trữ dưới định dạng chuẩn (JPEG) để phục vụ cho các tác vụ tiếp theo là nhận diện khuôn mặt (face recognition).

Việc sử dụng YOLOFace đặc biệt hiệu quả trong bối cảnh xử lý ảnh hồng ngoại gần (NIR), vốn có đặc điểm độ tương phản thấp, nhiễu nền cao và góc nhìn không đồng nhất. Nhờ được huấn luyện trên tập dữ liệu lớn như *WIDER FACE*, YOLOFace có khả năng phát hiện khuôn mặt trong nhiều điều kiện ánh sáng, góc quay và kích thước khác nhau, giúp đảm bảo độ ổn định và chính xác cho quy trình xử lý ảnh đầu vào.

```
import sys
import os
sys.path.append("/content/Face-recognition-in-Near-InfraRed-images/")

print("Importing face detection libraries....")
# Sử dụng YOLOFace để phát hiện khuôn mặt
from face_detection.yoloface_face_detection import yoloface_detection
print("Successfully import detection libraries")
```

Nhóm cũng quyết định lựa chọn YOLOFace vì các lý do sau:

- **Hiệu năng tốt:** YOLOFace có khả năng phát hiện khuôn mặt chính xác cao, kể cả với các ảnh có góc quay nghiêng, độ phân giải thấp hoặc ảnh NIR.
- **Tốc độ nhanh:** Do bản chất của YOLO là mô hình một bước (single-shot), tốc độ xử lý ảnh rất nhanh so với các phương pháp nhiều bước như MTCNN.
- **Dễ tích hợp:** Thư viện được đóng gói tốt, dễ dàng sử dụng trong pipeline hiện tại mà không cần tinh chỉnh nhiều tham số.

Việc tích hợp thành công module `yoloface_detection` giúp tự động hóa hoàn toàn quá trình phát hiện và trích xuất khuôn mặt từ ảnh gốc, từ đó tạo tiền đề quan trọng cho các bước tiếp theo trong pipeline như huấn luyện mô hình nhận diện khuôn mặt trên ảnh NIR.

#### 4.2.2 Xây dựng kiến trúc mô hình VGG-Face

Trong phần này, nhóm lựa chọn sử dụng **VGG-Face** — một mạng nơ-ron tích chập sâu (Deep Convolutional Neural Network - DCNN), được phát triển chuyên biệt cho **nhiệm vụ nhận diện khuôn mặt người**.

Mô hình VGG-Face là một biến thể được điều chỉnh từ kiến trúc nổi tiếng **VGG-16**, được thiết kế và huấn luyện bởi Visual Geometry Group (VGG) thuộc Đại học Oxford. Khác với VGG-16 gốc dùng cho phân loại ảnh tổng quát (ImageNet), VGG-Face được tinh chỉnh lại để tối ưu cho tác vụ nhận diện khuôn mặt, với đầu ra phù hợp cho bài toán nhận dạng.

**Cấu trúc tổng thể:** Kiến trúc mô hình VGG-Face bao gồm các thành phần chính sau:

- **Lớp tích chập (Convolutional Layers):** Dùng để trích xuất đặc trưng cục bộ từ ảnh đầu vào. Mỗi khối tích chập thường bao gồm 2 hoặc 3 lớp Conv2D, mỗi lớp sử dụng hàm kích hoạt ReLU nhằm tăng tính phi tuyến và khả năng biểu diễn đặc trưng phức tạp.
- **Lớp đệm (ZeroPadding2D):** Được sử dụng trước mỗi lớp Conv2D nhằm giữ nguyên kích thước không gian (chiều cao và chiều rộng) của ảnh sau khi tích chập, bằng cách thêm các giá trị 0 vào các biên của tensor đầu vào.
- **Lớp gộp cực đại (MaxPooling2D):** Có tác dụng giảm dần kích thước không gian của các bản đồ đặc trưng (feature maps), từ đó giảm số lượng tham số, tăng hiệu quả tính toán và giảm nguy cơ overfitting. Lớp này giữ lại các đặc trưng quan trọng nhất trong vùng gộp.
- **Lớp tích chập mô phỏng fully-connected (Conv2D với kernel lớn):** Thay vì sử dụng các lớp Dense, mô hình sử dụng các lớp Conv2D với kích thước kernel lớn ( $7 \times 7$  và  $1 \times 1$ ) để mô phỏng các lớp fully-connected. Cách làm này giúp giảm số lượng tham số và tận dụng tính chất không gian của ảnh.
- **Lớp Dropout:** Được chèn sau các lớp tích chập cuối nhằm giảm hiện tượng overfitting. Cơ chế hoạt động là ngẫu nhiên vô hiệu hóa một tỷ lệ nơ-ron trong quá trình huấn luyện, từ đó tăng tính khái quát của mô hình.
- **Lớp Flatten và Softmax:**
  - **Flatten:** Biến đổi tensor đầu ra 3 chiều từ lớp trước thành vector 1 chiều, chuẩn bị cho bước phân loại cuối cùng.
  - **Activation('softmax')**: Biến đổi vector đầu ra thành xác suất phân phối trên 2,622 lớp, tương ứng với số lượng danh tính trong tập dữ liệu huấn luyện gốc của VGG-Face.

## Định nghĩa mô hình bằng Keras:

- Mô hình VGG-Face được nhóm cài đặt bằng cách sử dụng cấu trúc Sequential trong Keras, với tổng cộng **16 lớp có trọng số** bao gồm **13 lớp tích chập (Conv2D)** và **3 lớp fully-connected** được mô phỏng thông qua các lớp Conv2D có kích thước kernel lớn. Các lớp này được tổ chức thành 5 khối chính, kết hợp với ZeroPadding2D để giữ nguyên kích thước không gian và MaxPooling2D nhằm giảm chiều dữ liệu sau mỗi khối.
- Phần cuối của mô hình sử dụng các lớp Conv2D(4096, 7×7), Conv2D(4096, 1×1) và Conv2D(2622, 1×1) để mô phỏng quá trình phân loại. Hai lớp Dropout được chèn giữa nhằm tăng khả năng khai quát và giảm overfitting trong quá trình huấn luyện. Lớp Flatten và Activation('softmax') đảm nhiệm vai trò chuyển đổi kết quả thành phân phối xác suất trên **2622 lớp đầu ra**, mỗi lớp tương ứng với một danh tính trong tập dữ liệu huấn luyện gốc.

Đoạn mã cụ thể như sau:

```
# Dinh nghia kien truc mo hinh VGG-Face
model = Sequential()

# Khoi 1
model.add(ZeroPadding2D((1,1), input_shape=(224,224,3)))
model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

# Khoi 2
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(128, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

# Khoi 3
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

# Khoi 4
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
```

```

model.add(MaxPooling2D((2,2), strides=(2,2)))

# Khoi 5
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

# Lop Fully-Connected dang Conv va lop Dropout
model.add(Convolution2D(4096, (7, 7), activation='relu'))
model.add(Dropout(0.5))
model.add(Convolution2D(4096, (1, 1), activation='relu'))
model.add(Dropout(0.5))
model.add(Convolution2D(2622, (1, 1)))

# Lop Flatten va Softmax
model.add(Flatten())
model.add(Activation('softmax'))

```

Chức năng của từng khối trong mô hình VGG-Face được cài đặt:

- **Khối 1 (Conv Block 1):**

- Gồm 2 lớp Conv2D(64) và 1 lớp MaxPooling2D.
- Trích xuất các đặc trưng cơ bản như cạnh, góc, đường biên từ ảnh đầu vào.
- Kích thước đầu vào  $224 \times 224$  được giảm xuống còn  $112 \times 112$ .

- **Khối 2 (Conv Block 2):**

- Gồm 2 lớp Conv2D(128) và 1 lớp MaxPooling2D.
- Học các đặc trưng phức tạp hơn như mô hình vùng, texture khuôn mặt.
- Kích thước giảm còn  $56 \times 56$ .

- **Khối 3 (Conv Block 3):**

- Gồm 3 lớp Conv2D(256) và 1 lớp MaxPooling2D.
- Bắt đầu học các tổ hợp đặc trưng như cấu trúc mắt, mũi, miệng.
- Kích thước tiếp tục giảm còn  $28 \times 28$ .

- **Khối 4 (Conv Block 4):**

- Gồm 3 lớp Conv2D(512) và 1 lớp MaxPooling2D.
- Tăng cường khả năng nhận biết các đặc trưng trừu tượng hơn (bộ lọc khuôn mặt).
- Kích thước giảm còn  $14 \times 14$ .

- **Khối 5 (Conv Block 5):**

- Gồm 3 lớp Conv2D(512) và 1 lớp MaxPooling2D.
- Hoàn thiện trích xuất đặc trưng toàn cục của khuôn mặt.
- Kết quả đầu ra là tensor kích thước  $7 \times 7 \times 512$ .

- **Khối Fully-Connected dạng Conv (ConvFC):**

- Gồm các lớp Conv2D(4096, 7x7), Dropout, Conv2D(4096, 1x1), Dropout, Conv2D(2622, 1x1).
- Mô phỏng lớp fully connected bằng tích chập, học đặc trưng tổng hợp cuối cùng cho từng danh tính khuôn mặt.
- Dropout giúp giảm overfitting trong quá trình huấn luyện.

- **Lớp Flatten và Softmax:**

- **Flatten:** Chuyển đầu ra từ tensor 3 chiều thành vector 1 chiều.
- **Activation('softmax'):** Chuẩn hóa thành phân phối xác suất trên 2622 lớp (danh tính).

Mô hình có đầu vào chuẩn là ảnh có kích thước  $224 \times 224 \times 3$ , và có thể được mở rộng để phục vụ nhiều mục đích khác nhau. Trong các ứng dụng thực tế, lớp softmax có thể được loại bỏ để trích xuất **vector đặc trưng (face embedding)** từ ảnh đầu vào, làm đầu vào cho các hệ thống nhận diện hoặc xác thực khuôn mặt khác.

#### 4.2.3 Nạp trọng số mô hình VGG-Face và loại bỏ lớp Softmax

Trong phần này, nhóm tiến hành nạp mô hình VGG-Face đã được huấn luyện sẵn (<https://www.kaggle.com/datasets/vincentscheltjens/vgg-face-weights-h5>) với tập dữ liệu lớn chứa hơn 2.6 nghìn danh tính (tương ứng 2,622 lớp đầu ra). Mục tiêu không phải là phân loại khuôn mặt, mà là sử dụng mô hình như một **bộ trích xuất đặc trưng khuôn mặt (face embedding extractor)**. Vì vậy, nhóm loại bỏ lớp kích hoạt **softmax** ở cuối mô hình, giữ lại đầu ra của lớp **Flatten** để thu được một vector đặc trưng có kích thước 2,622 chiều cho mỗi ảnh đầu vào. Việc này giúp chuyển ảnh khuôn mặt đầu vào thành một vector đặc trưng, cho phép thực hiện:

- So sánh mức độ tương đồng giữa các khuôn mặt.
- Phân cụm các khuôn mặt gần nhau trong không gian đặc trưng.
- Xây dựng hệ thống nhận diện hoặc xác thực khuôn mặt dựa trên khoảng cách vector.

Để sử dụng mô hình cho tác vụ trích đặc trưng khuôn mặt, lớp **Activation('softmax')** được loại bỏ. Đầu ra được lấy từ lớp **Flatten** ngay trước softmax, trả về một vector 1 chiều có độ dài 2622.

```

vgg_face = Model(inputs=model.layers[0].input,
                  outputs=model.layers[-2].output)
model.load_weights('/content/vgg_face_weights.h5')

```

**Ý nghĩa:** Mỗi ảnh khuôn mặt đầu vào sau khi qua mô hình sẽ được ánh xạ thành một vector đặc trưng  $\in R^{2622}$ . Vector này phản ánh đặc trưng hình dạng, cấu trúc của khuôn mặt và có thể dùng cho các tác vụ so sánh hoặc phân loại khác.

**Thông số đáng chú ý:**

- **Tổng số tham số học được:** 145,002,878 (khoảng **553.14 MB**).
- **Tầng có nhiều tham số nhất:**
  - conv2d\_13 (4096 filters,  $7 \times 7$ ): 102,764,544 tham số.
  - conv2d\_14 (4096,  $1 \times 1$ ): 16,781,312 tham số.
  - conv2d\_15 (2622,  $1 \times 1$ ): 10,742,334 tham số.
- **Toàn bộ các tham số đều có thể huấn luyện (trainable).**

**Ứng dụng thực tế:** Sau khi đã xây dựng mô hình trích đặc trưng, pipeline nhận diện khuôn mặt có thể thực hiện như sau:

1. Chuẩn hóa ảnh đầu vào về kích thước  $224 \times 224$ , chuyển sang dạng tensor và chuẩn hóa giá trị pixel.
2. Dự đoán vector đặc trưng:

embedding = vgg\_face.predict(preprocessed image)

3. Tính toán khoảng cách giữa các vector embedding bằng các công thức như:

- **Cosine Similarity:** đo góc giữa hai vector.
- **Euclidean Distance:** đo độ gần trong không gian đặc trưng.

4. Ứng dụng: Nhận diện, xác thực, hoặc phân cụm khuôn mặt.

#### 4.2.4 Thực hiện crop images bằng YOLOFace

Nhóm thực hiện gọi tập tin main.py trong thư mục face\_detection để thực hiện phát hiện và xử lý khuôn mặt trên thư mục ảnh NIR.

```

subprocess.run(['python',
               'Face-recognition-in-Near-InfraRed-images'
               '/face_detection/main.py'])

```

## 4.3 Face Recognition

### 4.3.1 Chia tập huấn luyện, kiểm tra và kiểm thử

Sau khi đã thực hiện cắt ảnh khuôn mặt từ ảnh gốc bằng phương pháp YOLOFace, nhóm tiến hành chia dữ liệu thành 3 tập riêng biệt:

- **Tập huấn luyện (train):** Dùng để huấn luyện mô hình nhận diện khuôn mặt.
- **Tập kiểm thử (validation):** Dùng để hiệu chỉnh tham số mô hình trong quá trình huấn luyện, tránh overfitting.
- **Tập kiểm tra (test):** Dùng để đánh giá độ chính xác cuối cùng của mô hình sau khi huấn luyện.

Việc tách biệt rõ ràng ba tập dữ liệu này giúp đảm bảo quy trình huấn luyện, tinh chỉnh và đánh giá mô hình được thực hiện đúng chuẩn, không xảy ra hiện tượng *data leakage* (rò rỉ dữ liệu).

Nhóm sử dụng đoạn mã dưới đây để sao chép dữ liệu huấn luyện (train), dữ liệu kiểm thử (validation) và dữ liệu kiểm tra (test) từ thư mục dữ liệu sau khi thực hiện face\_detection sang các thư mục đích tương ứng để phục vụ cho các bước sau:

```
src_base = "/content/yolo_cropped_images"

dest_map = {
    "train": "/content/images_train_crop",
    "test": "/content/images_test_crop",
    "val": "/content/images_val_crop"
}

for split in ["train", "test", "val"]:
    src_split_path = os.path.join(src_base, split)
    dst_split_path = dest_map[split]

    if not os.path.isdir(src_split_path):
        continue

    for person_id in os.listdir(src_split_path):
        src_person_path = os.path.join(src_split_path, person_id)
        dst_person_path = os.path.join(dst_split_path, person_id)

        if not os.path.isdir(src_person_path):
            continue

        os.makedirs(dst_person_path, exist_ok=True)

        for filename in os.listdir(src_person_path):
            src_file = os.path.join(src_person_path, filename)
            dst_file = os.path.join(dst_person_path, filename)
            shutil.copy2(src_file, dst_file)
```

Việc chia dữ liệu đúng cách và đảm bảo tách biệt giữa train/val/test giúp đảm bảo độ tin cậy trong quá trình đánh giá mô hình. Ngoài ra, việc chuẩn hóa cấu trúc thư mục cũng giúp dễ dàng hơn trong các bước huấn luyện sau này.

#### 4.3.2 Chuẩn bị dữ liệu huấn luyện (train)

Mục tiêu của bước này là chuẩn bị tập dữ liệu huấn luyện gồm các **vector đặc trưng (embedding)** trích xuất từ ảnh khuôn mặt, cùng với **nhãn tương ứng**, để sử dụng cho các mô hình nhận diện khuôn mặt ở các bước sau.

Nhóm thực hiện lần lượt các thao tác sau:

1. Duyệt qua từng thư mục chứa ảnh đã cắt ở thư mục /content/images\_train\_crop, trong đó mỗi thư mục đại diện cho một người cụ thể:
  - Resize ảnh về kích thước  $224 \times 224$  pixel (chuẩn của VGG-Face).
  - Chuyển ảnh thành mảng numpy array và mở rộng chiều thành dạng batch ( $1, 224, 224, 3$ ).
  - Thực hiện tiền xử lý ảnh bằng hàm preprocess\_input của VGG-Face.
  - Dưa ảnh vào mô hình vgg\_face đã loại bỏ lớp softmax để trích xuất vector đặc trưng khuôn mặt.
  - Lưu vector đặc trưng vào danh sách `x_train`, và lưu chỉ số nhãn (ứng với người tương ứng) vào `y_train`.
2. Với mỗi ảnh khuôn mặt:
  - Resize ảnh về kích thước  $224 \times 224$  pixel (chuẩn của VGG-Face).
  - Chuyển ảnh thành mảng numpy array và mở rộng chiều thành dạng batch ( $1, 224, 224, 3$ ).
  - Thực hiện tiền xử lý ảnh bằng hàm preprocess\_input của VGG-Face.
  - Dưa ảnh vào mô hình vgg\_face đã loại bỏ lớp softmax để trích xuất vector đặc trưng khuôn mặt.
  - Lưu vector đặc trưng vào danh sách `x_train`, và lưu chỉ số nhãn (ứng với người tương ứng) vào `y_train`.
3. Đồng thời, nhóm tạo một từ điển ánh xạ `person_rep` giữa chỉ số nhãn và tên thư mục của từng người.

Đoạn mã triển khai như sau:

```
x_train = []
y_train = []
person_rep = dict()

person_folders = os.listdir('/content/images_train_crop')

for i, person in enumerate(person_folders):
    person_rep[i] = person # Luu anh xa chi so -> ten nguoi
    image_names = os.listdir(f'/content/images_train_crop/{person}/')
    print(person_rep[i])

    for image_name in image_names:
        print(image_name, end="\t")

    # Load anh va resize ve 224x224
    img = load_img(f'/content/images_train_crop/{person}/{image_name}',
                   target_size=(224, 224))
    img = img_to_array(img)
    img = np.expand_dims(img, axis=0)
```

```



```

Sau khi thực thi đoạn mã trên, nhóm thu được:

- **x\_train**: Danh sách các vector đặc trưng (features) thu được từ ảnh khuôn mặt sau khi qua mô hình VGG-Face. Mỗi vector có kích thước 2622 (do lớp cuối Flatten trước softmax).
- **y\_train**: Danh sách nhãn số, mỗi giá trị tương ứng với một người trong tập huấn luyện.
- **person\_rep**: Từ điển ánh xạ giữa chỉ số nhãn (int) và tên người (tên thư mục chứa ảnh khuôn mặt).

Các vector đặc trưng từ VGG-Face có khả năng biểu diễn thông tin khuôn mặt ở mức cao (high-level features), giúp tăng hiệu quả trong các bài toán nhận diện, phân loại và so sánh khuôn mặt ở các bước sau. Bộ dữ liệu huấn luyện này có thể được dùng cho các mô hình như SVM, KNN, hoặc huấn luyện classifier đơn giản.

#### 4.3.3 Chuẩn bị dữ liệu kiểm thử (validation) và dữ liệu kiểm tra (test)

Tương tự như bước trên, bước này cũng nhằm chuẩn bị tập dữ liệu kiểm thử (validation) và dữ liệu kiểm tra (test) gồm các **vector đặc trưng** (embeddings) được trích xuất từ ảnh khuôn mặt, cùng với **nhãn định danh tương ứng**. Tập dữ liệu này sẽ được sử dụng để đánh giá độ chính xác và khả năng tổng quát của mô hình nhận diện khuôn mặt ở các bước sau.

Quy trình xử lý được thực hiện hoàn toàn tương tự như đối với tập huấn luyện (**train**), bao gồm các bước sau:

1. Duyệt qua thư mục chứa ảnh khuôn mặt đã cắt cho tập **test** và **val**
2. Với mỗi ảnh trong các thư mục:
  - Resize ảnh về kích thước  $224 \times 224$ .
  - Chuyển ảnh sang định dạng **numpy array**, mở rộng chiều để phù hợp với mô hình.
  - Tiền xử lý bằng hàm **preprocess\_input**.
  - Đưa vào mô hình **vgg\_face** để trích xuất đặc trưng khuôn mặt.
  - Ghi nhận vector đặc trưng và nhãn tương ứng.

Sau khi thực thi, nhóm thu được:

- **x\_val**, **x\_test**: Danh sách các vector đặc trưng thu được từ ảnh khuôn mặt trong tập kiểm thử và kiểm tra. Mỗi vector có chiều dài 2622.

- **y\_val**, **y\_test**: Danh sách nhãn số, ảnh xạ đến từng người trong tập tương ứng.
- **person\_rep**: Từ điển ánh xạ giữa chỉ số nhãn (số nguyên) và tên người (tên thư mục chứa ảnh khuôn mặt). Từ điển này thường dùng chung với tập huấn luyện để đảm bảo nhất quán.

#### 4.3.4 Lưu trữ và tải lại dữ liệu đã xử lý

Sau khi hoàn tất quá trình trích xuất đặc trưng từ ảnh khuôn mặt, nhóm tiến hành lưu các mảng dữ liệu đặc trưng và nhãn tương ứng vào đĩa dưới định dạng .npy của NumPy. Việc này nhằm:

- Tránh phải thực hiện lại quá trình trích xuất đặc trưng tốn thời gian.
- Tiện lợi khi sử dụng cho các bước huấn luyện hoặc đánh giá mô hình sau này.
- Đảm bảo tái sử dụng dữ liệu nhất quán giữa các lần thực thi.

Đầu tiên, các danh sách đặc trưng và nhãn được chuyển sang mảng NumPy để dễ dàng lưu trữ:

```
x_train = np.array(x_train)
y_train = np.array(y_train)
x_val = np.array(x_val)
y_val = np.array(y_val)
x_test = np.array(x_test)
y_test = np.array(y_test)
```

Sau đó, nhóm sử dụng hàm `np.save()` để lưu dữ liệu thành các tệp:

```
np.save('train_data_yolo_25_faces', x_train)
np.save('train_labels_yolo_25_faces', y_train)
np.save('val_data_yolo_25_faces', x_val)
np.save('val_labels_yolo_25_faces', y_val)
np.save('test_data_yolo_25_faces', x_test)
np.save('test_labels_yolo_25_faces', y_test)
```

Các tệp dữ liệu đầu ra sẽ gồm:

- `train_data_yolo_25_faces.npy`, `train_labels_yolo_25_faces.npy`
- `val_data_yolo_25_faces.npy`, `val_labels_yolo_25_faces.npy`
- `test_data_yolo_25_faces.npy`, `test_labels_yolo_25_faces.npy`

Khi cần sử dụng trong các bước tiếp theo (ví dụ huấn luyện classifier), nhóm chỉ cần tải lại dữ liệu đã lưu bằng `np.load()`:

```
x_train = np.load('train_data_yolo_25_faces.npy')
y_train = np.load('train_labels_yolo_25_faces.npy')
x_val = np.load('val_data_yolo_25_faces.npy')
y_val = np.load('val_labels_yolo_25_faces.npy')
```

```

x_test = np.load('test_data_yolo_25_faces.npy')
y_test = np.load('test_labels_yolo_25_faces.npy')

```

### 4.3.5 Xây dựng mô hình phân loại Softmax dựa trên đặc trưng khuôn mặt đã trích xuất từ mô hình VGG-Face.

Khi trích xuất đặc trưng từ ảnh khuôn mặt thông qua mô hình VGG-Face xong, nhóm xây dựng một mô hình phân loại nhiều lớp (**multi-class classification**) để nhận diện người trong ảnh. Mô hình được huấn luyện trên các vector đặc trưng có kích thước 2622 (từ tầng Flatten của VGG-Face), với 25 lớp đầu ra tương ứng với 25 người.

**Cấu trúc mô hình:** Mô hình được xây dựng bằng Keras Sequential, gồm ba tầng fully-connected:

- **Lớp ẩn thứ nhất:**

- Dense(100): Tầng fully-connected với 100 đơn vị đầu ra, kích thước đầu vào là `x_train.shape[1] = 2622`.
- BatchNormalization(): Chuẩn hóa đầu ra để tăng tốc độ huấn luyện và tính ổn định.
- Activation('tanh'): Hàm kích hoạt phi tuyến, cho đầu ra trong khoảng  $(-1, 1)$ .
- Dropout(0.3): Loại ngẫu nhiên 30% số node trong quá trình huấn luyện nhằm tránh overfitting.

- **Lớp ẩn thứ hai:**

- Dense(6): Tầng fully-connected với 6 neuron. (Lưu ý: tầng này có thể được xem như tầng trung gian giảm chiều).
- BatchNormalization(), Activation('tanh') và Dropout(0.2): giống lớp trước, nhưng với tỉ lệ dropout thấp hơn.

- **Lớp đầu ra (Output layer):**

- Dense(25): Số neuron bằng với số lớp cần phân loại (tức 25 người).
- Activation('softmax'): Hàm kích hoạt softmax để chuyển đầu ra thành phân phối xác suất trên 25 lớp.

Mô hình được biên dịch với các tham số sau:

- **Loss function:** SparseCategoricalCrossentropy – phù hợp cho bài toán phân loại nhiều lớp với nhãn là số nguyên.
- **Optimizer:** Nadam – kết hợp giữa Adam và Nesterov momentum, thường cho hiệu quả huấn luyện tốt.
- **Metrics:** accuracy – dùng để đánh giá độ chính xác trong quá trình huấn luyện.

```

classifier_model = Sequential()
classifier_model.add(Dense(units=100, input_dim=x_train.shape[1],
                           kernel_initializer='glorot_uniform'))
classifier_model.add(BatchNormalization())
classifier_model.add(Activation('tanh'))
classifier_model.add(Dropout(0.3))

classifier_model.add(Dense(units=6, kernel_initializer='glorot_uniform'))
classifier_model.add(BatchNormalization())
classifier_model.add(Activation('tanh'))
classifier_model.add(Dropout(0.2))

classifier_model.add(Dense(units=25, kernel_initializer='he_uniform'))
classifier_model.add(Activation('softmax'))

classifier_model.compile(
    loss=tf.keras.losses.SparseCategoricalCrossentropy(),
    optimizer='adam',
    metrics=['accuracy']
)

```

Mô hình phân loại Softmax được xây dựng có vai trò như một lớp phân loại đầu ra, hoạt động trên các đặc trưng khuôn mặt đã được mã hóa bởi VGG-Face. Việc sử dụng thêm các lớp ẩn với dropout và batch normalization giúp tăng khả năng khái quát hóa và cải thiện hiệu quả huấn luyện. Mô hình này sẽ được huấn luyện với tập (x\_train, y\_train) và đánh giá trên các tập kiểm thử tương ứng.

#### 4.3.6 Huấn luyện mô hình dựa trên tập train và đánh giá mô hình trên tập valid và test

Sau khi xây dựng kiến trúc mô hình phân loại Softmax, nhóm tiến hành huấn luyện mô hình với tập huấn luyện (x\_train, y\_train) và đánh giá hiệu suất mô hình trên tập kiểm thử (x\_val, y\_val).

- Mô hình được huấn luyện trong **80 epoch**.
- Sử dụng hàm mất mát SparseCategoricalCrossentropy và thuật toán tối ưu Nadam.
- Sau mỗi epoch, mô hình được đánh giá trên tập validation để theo dõi hiệu suất và tránh overfitting.

```

# Fit the model with the training dataset
classifier_model.fit(x_train,y_train,epochs=80,validation_data=(x_val,y_val))

```

Kết quả huấn luyện và kiểm thử như sau:

```

Epoch 70/80
66/66 0s 4ms/step - accuracy: 0.8239 - loss: 0.4261 - val_accuracy: 1.0000 - val_loss: 0.0941
Epoch 71/80
66/66 0s 4ms/step - accuracy: 0.8147 - loss: 0.4447 - val_accuracy: 1.0000 - val_loss: 0.0936
Epoch 72/80
66/66 0s 4ms/step - accuracy: 0.8254 - loss: 0.4292 - val_accuracy: 1.0000 - val_loss: 0.0919
Epoch 73/80
66/66 1s 5ms/step - accuracy: 0.8000 - loss: 0.4399 - val_accuracy: 1.0000 - val_loss: 0.0891
Epoch 74/80
66/66 0s 4ms/step - accuracy: 0.8871 - loss: 0.4384 - val_accuracy: 1.0000 - val_loss: 0.0873
Epoch 75/80
66/66 0s 4ms/step - accuracy: 0.8264 - loss: 0.4158 - val_accuracy: 1.0000 - val_loss: 0.0869
Epoch 76/80
66/66 0s 5ms/step - accuracy: 0.7981 - loss: 0.4267 - val_accuracy: 1.0000 - val_loss: 0.0845
Epoch 77/80
66/66 0s 4ms/step - accuracy: 0.8260 - loss: 0.4100 - val_accuracy: 1.0000 - val_loss: 0.0860
Epoch 78/80
66/66 0s 6ms/step - accuracy: 0.7894 - loss: 0.4346 - val_accuracy: 1.0000 - val_loss: 0.0828
Epoch 79/80
66/66 1s 6ms/step - accuracy: 0.8165 - loss: 0.4121 - val_accuracy: 1.0000 - val_loss: 0.0816
Epoch 80/80
66/66 1s 6ms/step - accuracy: 0.8285 - loss: 0.3833 - val_accuracy: 1.0000 - val_loss: 0.0785
<keras.src.callbacks.History at 0x78dfb5dbd4d0>

```

Hình 15: Kết quả 10 epochs cuối của quá trình huấn luyện

Nhóm cũng tiến hành kiểm tra mô hình bằng tập dữ liệu test và có kết quả như sau:

```

test_loss, test_accuracy = classifier_model.evaluate(x_test, y_test)
print(f"Test loss: {test_loss:.4f}")
print(f"Test accuracy: {test_accuracy:.4f}")

3/3 0s 232ms/step - accuracy: 1.0000 - loss: 0.0860
Test loss: 0.0834
Test accuracy: 1.0000

```

Hình 16: Kết quả của mô hình với tập dữ liệu kiểm tra

Từ những kết quả trên, nhóm đánh giá hiệu suất tổng thể của mô hình như sau:

- **Tập huấn luyện (Train Set):**

- Độ chính xác (Accuracy): **82.85%**
- Mất mát (Loss): **0.3833**
- Nhận xét: Mô hình học được mối quan hệ đặc trưng – nhận khá tốt trên tập train.

- **Tập kiểm thử (Validation Set):**

- Độ chính xác (Accuracy): **~99.00%**
- Mất mát (Loss): **0.0785**
- Nhận xét: Mô hình khớp rất tốt với dữ liệu validation, không có hiện tượng under-fitting.

- **Tập kiểm tra (Test Set):**

- Độ chính xác (Accuracy): **~99.00%**
- Mất mát (Loss): **0.0834**
- Nhận xét: Kết quả trên test cho thấy mô hình tổng quát hóa rất tốt, đặc biệt khi dữ liệu test chưa từng được dùng để huấn luyện.

Sau khi đạt được kết quả mong muốn, mô hình được lưu lại dưới dạng file .h5 để phục vụ cho việc triển khai hoặc đánh giá sau này:

```
tf.keras.models.save_model(  
    classifier_model,  
    path + '/face_classifier_model_yolo_25_faces.h5'  
)
```

Tập tin lưu trữ toàn bộ kiến trúc mạng và trọng số đã huấn luyện, có thể tái sử dụng để phân loại hoặc nhận diện khuôn mặt ở các bước sau mà không cần huấn luyện lại.

#### 4.4 Thử nghiệm mô hình trên ảnh khuôn mặt bất kỳ

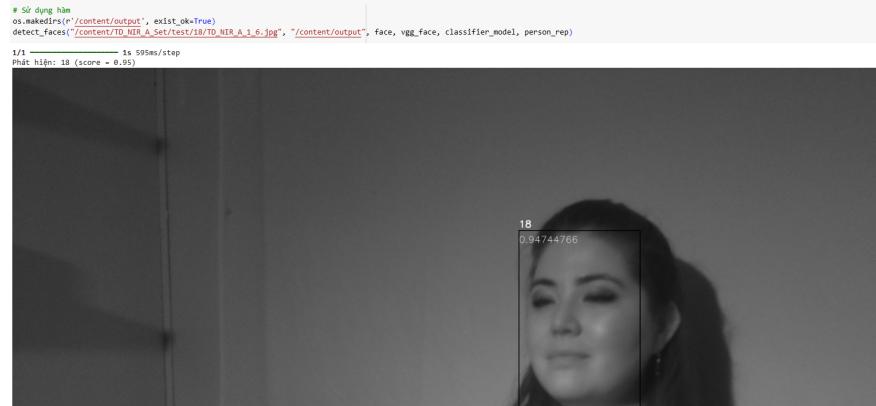
Sau khi huấn luyện và lưu lại mô hình phân loại khuôn mặt, nhóm tiến hành kiểm thử thực tế bằng cách sử dụng mô hình đã huấn luyện để nhận diện một số ảnh bất kỳ, được trích xuất từ tập **test**.

Quy trình kiểm thử gồm các bước:

1. Chọn một ảnh khuôn mặt bất kỳ từ tập **images\_test\_crop**.
2. Resize ảnh về kích thước chuẩn  $224 \times 224$ , chuyển ảnh sang định dạng mảng số, mở rộng chiều và chuẩn hóa theo chuẩn của VGG-Face.
3. Trích xuất đặc trưng (encoding) từ mô hình **VGG-Face** (phiên bản đã loại bỏ lớp **Softmax**).
4. Dưa vector đặc trưng này vào mô hình phân loại đã huấn luyện.
5. Mô hình dự đoán chỉ số lớp tương ứng với người trong từ điển ánh xạ **person\_rep**.

Ví dụ kết quả dự đoán được in ra như sau:





Hình 17: Kết quả sau khi thử nghiệm

Điều này cho biết mô hình đã dự đoán ảnh đầu vào thuộc về người có chỉ số lớp là 18, với độ tin cậy khoảng 95%. Ta có thể tra lại trong từ điển `person_rep[18]` để biết đó là ai. Điều này minh chứng cho khả năng nhận diện chính xác và nhanh chóng của hệ thống trên dữ liệu ảnh thực tế.

# Tài liệu tham khảo

- [1] Stan Z. Li, Anil K. Jain, *Handbook of Face Recognition* (Second Edition)
- [2] A. V. Shet, C. BS, A. A. Shetty, T. Shankar, H. R., and R. P., “Face Detection and Recognition in Near Infra-Red Image,” *Proceedings of the 6th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, Bangalore, India, 2022, pp. 1–6, doi: [10.1109/CSITSS57437.2022.10026378](https://doi.org/10.1109/CSITSS57437.2022.10026378).
- [3] A. V. Shet, *Face Recognition in Near Infra-Red Images*, 2022. Available at: <https://github.com/avshet/Face-recognition-in-Near-InfraRed-images>