

Đại học Quốc gia TP. HCM  
Trường Đại học Khoa học Tự nhiên  
Bộ Môn Khoa học Máy tính



---

**BÁO CÁO SEMINAR CUỐI KÌ  
FACE RECOGNITION**

Nhận Dạng (CSC14006)

Nhóm 9

---

TP Hồ Chí Minh, ngày 01/06/2021

## Contents

<b>1</b>	<b>Thông tin nhóm</b>	<b>3</b>
<b>2</b>	<b>Phân công các thành viên trong nhóm</b>	<b>3</b>
<b>3</b>	<b>Mở Đầu</b>	<b>4</b>
<b>4</b>	<b>Các Kỹ Thuật Cơ Bản</b>	<b>5</b>
4.1	Phân tích thành phần chính (Principal Component Analysis - PCA) . . . . .	5
4.1.1	Tổng quan về PCA . . . . .	5
4.1.2	Cách bước tính của PCA . . . . .	5
4.1.3	Ví dụ minh họa . . . . .	6
4.2	Phân tích biệt thức tuyến tính (Linear Discriminant Analysis - LDA) . . . . .	8
4.2.1	Tổng quan về LDA . . . . .	8
4.2.2	Các bước tính của LDA . . . . .	8
4.2.3	Ví dụ minh họa . . . . .	9
4.3	Support Vector Machine – SVM . . . . .	10
4.3.1	Tổng quan về SVM . . . . .	10
4.3.2	Các bước tính của SVM . . . . .	10
4.3.3	Ví dụ minh họa SVM . . . . .	10
4.3.4	Một số hàm kernel thông dụng . . . . .	11
<b>5</b>	<b>Các cơ sở dữ liệu</b>	<b>12</b>
5.1	FRGC database . . . . .	12
5.2	FERET database . . . . .	13
5.3	PIE database . . . . .	13
5.4	AR database . . . . .	14
5.5	Yale Face Database . . . . .	15
5.6	MS-CELEB-1M . . . . .	15
5.7	LFW Database . . . . .	16

5.8	AgeDB Database . . . . .	17
5.9	CFPW Database . . . . .	18
<b>6</b>	<b>Mô hình nhận dạng gương mặt ArcFace</b>	<b>18</b>
6.1	Convolutional Neural Network (CNNs) . . . . .	18
6.1.1	Fully Connected Neural Network . . . . .	18
6.1.2	Convolution Operation . . . . .	20
6.1.3	Hàm Kích Hoạt . . . . .	21
6.1.4	Pooling . . . . .	22
6.1.5	CNNs trong bài toán phân lớp . . . . .	23
6.2	Residual Network (ResNet) . . . . .	25
6.2.1	Vấn đề độ sâu của các mạng học sâu . . . . .	25
6.2.2	Skip Connection . . . . .	25
6.2.3	Residual Network . . . . .	26
6.2.4	Bottleneck Block . . . . .	27
6.2.5	Overfitting . . . . .	27
6.3	Additive Angular Margin Loss (ArcFace) . . . . .	28
6.3.1	Hạn chế của hàm lỗi Softmax . . . . .	28
6.3.2	Center Loss . . . . .	29
6.3.3	Additive Angular Margin Loss (ArcFace) . . . . .	30
6.4	Evaluation & Demo . . . . .	32
6.4.1	Demo . . . . .	32
6.5	Polynomial Neural Network(Π-Net) . . . . .	32
6.5.1	Xấp xỉ bằng hàm đa thức . . . . .	32
6.5.2	ProdPoly . . . . .	33
<b>7</b>	<b>Tổng kết</b>	<b>35</b>
<b>8</b>	<b>Tham khảo</b>	<b>35</b>

## 1 Thông tin nhóm

STT	MSSV	Họ tên	Email
1	18120078	Ngô Phù Hữu Đại Sơn	18120078@student.hcmus.edu.vn
2	18120533	Dương Đoàn Bảo Sơn	18120533@student.hcmus.edu.vn
3	18120164	Lê Minh Đức	18120164@student.hcmus.edu.vn

Table 1: Bảng danh sách thành viên nhóm

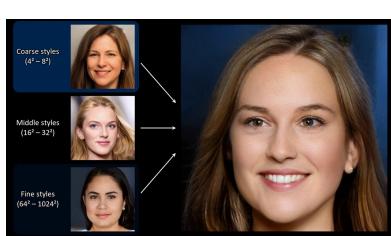
## 2 Phân công các thành viên trong nhóm

STT	Họ tên	Công việc tham gia	Hoàn thành (%)
1	Ngô Phù Hữu Đại Sơn	CNNs, ResNet và ArcFace loss	100/100
2	Dương Đoàn Bảo Sơn	Databases	100/100
5	Lê Minh Đức	Các kỹ thuật cơ bản nhận dạng	100/100

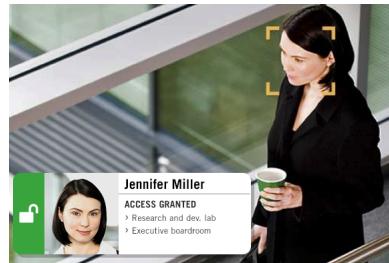
Table 2: Bảng phân tích tỷ lệ hoàn thành công việc

### 3 Mở Đầu

Các hệ thống nhận dạng khuôn mặt ngày nay vai trò quan trọng trong nhiều lĩnh vực ví dụ như giúp nhận dạng các phần tử khủng bố. Ngoài ra chúng còn giúp tăng tính bảo mật của các hệ thống bảo mật ngày nay. Tuy nhiên để có thể xây dựng được một hệ thống nhận diện khuôn mặt hiệu quả vẫn còn là một thách thức trong lĩnh vực nghiên cứu. Những vấn đề mà ta đang gặp phải như sự thay đổi của gương mặt thông qua biểu cảm, tư thế gương mặt, lão hóa, trang điểm, sử dụng mắt kính, khẩu trang,... Ngoài ra các điều kiện chiếu sáng và góc chụp khác nhau cũng dẫn đến các lỗi trong nhận diện. Trong báo cáo này, nhóm em xin trình bày các kĩ thuật cơ bản và cổ điển trong các bài toán nhận diện khuôn mặt như *Principal Component Analysis (PCA)*, *Linear Disciminant Analysis (LDA)* và *Support Vector Machine (SVM)*. Sau đó, nhóm sẽ trình bày các tập dữ liệu kinh điển (*FGRC*, *FERET*,...) và một số tập dữ liệu mới ngày nay (*LFW*, *CFP-FP*, *AgeDB*,...). Cuối cùng, nhóm em xin trình bày kĩ thuật mới và hiện đại hơn trong nhận diện gương mặt là *Convolutional Neural Network(CNNs)* - cấu trúc mạng học sâu quen thuộc ngày nay trong các bài toán về thị giác máy tính, *Residual Network (ResNet)* - Cấu trúc giúp tối ưu mạng học sâu có nhiều lớp và *ArcFace Loss* giúp các hệ thống phân lớp gương mặt hiệu quả hơn. Nhóm em còn thực hiện thêm một demo sử dụng *ArcFace* giải quyết bài toán *Face Verification* để có cái nhìn rõ hơn về phương pháp đã nêu trên.



(a) Face Generation



(b) Face Identification



(c) Face Verification

Figure 1: Một số ứng dụng của nhận diện gương mặt

## 4 Các Kĩ Thuật Cơ Bản

### 4.1 Phân tích thành phần chính (Principal Component Analysis - PCA)

#### 4.1.1 Tổng quan về PCA

PCA là kĩ thuật giảm chiều dữ liệu từ n chiều sang dữ liệu m chiều ( $m < n$ ) mà vẫn giữ được nhiều thông tin nhất có thể.

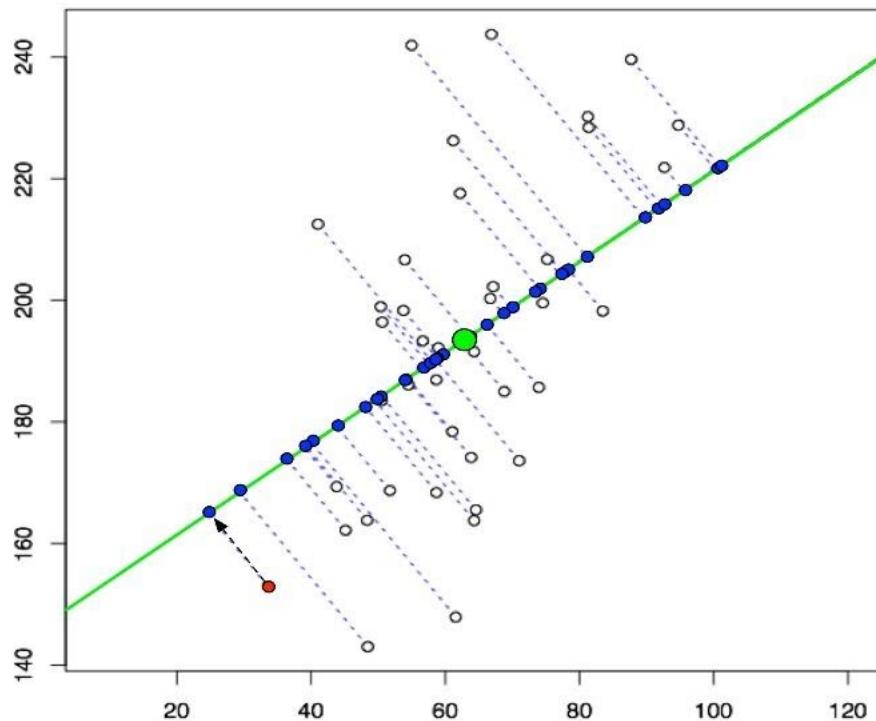


Figure 2: Minh họa PCA

#### 4.1.2 Cách bước tính của PCA

- Bước 1: Tính giá trị trung bình:  $\bar{x} = \frac{1}{N} \sum_{n=1}^N x_n$ .
- Bước 2: Chuẩn hóa:  $\hat{x} = x_n - \bar{x}$ .
- Bước 3: Tính ma trận hiệp phương sai:  $S = \frac{1}{N} \hat{X} \hat{X}^T$ .
- Bước 4: Tính các trị riêng  $\lambda_i$  và vector riêng  $v_i : S v_i = \lambda_i v_i$ .

- Bước 5: Chọn K vector riêng ứng với K trị riêng lớn nhất để xây dựng ma trận  $U_K$  có các cột tạo thành một hệ trực giao. K vector này, còn được gọi là các thành phần chính, tạo thành một không gian con gần với phân bố của dữ liệu ban đầu đã chuẩn hóa.
- Bước 6: Chiếu dữ liệu ban đầu đã chuẩn hóa X xuống không gian con tìm được. Dữ liệu mới chính là toạ độ của các điểm dữ liệu trên không gian mới:  $Z = U_K^T \hat{X}$

#### 4.1.3 Ví dụ minh họa

Có các điểm sau  $(2.5, 2.4), (0.5, 0.7), (2.2, 2.9), (1.9, 2.2), (3.1, 3.0), (2.3, 2.7), (2, 1.6), (1, 1.1), (1.5, 1.6), (1.1, 0.9)$ . Sử dụng PCA để giảm chúng về 1 chiều.

- Bước 1: Tính trung bình, chuẩn hóa.

	$x$	$y$		$x$	$y$
Data =	2.5	2.4		.69	.49
	0.5	0.7		-1.31	-1.21
	2.2	2.9		.39	.99
	1.9	2.2		.09	.29
	3.1	3.0	DataAdjust =	1.29	1.09
	2.3	2.7		.49	.79
	2	1.6		.19	-.31
	1	1.1		-.81	-.81
	1.5	1.6		-.31	-.31
	1.1	0.9		-.71	-1.01

Figure 3: Tính trung bình, chuẩn hóa

- Bước 2: Ma trận hiệp phương sai.

$$cov(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{(n - 1)}$$

$$cov = \begin{pmatrix} .616555556 & .615444444 \\ .615444444 & .716555556 \end{pmatrix}$$

Figure 4: Ma trận hiệp phương sai

- Bước 3: Tính vector riêng và giá trị riêng cho ma trận hiệp phương sai.

$$\text{eigenvalues} = \begin{pmatrix} .0490833989 \\ 1.28402771 \end{pmatrix}$$

$$\text{eigenvectors} = \begin{pmatrix} -.735178656 & -.677873399 \\ .677873399 & -.735178656 \end{pmatrix}$$

Figure 5: Ma trận hiệp phương sai

- Bước 4: Biến đổi dữ liệu:

```
[[ -0.82797019]
 [ 1.77758033]
 [-0.99219749]
 [-0.27421042]
 [-1.67580142]
 [-0.9129491 ]
 [ 0.09910944]
 [ 1.14457216]
 [ 0.43804614]
 [ 1.22382056]]
```

Figure 6: Dữ liệu sao khi biến đổi

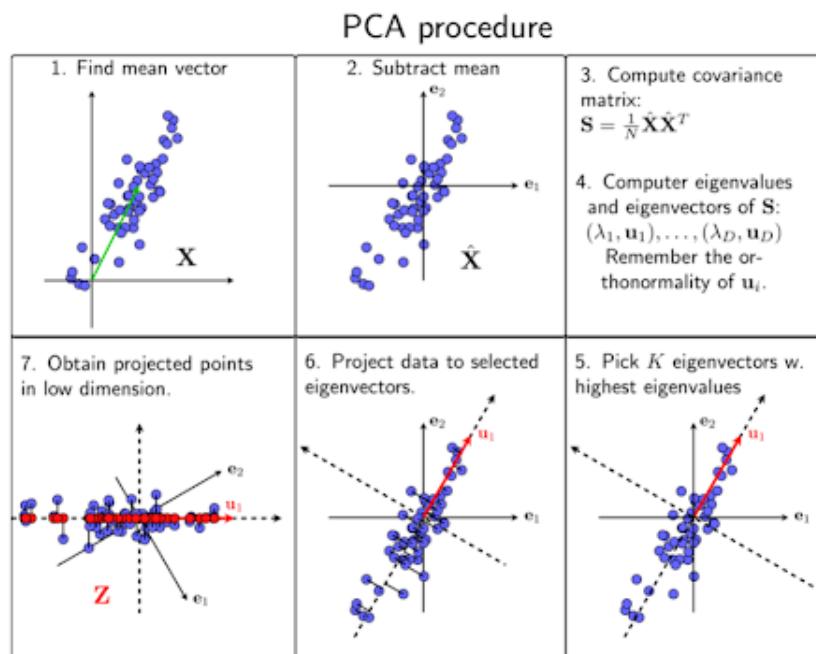


Figure 7: Tổng kết toàn bộ quá trình PCA

## 4.2 Phân tích biệt thức tuyến tính (Linear Discriminant Analysis - LDA)

### 4.2.1 Tổng quan về LDA

LDA là một thuật toán học có giám sát, giảm chiều dữ liệu.

Mục đích LDA là tìm sự khác nhau giữa các thành phần trong 1 class (within-class) là nhỏ và sự khác nhau giữa các classes là lớn.

Khác với PCA, LDA tìm phép chiếu sao cho tối đa hóa sự khác biệt giữa các lớp để có thể phân lớp hiệu quả.

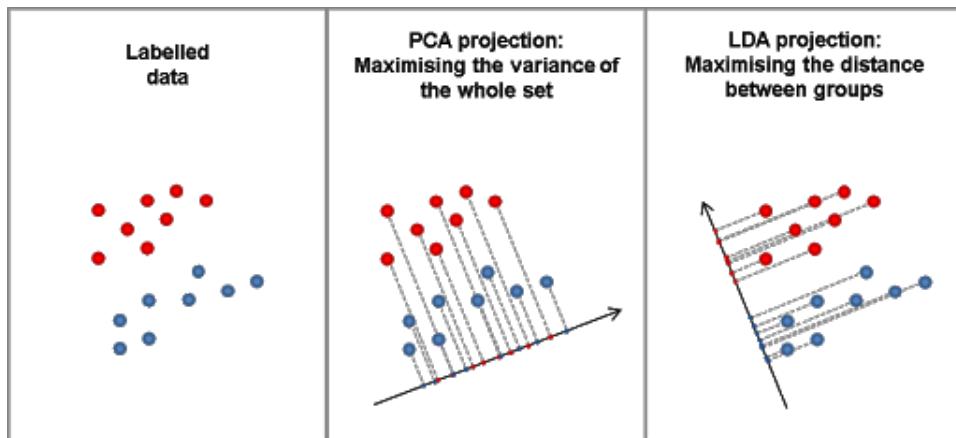


Figure 8: So sánh LDA và PCA (1)

### 4.2.2 Các bước tính của LDA

- Bước 1: Tính ma trận phân tán giữa các nhóm.

$$S_B = \sum_{i=1}^C n_i (\mu_i - \mu)(\mu_i - \mu)^T \quad (1)$$

$\mu_i$  là giá trị trung bình của từng lớp.

$\mu$  là giá trị trung bình của tất cả dữ liệu.

- Bước 2: Tính ma trận phân tán tích lũy ứng với từng nhóm.

$$S_W = \sum_{j=1}^C \sum_{i=1}^{n_j} (x_{ij} - \mu_j)(x_{ij} - \mu_j)^T \quad (2)$$

- Bước 3: Xây dựng hàm tiêu chí tách lớp.

$$W = S_W^{-1} S_B \quad (3)$$

- Bước 4: Dự đoán nhãn của mẫu dữ liệu nhập (so sánh vector trung bình của từng nhóm – gần vector trung bình nhất).

### 4.2.3 Ví dụ minh họa

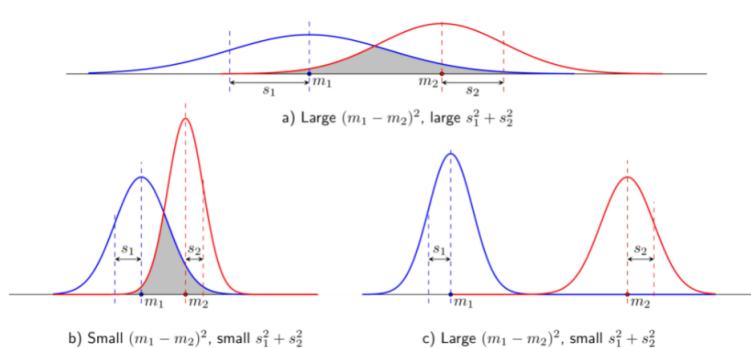
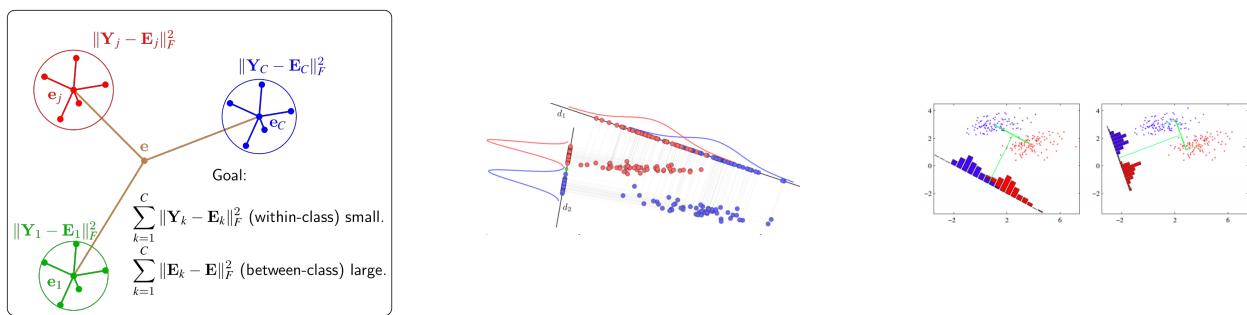


Figure 9: Ví dụ minh họa LDA

Hình trên là khoảng cách giữa các kỳ vọng và tổng các phương sai ảnh hưởng tới độ phân biệt của dữ liệu. a) Khoảng cách giữa hai kỳ vọng là lớn nhưng phương sai trong mỗi class cũng lớn, khiến cho hai phân phối chồng lấn lên nhau (phần màu xám). b) Phương sai cho mỗi class là rất nhỏ nhưng hai kỳ vọng quá gần nhau, khiến khó phân biệt 2 class. c) Khi phương sai đủ nhỏ và khoảng cách giữa hai kỳ vọng đủ lớn, ta thấy rằng dữ liệu phân biệt hơn.



(a) Tối ưu khoảng cách trong LDA

(b) So sánh PCA và LDA (2)

(c) So sánh PCA và LDA (3)

Figure 10: Một số hình ảnh ví dụ về LDA

## 4.3 Support Vector Machine – SVM

### 4.3.1 Tổng quan về SVM

SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc hồi quy.

SVM là tìm một siêu phẳng (hyperplane) để phân tách các điểm dữ liệu thành 2 lớp riêng biệt, ta cần phải tối ưu hóa siêu phẳng này.

SVM dùng thủ thuật để ánh xạ bộ dữ liệu đó vào không gian nhiều chiều hơn ( $n$  chiều), từ đó tìm ra siêu phẳng (hyperplane) để phân chia.

Đối với dữ liệu 2 chiều, hyperplane là 1 đường thẳng; dữ liệu 3 chiều là 1 mặt phẳng, ...

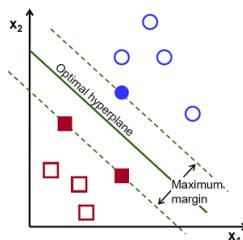


Figure 11: Minh họa SVM

### 4.3.2 Các bước tính của SVM

- Chọn hàm Kernel.
- Chọn giá trị điều khiển biến cho dữ liệu huấn luyện  $C$ .
- Bài toán tối ưu bậc hai để tìm tham số cho vector hỗ trợ.
- Xây dựng hàm tách lớp từ các vector hỗ trợ.

### 4.3.3 Ví dụ minh họa SVM

- Đối với dữ liệu tuyến tính:

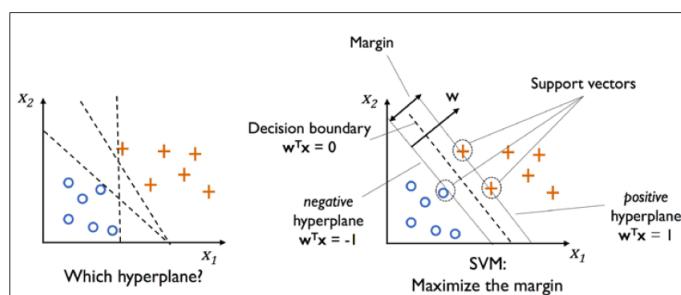


Figure 12: SVM với dữ liệu tuyến tính

- Đối với dữ liệu phi tuyến ta không thể chia trực tiếp bằng các đường thẳng, ta cần phải sử dụng hàm Kernel để ánh xạ dữ liệu đó vào không gian nhiều chiều hơn:

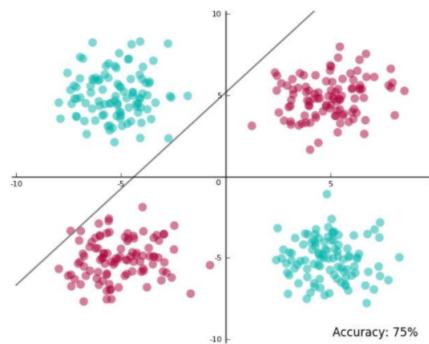


Figure 13: SVM với dữ liệu phi tuyến

Ta sử dụng hàm Kernel để ánh xạ tập dữ liệu 2 chiều trên thành dữ liệu 3 chiều từ đó dễ dàng tìm ra siêu phẳng để phân lớp hơn.

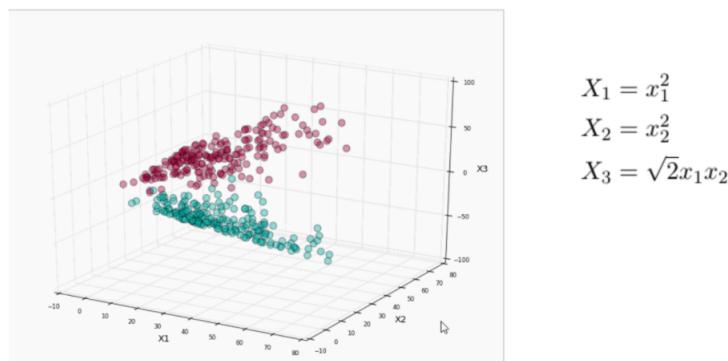


Figure 14: Tăng chiều dữ liệu bằng các hàm kernel

#### 4.3.4 Một số hàm kernel thông dụng

- Hàm tuyến tính:  $K(x_i, x_j) = x_i^T x_j$ .
- Hàm đa thức:  $K(x_i, x_{-j}) = (1 + x_i^T x_j)^p$
- Gaussian:  $K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$ .
- Sigmoid:  $K(x_i, x_j) = \tanh(\beta_0 x_i^T x_j + \beta_1)$

## 5 Các cơ sở dữ liệu

### 5.1 FRGC database

Dữ liệu cho Face Recognition Grand Challenge database (FRGC) là bộ dataset lớn bao gồm 50.000 bản ghi được chia thành tập training và validation.

Tập validation bao gồm dữ liệu từ 4.003 subject sessions. Một subject sessions là tập hợp tất cả các hình ảnh của một người được chụp qua mỗi lần thu thập dữ liệu sinh trắc học và gồm có 4 hình ảnh tĩnh được kiểm soát, 2 hình ảnh tĩnh không được kiểm soát và 1 hình ảnh ba chiều.

Các hình ảnh được kiểm soát được chụp trong bối cảnh studio, là hình ảnh toàn bộ khuôn mặt trực diện được chụp trong hai điều kiện ánh sáng và với hai biểu cảm khuôn mặt (tươi cười và trung tính).

Các hình ảnh không được kiểm soát được chụp trong các điều kiện ánh sáng khác nhau; ví dụ: hành lang, nhĩ thất hoặc bên ngoài. Mỗi bộ ảnh không kiểm soát có hai biểu cảm (tươi cười và trung tính).

Hình ảnh 3D được chụp trong điều kiện ánh sáng được kiểm soát. Hình ảnh 3D được thu thập bởi Minolta Vivid 900/910 series sensor.



Figure 15: Ảnh chụp có kiểm soát (trái) và ảnh chụp không kiểm soát (phải)

## 5.2 FERET database

The Facial Recognition Technology (FERET) database được sử dụng để đánh giá hệ thống nhận dạng khuôn mặt là một phần của the Face Recognition Technology program.

FERET database được thu thập trong từ tháng 12 năm 1993 đến tháng 8 năm 1996.

Cơ sở dữ liệu chứa 1564 bộ ảnh trong tổng số 14,126 ảnh bao gồm 1199 người và 365 bộ ảnh trùng lặp.

Bộ ảnh trùng lặp là tập hợp hình ảnh thứ hai của một người đã có trong cơ sở dữ liệu và thường được chụp vào một ngày khác.



Figure 16: Ảnh của các đối tượng được chụp trong ngày khác nhau

## 5.3 PIE database

CMU Pose, Illumination and Expression (PIE) database được thu thập giữa tháng 10 và 12/2000

PIE database chứa 41.368 hình ảnh khuôn mặt của 68 người.

Các hình ảnh có được trên các tư thế, dưới các ánh sáng và với các nét mặt khác nhau.

Hình ảnh của mỗi người được chụp dưới 13 tư thế khác nhau, 43 điều kiện chiếu sáng khác nhau và 4 loại nét mặt.



(a) Sample poses among 13 poses



(b) Sample lighting conditions among 43 different lighting conditions

**Fig. 3.5.** Sample images of the PIE database.

Figure 17: Ảnh trong PIE

## 5.4 AR database

AR database được tạo ra bởi Aleix Martinez and Robert Benavente tại the Computer Vision Center (CVC) của the U.A.B.

Gồm hơn 4000 ảnh màu tương ứng với khuôn mặt của 126 người(70 nam và 56 nữ).

Hình ảnh trực diện khuôn mặt với các biểu cảm, điều kiện ánh sáng và sự che khuất(kính và khăn quàng cổ) khác nhau.



Figure 18: Ảnh trong AR

## 5.5 Yale Face Database

Gồm 165 ảnh grayscale ở định dạng GIF của 15 cá nhân.

Có 11 hình ảnh với mỗi cá nhân, cho từng biểu hiện hoặc cấu hình khuôn mặt sau: center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, sleepy, surprised, and wink.

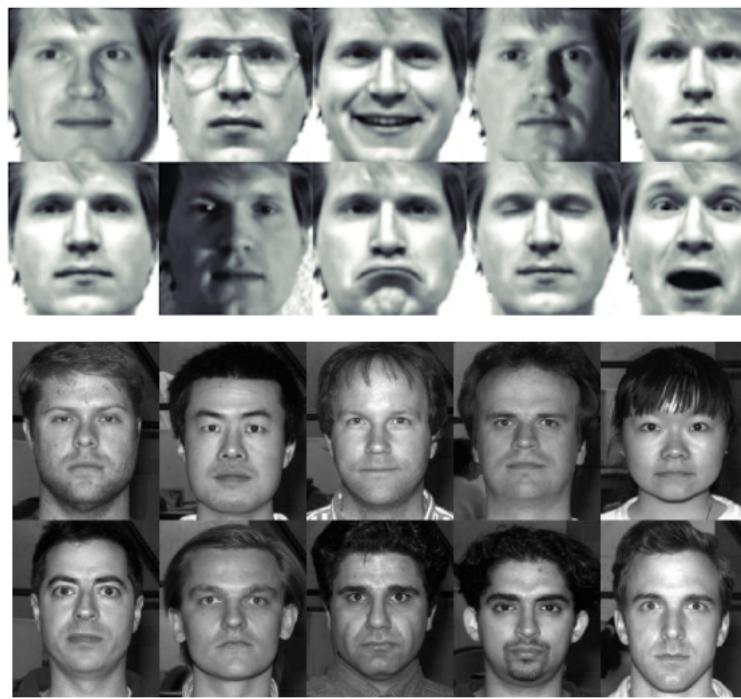


Figure 19: Ảnh trong Yale

## 5.6 MS-CELEB-1M

Là tập dữ liệu nhận dạng khuôn mặt quy mô lớn bao gồm 100K danh tính và mỗi danh tính có khoảng 100 hình ảnh khuôn mặt. Được lấy từ các công cụ tìm kiếm phổ biến.

Mỗi người được định danh bằng một unique entity key, được liên kết với vài thông tin bao gồm ngày sinh, nghề nghiệp, nơi sinh,...

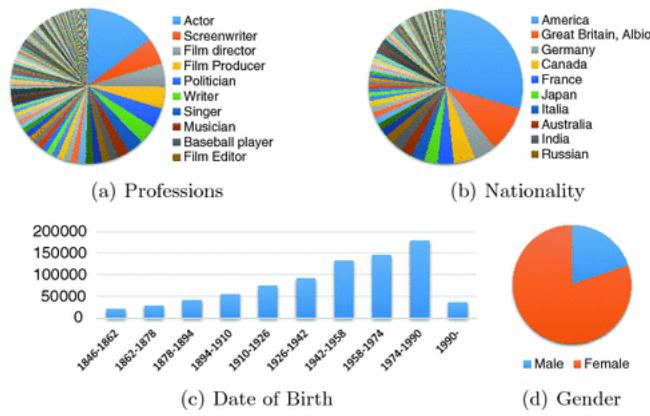


Figure 20: Biểu đồ phân bố dữ liệu trong MS-CELEB-1M

## 5.7 LFW Database

Bộ dữ liệu bao gồm hơn 13.000 hình ảnh về các khuôn mặt được thu thập từ web của 5749 người có 1 ảnh và 1680 người có 2 ảnh trở lên. Mỗi khuôn mặt đã được dán nhãn với tên của người trong hình thường được chia thành 2 tập training set và testing set.

Training set (pairsDevTrain.txt) bao gồm 1100 cặp matched images và 1100 cặp mismatched images.

Testing set (pairsDevTest.txt) bao gồm 500 cặp matched images và 500 cặp mismatched images.

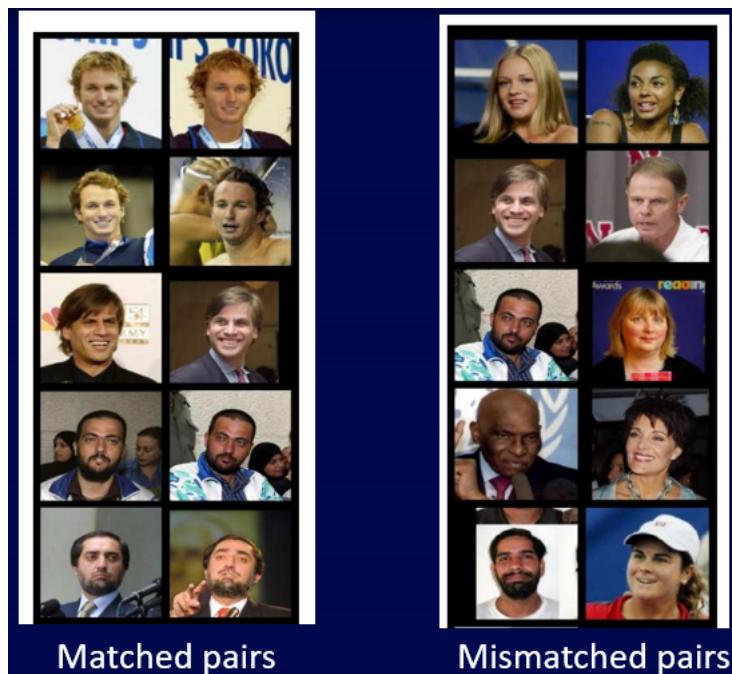


Figure 21: Ảnh trong LFW

## 5.8 AgeDB Database

Dữ liệu được thu thập ở môi trường tự nhiên gồm 16,488 hình ảnh về những người nổi tiếng khác nhau, chẳng hạn như diễn viên, nhà văn, nhà khoa học, chính trị gia, v.v ... Tổng cộng 568 đối tượng.

Số lượng ảnh trung bình cho mỗi đối tượng là 29.

Mỗi hình ảnh đều được chú thích liên quan đến danh tính, thuộc tính tuổi và giới tính.

Độ tuổi tối thiểu và tối đa lần lượt là 1 và 101. Độ tuổi trung bình của mỗi đối tượng là 50,3 tuổi.

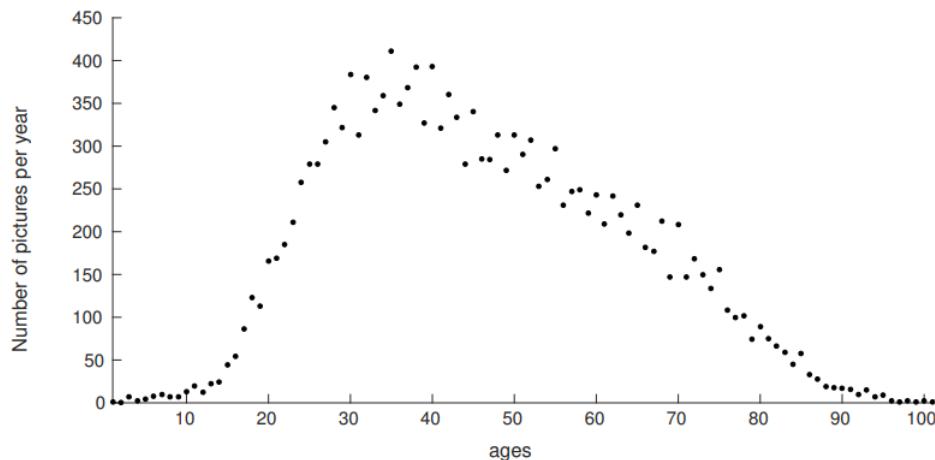


Figure 22: Phân bố độ tuổi trong AgeDB



Figure 23: Ảnh trong AgeDB

## 5.9 CFPW Database

Tập dữ liệu bao gồm 10 ảnh trực diện (Frontal) và 4 ảnh mặt nghiêng (Profile) của 500 cá nhân nổi tiếng.

Được tách ra 10 phần: mỗi phần gồm 350 cặp giống nhau và 350 cặp khác nhau (tương tự như LFW).

CFP-FP(Frontal-Profile), CFP-FF(Frontal-Frontal) là 2 bộ thử nghiệm được tạo từ CFP database với cách bắt cặp khác nhau (F-P) và (F-F).

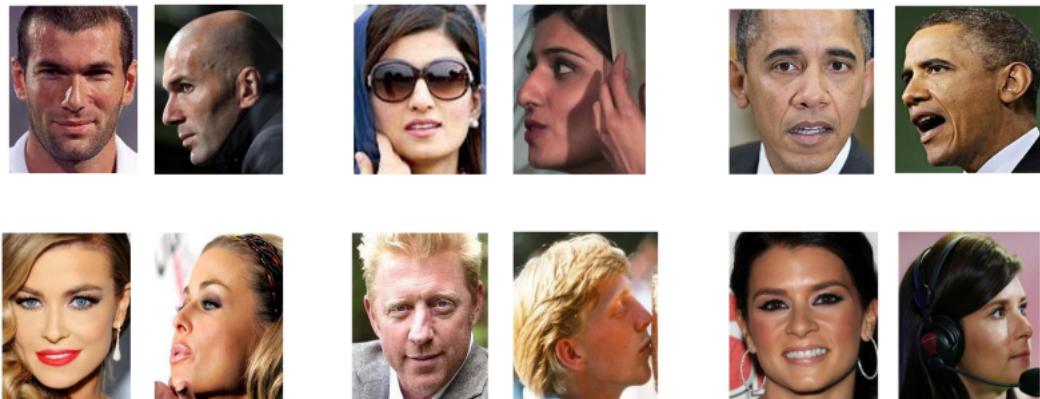


Figure 24: Ảnh mẫu một số người nổi tiếng trong CFP

## 6 Mô hình nhận dạng khuôn mặt ArcFace

### 6.1 Convolutional Neural Network (CNNs)

#### 6.1.1 Fully Connected Neural Network

Fully Connected Neural Network (FCNN) là một mô hình mạng học sâu gồm có nhiều lớp ẩn (hidden layers). Mỗi lớp ẩn sẽ kết nối hoàn toàn (*fully connected*) với lớp trước đó. Kết nối hoàn toàn nghĩa là mọi đầu vào sẽ kết nối với mọi đầu ra.

Giả sử chúng ta sử dụng FCNN để giải quyết bài toán nhận dạng khuôn mặt. Ta cần thực hiện các bước sau:

- Duỗi bức ảnh cần nhận diện thành 1 vector có kích thước (width \* height, 1).
- Dưa vector đó vào input của FCNN.

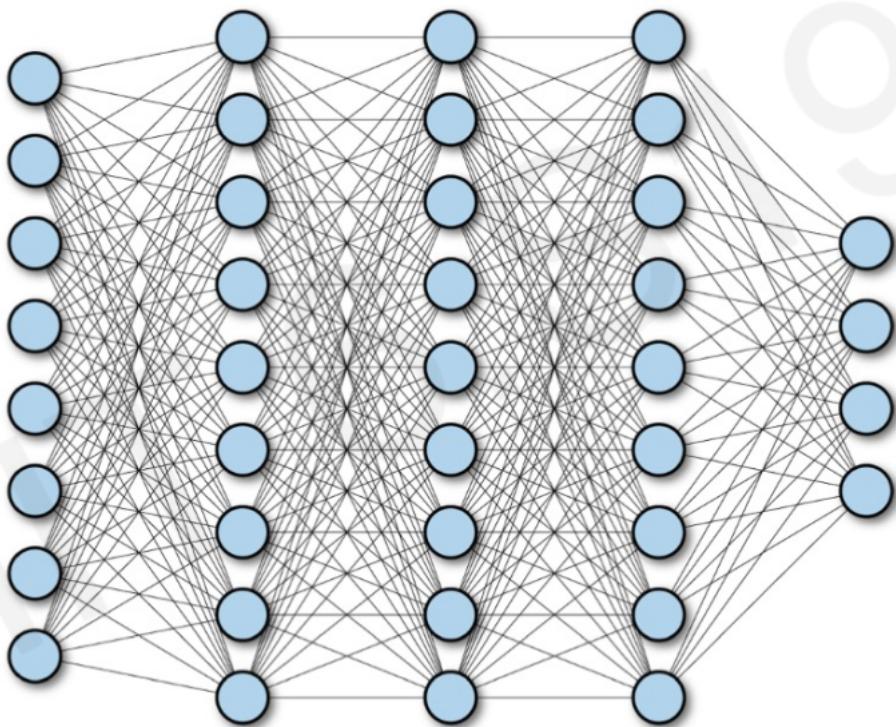


Figure 25: Cấu trúc FCNN

Nhược điểm của FCNN có thể thấy dễ dàng được là:

- Số lượng tham số quá nhiều, dẫn đến hao tốn nhiều tài nguyên bộ nhớ và tài nguyên tính toán của máy tính.
- Cấu trúc đuôi thẳng ảnh để đưa vào mạng như trên sẽ làm mất hết các đặc trưng không gian (*spacial features*) của ảnh. Trong khi đó, những bài toán liên quan đến nhận diện ảnh thì các đặc trưng không gian đóng một vai trò rất quan trọng.

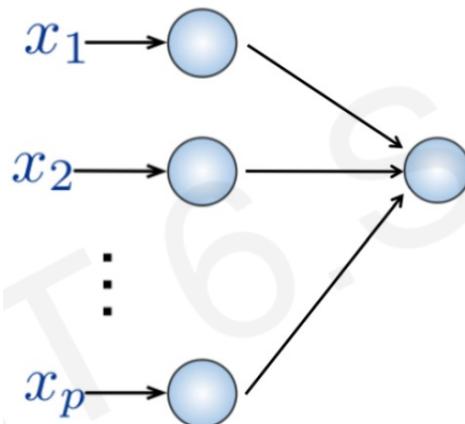


Figure 26: Input ảnh vào FCNN

Ta cần dùng một cách nào đó để có thể đưa những đặc trưng không gian vào mô hình mạng học sâu, từ đó làm cho kết quả phân lớp ảnh của ta được cải thiện nhiều hơn.

### 6.1.2 Convolution Operation

Trong phép tích chập (*Convolution Operation*), đầu vào là ảnh có kích thước  $n \times n$  (F), ta sử dụng 1 bộ lọc (filter hay kernel) có kích thước  $f \times f$  (h), để xuất ra ảnh có kích thước  $(n - f + 1) \times (n - f + 1)$  (G).

$$G = h * F \quad (4)$$

$$G[i, j] = \sum_{u=-f}^f \sum_{v=-f}^f h[u, v] F[i - u, j - v] \quad (5)$$

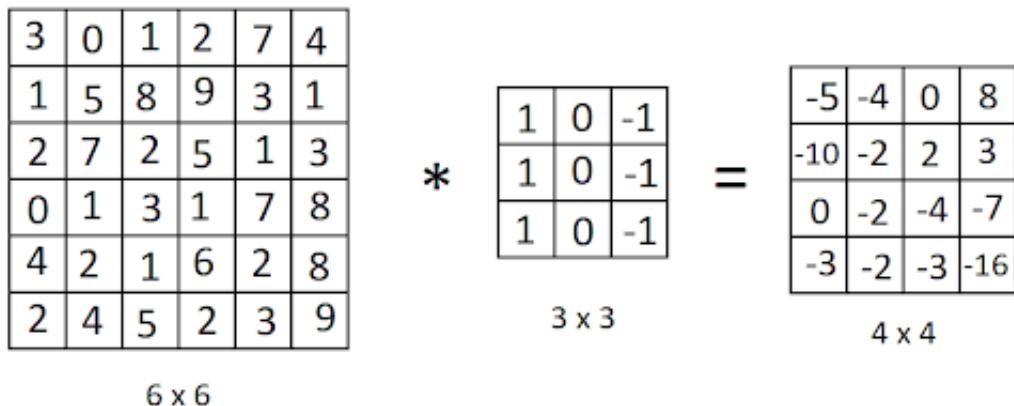


Figure 27: Minh họa phép tích chập

Vấn đề có thể thấy được ở phép tích chập đó là ảnh đầu ra ( $G$ ) có kích thước ngày càng nhỏ. Nếu ảnh của ta đi qua nhiều lớp sẽ dẫn đến kích thước ảnh nhỏ dần làm mờ đi thông tin.

Ngoài ra, các điểm ảnh ở phần biên của ảnh đóng góp ít hơn vào các phép tích chập so với các phần tử phía bên trong của ảnh. Làm mất đi những thông tin hữu ích ở phần biên của ảnh.

Để giải quyết vấn đề trên, người ta thêm vào xung quanh ảnh những điểm ảnh có giá trị là 0 (Có nhiều các khác nhau như giá trị mặc định hoặc điểm ảnh gần nhất), độ dày của phần điểm ảnh được thêm vào gọi là *Padding* ( $p$ )

Ngoài ra, trong một số trường hợp, ta muốn kết quả của phép tích chập cho ra ảnh có kích thước nhỏ hơn so với ảnh đầu vào. Ta định nghĩa *Stride*(s) là số lượng pixel mà cửa sổ sẽ trượt qua khi thực hiện phép tích chập. Công dụng của *Stride* là giảm được số lượng tham số cần phải sử dụng, nhờ đó mà tiết kiệm tài nguyên máy tính.

Nhờ vào *Padding* và *Stride*, ta có thể điều khiển được kích thước của G. Ta có công thức:

$$\left\lceil \frac{n + 2p - f}{s} + 1 \right\rceil \times \left\lceil \frac{n + 2p - f}{s} + 1 \right\rceil \quad (6)$$

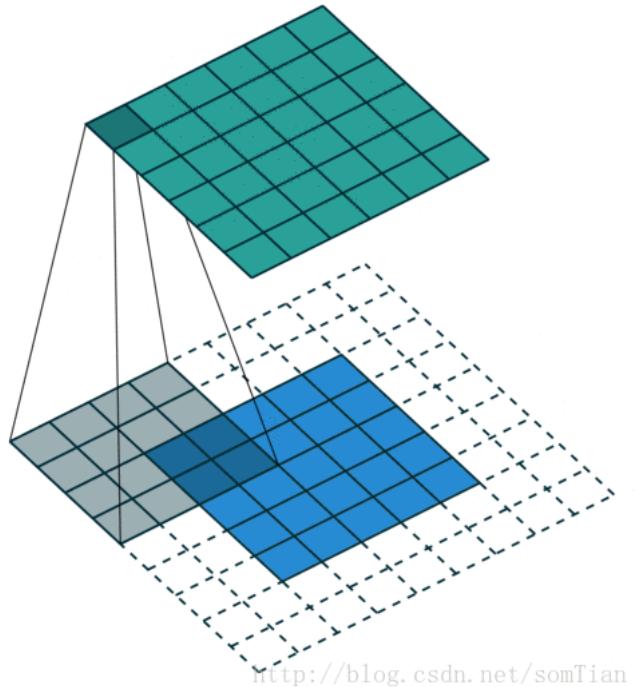


Figure 28: Padding và Stride

### 6.1.3 Hàm Kích Hoạt

Sau khi thực hiện phép tích chập, ảnh đầu ra (G) được đưa vào 1 hàm kích hoạt (phi tuyến). Hàm này sẽ áp dụng trên mọi điểm ảnh của G và xuất ra ảnh có cùng kích thước với G.

Có nhiều loại hàm kích hoạt khác nhau. Trong đó *Rectified Linear Unit (ReLU)* được sử dụng phổ biến đối với mạng CNNs ngày nay. Các điểm ảnh có giá trị âm khi vào ReLU sẽ được chuyển thành 0, các điểm ảnh có giá trị lớn hơn 0 sẽ được giữ nguyên giá trị.

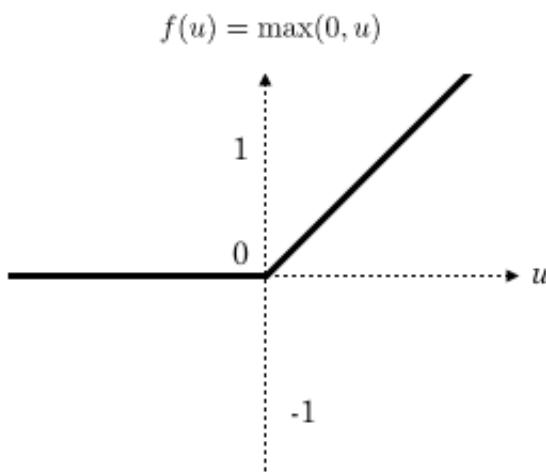


Figure 29: Hàm kích hoạt ReLU

Tác dụng của các hàm kích hoạt nói chung và của *ReLU* nói riêng là giúp giới hạn giá trị của các nút mạng, tránh tình trạng bùng nổ giá trị ở các neuron. Ngoài ra, chúng còn giúp khử nhiễu, làm nổi bật lên các đặc trưng trong ảnh. Và một vấn đề nữa là hầu như các bài toán học máy thực tế đều là phi tuyến.

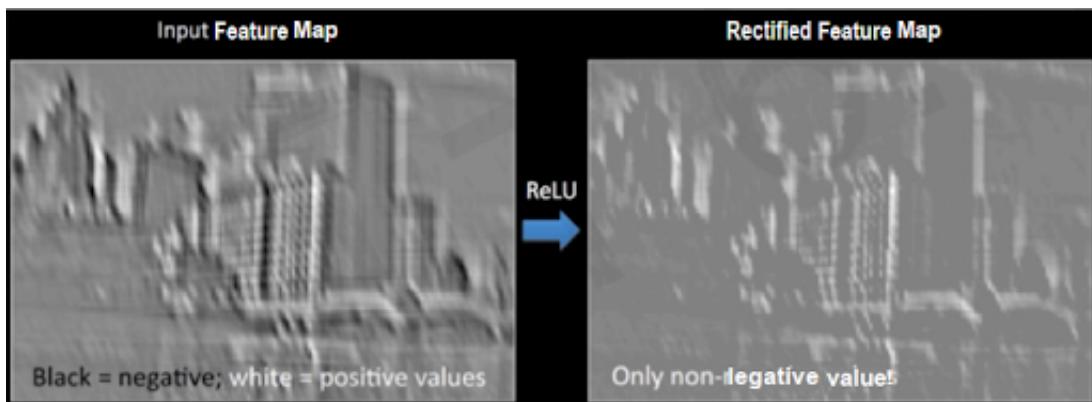


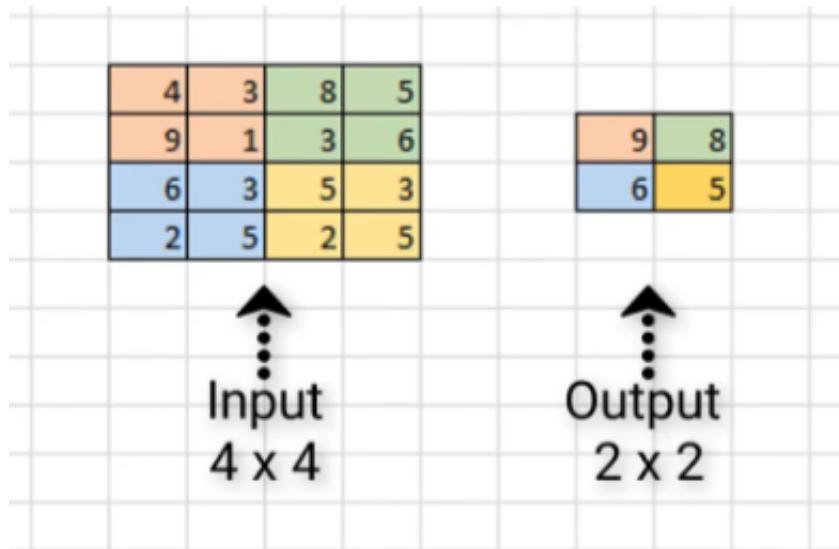
Figure 30: Thực hiện ReLU lên ảnh

#### 6.1.4 Pooling

Một thành phần quan trọng khác trong một mô hình CNN là các lớp *Pooling*. Pooling có chức năng giảm chiều của dữ liệu. Và có thể thực hiện phía sau một hoặc một vài lớp CONV (Convolution layers).

Trong các kỹ thuật *Pooling*, kỹ thuật được sử dụng phổ biến nhất là *Max Pool*. Ta thực hiện trượt một cửa sổ kích thước  $f \times f$  trên ảnh đầu vào, sau đó lấy ra điểm ảnh có giá trị lớn nhất nằm trong cửa sổ, và trượt cửa sổ qua các ô kế tiếp.

- *Ví dụ:* Ảnh đầu vào có kích thước  $4 \times 4$ , và cửa sổ có kích thước  $2 \times 2$ . Kết quả sẽ là ảnh có kích thước  $2 \times 2$  (Hình 17).

Figure 31: Thực hiện max pool lên ảnh có kích thước  $4 \times 4$ 

*Max Pool* giúp ta giảm được kích thước ảnh, tiết kiệm tài nguyên của máy tính nhưng vẫn giữ được các đặc trưng không gian của ảnh. *Max Pool* còn giúp giảm đi các vấn đề về *Overfitting* do chỉ giữ lại các điểm ảnh có giá trị lớn trong từng cửa sổ giúp khử nhiễu cho ảnh.

Ngoài *Max Pool* ra, còn có rất nhiều các kỹ thuật *Pooling* khác mà ta có thể tìm hiểu thêm. Ví dụ như *Average Pool*, thay vì lấy giá trị lớp nhất trong cửa sổ thì *Average Pool* sẽ lấy giá trị trung bình của các điểm ảnh tương ứng. *Max Pool* và *Average Pool* có những lợi ích khác nhau trong các bài toán cụ thể.

### 6.1.5 CNNs trong bài toán phân lớp

Bằng cách kết hợp các thành phần trên lại với nhau, ta có thể xây dựng được một mô hình CNN học các đặc trưng của ảnh một cách có cấp bậc (*hierarchical*).

- *Ví dụ:* Lớp CONV thứ nhất sẽ bao gồm các filter học các đặc trưng cạnh, điểm, góc,... của ảnh. Lớp CONV thứ hai sẽ học các đặc trưng phức tạp hơn như mắt, tai, mũi,.. Và lớp CONV thứ 3 học các đặc trưng gương mặt khác nhau. (Hình 18)

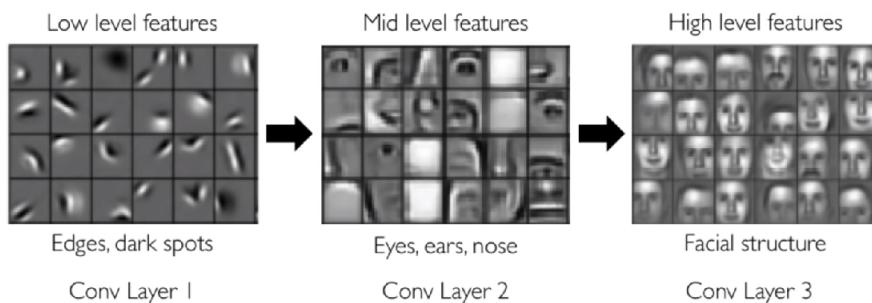


Figure 32: Các đặc trưng phân cấp

Một mô hình CNN trong bài toán phân lớp được chia ra làm 2 phần:

- **Feature Learning**

- Sử dụng các lớp CONV để rút trích ra các đặc trưng quan trọng của ảnh.
- Các đặc trưng được lọc qua hàm kích hoạt (ReLU) để giới hạn miền giá trị của các pixel và giúp giải quyết vấn đề phi tuyến của ảnh (Dữ liệu phân lớp trong thực tế không tuyến tính).
- Giảm kích thước của ảnh nhưng vẫn giữ lại được các đặc trưng không gian của ảnh bằng các lớp POOL (Pooling layers).

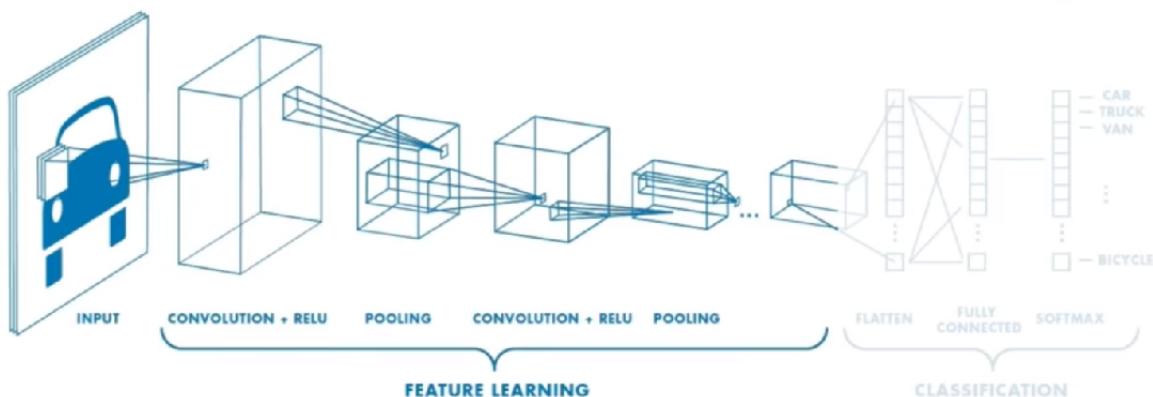


Figure 33: Học đặc trưng trong CNNs

- **Classification**

- Sử dụng các đặc trưng được rút trích từ các lớp CONV và POOL làm đầu vào cho mô hình phân lớp.
- Fully Connected Neural Network sử dụng các đặc trưng này để phân lớp cho ảnh ban đầu.
- Output của phần này là xác suất ảnh ban đầu thuộc vào các lớp trong bài toán phân lớp (thường sử dụng *Softmax*).

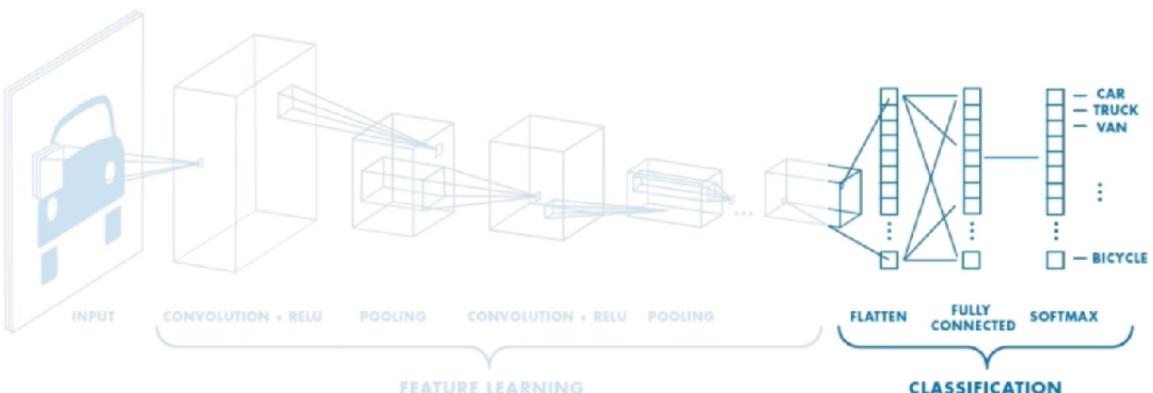


Figure 34: Phân lớp trong CNN

## 6.2 Residual Network (ResNet)

### 6.2.1 Vấn đề độ sâu của các mạng học sâu

Trên lý thuyết, nếu ta có thể tăng số lớp của mạng học sâu, ta có thể đạt được các kết quả tốt hơn, khái quát hơn và giảm thiểu độ lỗi đáng kể. Nhưng trên thực tế, để có thể tối ưu hóa một mạng có nhiều lớp là rất khó.

- *Ví dụ:* Có thể thấy thực nghiệm trong hình bên dưới, khi ta tăng số lớp của mạng học sâu lên, độ lỗi trên tập huấn luyện cũng tăng theo. Vấn đề mà ta gặp phải ở đây là *Vanishing/Exploding Gradients*, có thể hiểu là sự thiếu hiệu quả của việc học đối với các lớp gần phía đầu của mạng trong quá trình lan truyền ngược (*Backpropagation*).

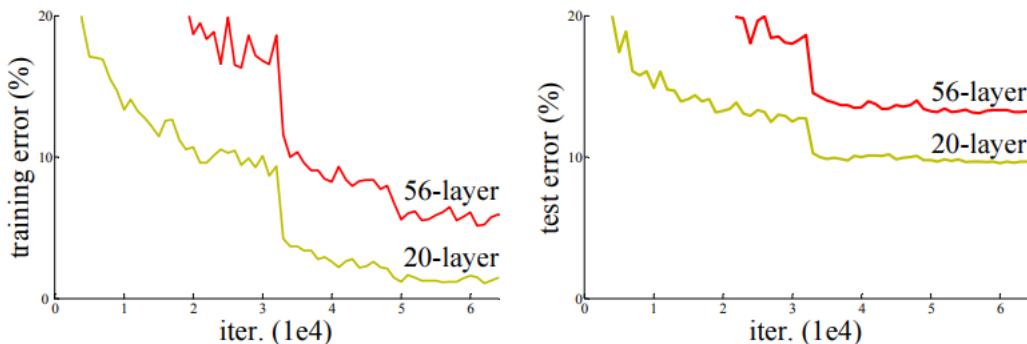


Figure 35: Đồ thị độ lỗi trên tập huấn luyện (trái) và tập kiểm thử (phải) với mô hình CIFAR-10 (20 lớp và 56 lớp)

### 6.2.2 Skip Connection

Khi khởi tạo một mạng học sâu, chúng ta thường khởi tạo các số có giá trị bằng 0 hoặc gần bằng 0 (Gieo ngẫu nhiên từ phân phối chuẩn tắc) hoặc đôi khi ta còn thực hiện các kỹ thuật chuẩn hóa tham số ( $L_2$  normalisation) khiến cho quá trình học ban đầu rất chậm và thậm chí còn chậm hơn nếu mạng có nhiều lớp.

Để giải quyết vấn đề này, ý tưởng của ResNet rất đơn giản, giả sử ta có một mô hình mạng học sâu gồm 2 lớp ẩn, đầu vào ( $x$ ) ngoài đi theo con đường chính của mạng qua 2 lớp ẩn, nó sẽ được truyền theo một con đường khác và được cộng vào đầu ra của lớp thứ hai trước khi đưa vào hàm kích hoạt (*ReLU*).

Có thể thấy khi  $\mathcal{F}(x) \approx 0$  thì  $\mathcal{H}(x) = \mathcal{F}(x) + x \approx x$ . Nhờ vậy mà bảo toàn được các giá trị của  $x$  đến các lớp sau của mạng. Một tổ hợp mạng như vậy được gọi là *Residual Block*.

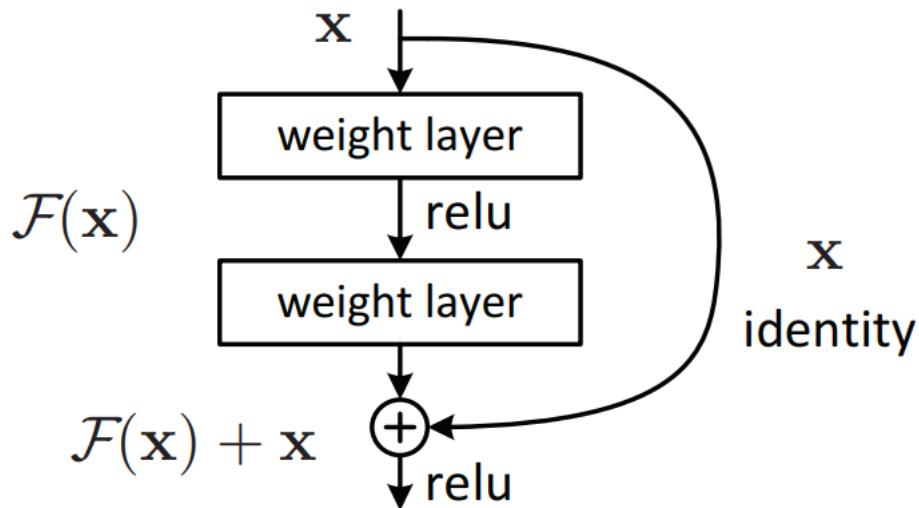


Figure 36: Residual Block

### 6.2.3 Residual Network

Bằng cách nối các *Residual block* lại với nhau, ta được một mạng *Residual Network*.

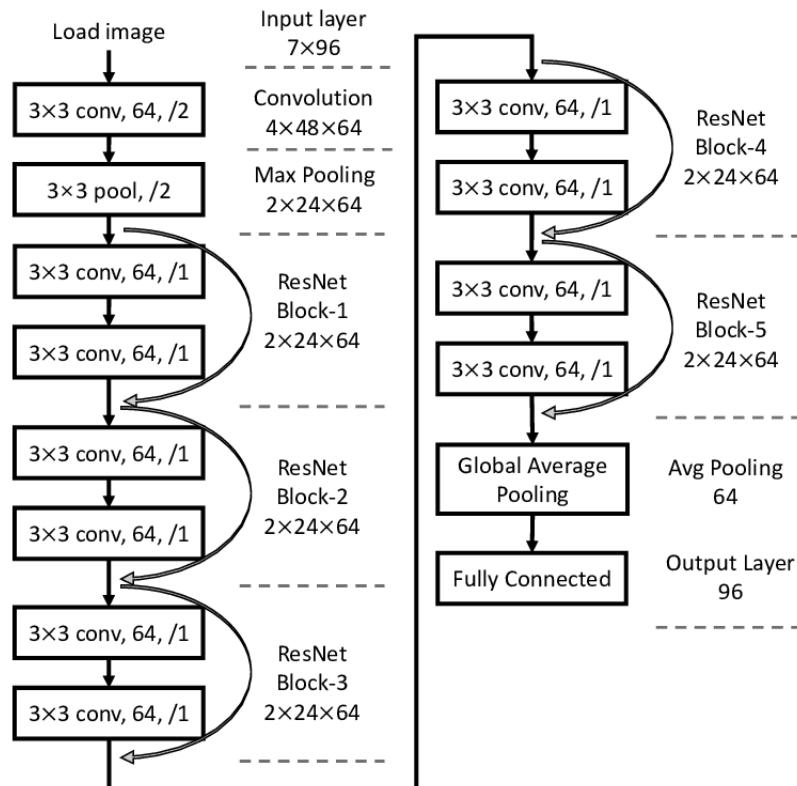


Figure 37: Residual Network

Tác giả đã thực nghiệm các so sánh trên mô hình ImageNet khi tăng số lớp từ 18 lớp đến 34

lớp. Có thể thấy khi tăng số lớp lên thì độ lỗi của mô hình không sử dụng ResNet tăng lên, trong khi đó ResNet khi tăng số lớp thì độ lỗi được giảm đi.

	plain	ResNet
18 layers	27.94	27.88
34 layers	28.54	<b>25.03</b>

Figure 38: Top-1 error(%, 10-crop testing) trên tập validation

### 6.2.4 Bottleneck Block

Để có thể xây dựng các mạng ResNet có độ sâu lên đến hàng trăm lớp, ta cần có 1 cấu trúc tiết kiệm tài nguyên hơn so với các *Residual block*. Mỗi *Bottleneck block* được hình thành từ 3 lớp CONV thay vì 2 như *Residual block* (Hình 25). Các lớp CONV lần lượt là  $1 \times 1$ ,  $3 \times 3$  và  $1 \times 1$ . Lớp CONV  $1 \times 1$  đầu tiên có chức năng giảm chiều của ảnh, từ đó mà lớp CONV  $3 \times 3$  có thể thực hiện phép tích chập với dữ liệu có chiều nhỏ hơn, giảm số lượng các phép tính toán. Và sau đó lớp CONV  $1 \times 1$  cuối cùng khôi phục lại chiều ban đầu của ảnh.

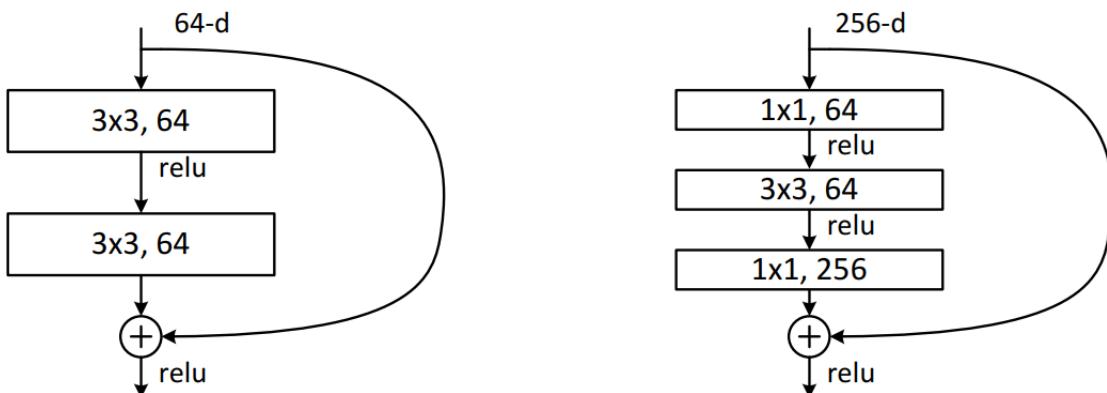


Figure 39: Residual block ( $56 \times 56$  feature maps) dùng trong ResNet-34 (trái). Bottleneck block dùng trong mạng ResNet-50/101/152.

### 6.2.5 Overfitting

Tuy ResNet đã giải quyết được vấn đề *Vanishing/Exploding Gradients*. Tuy nhiên, ResNet cũng có những hạn chế của nó. Tác giả ResNet đã thực hiện các thí nghiệm tăng số lớp của ResNet (Hình 26). Kết quả cho thấy độ lỗi trên tập test của ResNet giảm dần khi tăng số lớp lên đến 110, nhưng lại tăng lên khi tăng lên 1202 lớp.

method	error (%)		
Maxout [10]		9.38	
NIN [25]		8.81	
DSN [24]		8.22	
	# layers	# params	
FitNet [35]	19	2.5M	8.39
Highway [42, 43]	19	2.3M	7.54 ( $7.72 \pm 0.16$ )
Highway [42, 43]	32	1.25M	8.80
ResNet	20	0.27M	8.75
ResNet	32	0.46M	7.51
ResNet	44	0.66M	7.17
ResNet	56	0.85M	6.97
ResNet	110	1.7M	<b>6.43</b> ( $6.61 \pm 0.16$ )
ResNet	1202	19.4M	7.93

Figure 40: Độ lỗi mô hình CIFAR-10 trên tập test. Các mô hình đều được áp dụng tăng cường dữ liệu *data augmentation*. Với ResNet-110, được chạy 5 lần và đưa ra kết quả "best(mean±std)"

Có thể thấy trong hình bên dưới, độ lỗi trên tập huấn luyện của ResNet khi tăng số lớp lên ngày càng giảm, trong khi độ lỗi trên tập kiểm thử lại tăng lên. Từ đó, ta có thể khẳng định rằng, khi tăng số lớp của ResNet lên 1202 thì mô hình đã bị *Overfitting*.

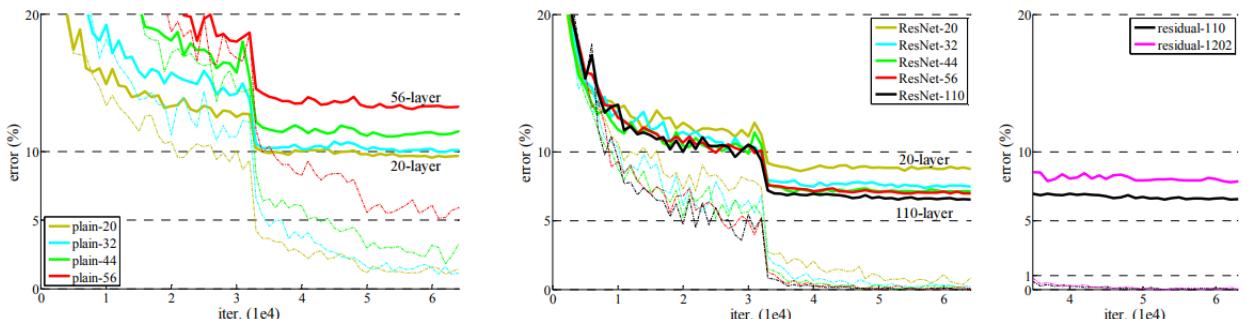


Figure 41: Huấn luyện trên CIFAR-10. Các đường nét đứt thể hiện độ lỗi trên tập huấn luyện, còn các đường in đậm thể hiện độ lỗi trên. Plain Network (trái), Residual Network(giữa), ResNet so sánh giữa 110 lớp và 1202 lớp (phải).

### 6.3 Additive Angular Margin Loss (ArcFace)

#### 6.3.1 Hạn chế của hàm lỗi Softmax

Softmax cho ta biết xác suất của 1 mẫu thuộc về các lớp là bao nhiêu phần trăm.

$$L_1 = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^N e^{W_j^T x_i + b_j}} \quad (7)$$

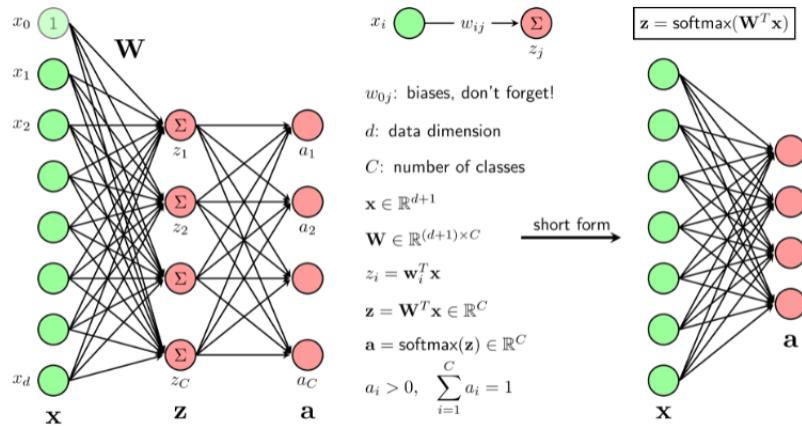


Figure 42: Mô Hình Mạng Neural Softmax

Nhược điểm của hàm softmax là không tối ưu hóa độ tương đồng giữa các embedding trong cùng 1 lớp và độ khác biệt giữa các lớp khác nhau. Vì thế làm giảm hiệu xuất của hệ thống phân lớp khi phuơng sai của lớp tăng lên (các tư thế gương mặt, biểu cảm khác nhau, sự lão hóa,...). Hình dưới cho ta thấy khi phải phân lớp cho các phần tử ở gần biên giữa 2 lớp *class 1* và *class 2* sẽ khó khăn hơn và dễ xảy ra sai sót.

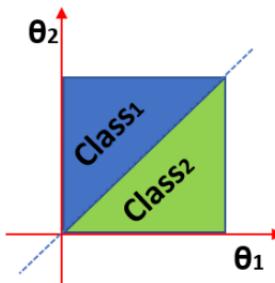


Figure 43: Hạn chế của Softmax

### 6.3.2 Center Loss

Ý tưởng của *Center loss* là tìm ra tâm của các lớp. Từ đó mà tối ưu khoảng cách Euclian từ các điểm dữ liệu đến tâm của lớp chứa điểm dữ liệu đó. Đồng thời cũng tối ưu khoảng cách giữa các lớp càng xa nhau.

$$\mathcal{L}_C = \frac{1}{2} \sum_{i=1}^N \|x_i - c_{y_i}\|_2^2 \quad (8)$$

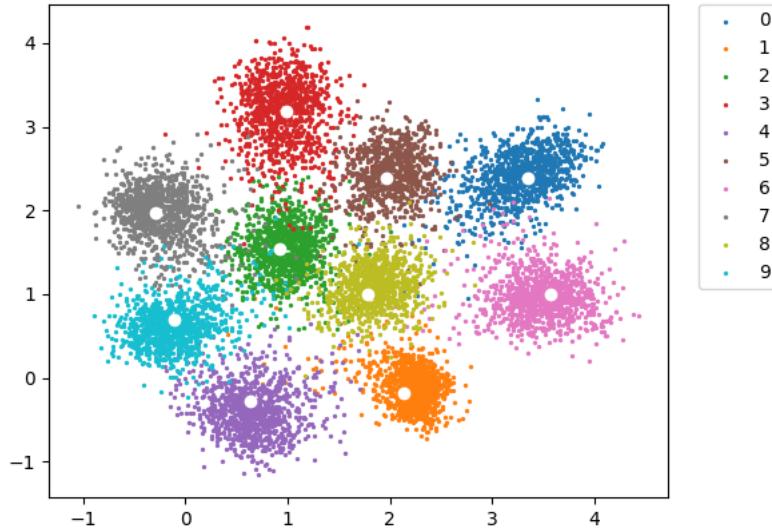


Figure 44: Tâm của các lớp khi đã sử dụng center loss

Nhược điểm của *Center loss* nằm ở việc cập nhập các tâm của lớp trong quá trình huấn luyện cực kì khó khi số lượng các lớp khuôn mặt cần phải huấn luyện ngày càng nhiều lên trong thực tế.

### 6.3.3 Additive Angular Margin Loss (ArcFace)

*ArcFace* hay các độ lỗi khác sử dụng góc giữa các điểm dữ liệu và các vector tham số trước đó như *SphereFace* hay *textitCosFace* đã đưa ra ý tưởng sử dụng các vector tham số làm tâm của các lớp và từ đó tối ưu "khoảng cách" giữa các điểm dữ liệu đến tâm của nó. Ta có thể hình dung được rõ hơn qua công thức sau đây:

$$W_j^T x_i = \|W_j\| \cdot \|x_i\| \cos \theta_j \quad (9)$$

Với  $W_j$  là vector tham số tương ứng với lớp thứ  $j$ ,  $x_i$  là điểm dữ liệu và  $\theta_j$  là giữa  $W_j$  và  $x_i$ .

Ta thực hiện chuẩn hóa ( $L_2$  normalisation) lên các vector  $W_j$  và  $x_i$ . Và nhân các  $x_i$  với 1 lượng  $s$ . Bằng cách chuẩn hóa như trên, ta sẽ có  $\|W_j\| = 1$  và  $\|x_i\| = s$ , nhờ đó mà dự đoán khi phân lớp chỉ phụ thuộc vào  $\theta_j$ . Công thức (7) được đổi thành:

$$L_2 = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{s \cos \theta_{y_i}}}{e^{s \cos \theta_{y_i}} + \sum_{j=1, j \neq y_i}^N e^{s \cos \theta_j}} \quad (10)$$

Từ (10), ta có thể hình dung ra các embedding sẽ tập trung xung quanh các tâm của lớp và phân bố trên 1 mặt siêu cầu (*hypersphere*). Sau đó, ta cộng một siêu tham số  $m$  (*Additive Angular Margin*) vào góc giữa  $x_i$  và  $W_j$  để có thể tối ưu tốt hơn các khoảng cách giữa các embedding. Minh họa bằng hình 45, ta thấy khi sử dụng hàm *Softmax loss* để phân lớp, đường

ranh giới giữa các lớp rất mờ, khiến cho việc phân lớp cho các điểm gần ranh giới này dễ xảy ra sai sót. Còn đối với *Additive angular margin loss* bằng cách sử dụng margin để ngăn cách giữa các lớp, Các điểm dữ liệu co cụm lại gần tâm của nó để lại một khoảng cách xa giữa các lớp từ đó giúp quá trình phân lớp ít lỗi hơn.

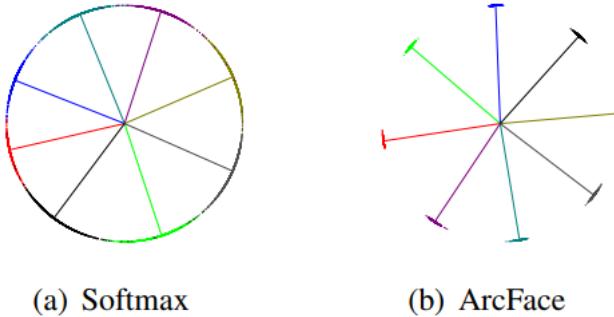


Figure 45: So sánh giữa softmax và arcface

Có thể có cái nhìn tổng quát hơn về khác biệt của các độ lỗi bằng hình bên dưới, có thể thấy các phương pháp *SphereFace* và *CosFace* có các đường biên quyết định (*decision boundary*) có dạng phi tuyến, trong khi *ArcFace* lại cho kết quả tuyến tính giúp phân lớp dễ dàng hơn.

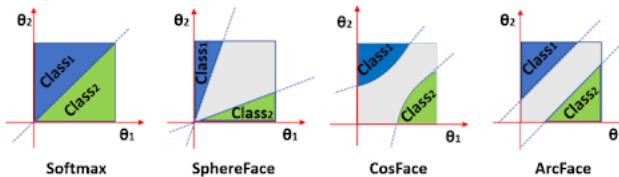


Figure 46: So sánh giữa các độ lỗi

Toàn bộ quá trình phân lớp các embedding bằng *ArcFace* có thể minh họa bằng hình dưới.

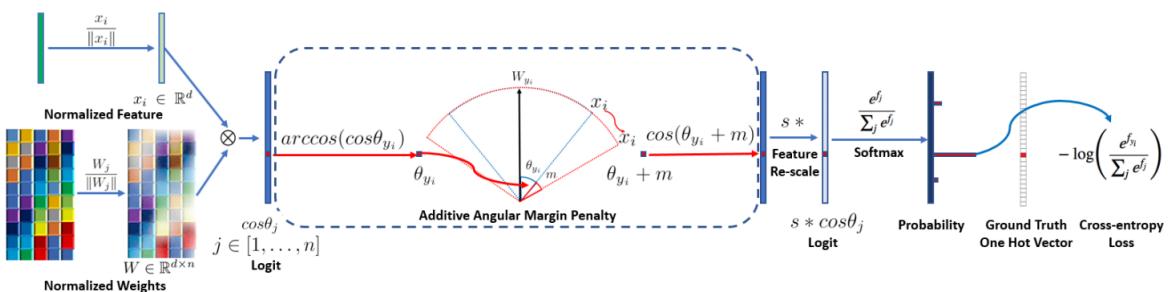


Figure 47: Huấn luyện mô hình CNN sử dụng độ lỗi ArcFace

## 6.4 Evaluation & Demo

Cuối cùng nhóm em xin được trình bày demo dữ dụng ResNet-50 + ArcFace [13], để giải quyết bài toán Face Verification. Model được huấn luyện trên tập dữ liệu MS-CELEB-1M, và được kiểm thử trên các tập dữ liệu LFW, AgeDB-30, CFP-FP. kết quả được trình bày sau đây:

Backbone	Head	Loss	Center Crop	LFW	AgDB-30	CFP-FP
ResNet-50	ArcFace	Sofmax	False	99.35	95.03	90.36
MobileNetV2	ArcFace	Sofmax	False	98.67	90.87	88.51

Table 3: Kết quả accuracy trên tập test của mô hình sử dụng ResNet-40 so sánh với MobileNetV2

### 6.4.1 Demo

Demo chúng em trình bày trong file *demo.ipynb* và thực hiện các chức năng sau:

1. Xuất ra embedding sau khi chạy qua mạng.
2. So sánh embedding giữa cá đối tượng (ảnh được tải từ google) và cho kết quả như sau:

Image 1	Image 2	Is same
steve.jpg	BruceLee1.jpg	False
BruceLee1.jpg	BruceLee2.jpg	True
BruceLee1.jpg	TranQuocKhon.jpg	False
BruceLee2.jpg	BruceLee3.jpg	True
BruceLee3.jpg	TranQuocKhon.jpg	False

Table 4: Kết quả verification trên các ảnh trong demo

## 6.5 Polynomial Neural Network( $\Pi$ -Net)

### 6.5.1 Xấp xỉ bằng hàm đa thức

Nhóm tác giả của  $\Pi$ -Net đã chứng minh được rằng, *mọi hàm số đều có thể xấp xỉ bằng một hàm đa thức*. Từ đó họ xây dựng một cấu trúc mạng không sử dụng các hàm kích hoạt phi tuyến, giúp tăng tốc độ học và độ chính xác của mạng. Một đa thức  $G : \mathbb{R}^d \rightarrow \mathbb{R}^o$  bậc  $N \in \mathbb{N}$  sẽ có dạng như sau:

$$\mathbf{x} = G(\mathbf{z}) = \sum_{n=1}^N \left( \mathbf{W}^{[n]} \times_2 \mathbf{b}_{[N+1-n]} \prod_{j=3}^{n+2} \times_j \mathbf{z} \right) + \boldsymbol{\beta} \quad (11)$$

Với  $\mathbf{z} \in \mathbb{R}^d$ ,  $\mathbf{x} \in \mathbb{R}^o$ ,  $\{\mathbf{W}^{[n]} \in \mathbb{R}^{o \times w \times \Pi_m^n \times m^d}\}_{n=1}^N$ ,  $\{\mathbf{b}_{[n]} \in \mathbb{R}^w\}_{n=1}^N$  và  $\boldsymbol{\beta} \in \mathbb{R}^o$ .

Nhờ vào phép phân tích CP (CP decomposition), các tensor  $\mathbf{W}^{[n]}$  bậc cao có thể phân tích thành các tensor có bậc thấp hơn, giúp tiết kiệm được lượng tham số.

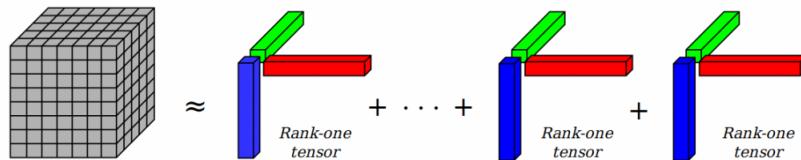


Figure 48: CP Decomposition trên tensor 3 chiều

Trong các kĩ thuật phân tích tensor mà nhóm tác giả đã đề cập thì *NCP-skip* (*Nested coupled CP decomposition with skip*) là một kĩ thuật khá thú vị vì nó ứng dụng các nút *skip connection* giống với ý tưởng của *ResNet*.

$$\mathbf{x}_n = (\mathbf{A}_{[n]}^T \mathbf{z}) * (\mathbf{S}_{[n]}^T \mathbf{x}_{n-1} + \mathbf{B}_{[n]}^T \mathbf{b}_{[n]}) + \mathbf{V}_{[n]} \mathbf{x}_{n-1} \quad (12)$$

Cho  $n = 2, \dots, N$  với  $\mathbf{x}_1 = (\mathbf{A}_{[1]}^T \mathbf{z}) + (\mathbf{B}_{[1]}^T \mathbf{b}_{[1]})$  và  $\mathbf{x} = \mathbf{C} \mathbf{x}_N + \boldsymbol{\beta}$ . Các tham số  $\mathbf{C} \in \mathbb{R}^{o \times k}$ ,  $\mathbf{A}_{[n]} \in \mathbb{R}^{d \times k}$ ,  $\mathbf{S}_{[n]} \in \mathbb{R}^{k \times k}$ ,  $\mathbf{B}_{[n]} \in \mathbb{R}^{w \times k}$ ,  $\mathbf{b}_{[n]} \in \mathbb{R}^w$  và  $\mathbf{V}_{[n]} \in \mathbb{R}^{k \times k}$  là các tham số học.

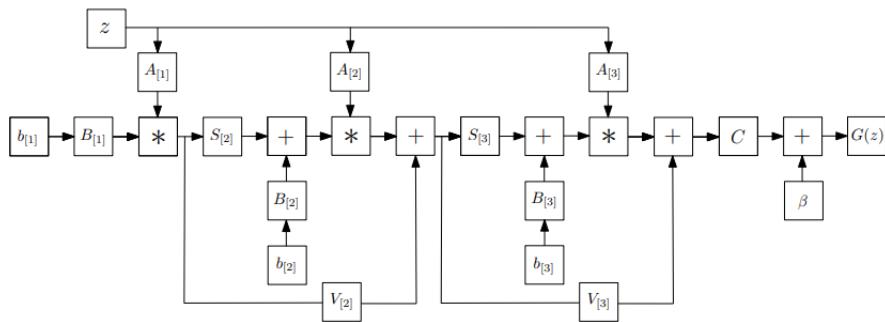


Figure 49: Mạng đa thức bậc 3

### 6.5.2 ProdPoly

Thay vì sử dụng một hàm đa thức bậc cao phức tạp, ta có thể tích các đa thức có bậc thấp (2 hoặc 3) để hình thành lên các đa thức bậc cao hơn. Phép tích được thực hiện bằng cách dùng output của đa thức thứ i làm input cho đa thức thứ (i + 1).

- *Ví dụ:* (Hình 50) Ta sử dụng N mạng đa thức có bậc là 2 nối lại với nhau, ta sẽ được 1 mạng đa thức có bậc là  $2^N$ . Tổng quát hơn, nếu ta có N mạng đa thức có bậc là B, thì khi nối chúng lại với nhau ta sẽ được một mạng đa thức có bậc là  $B^N$ . Cấu trúc mạng như vậy được gọi là ***Prodpoly***.

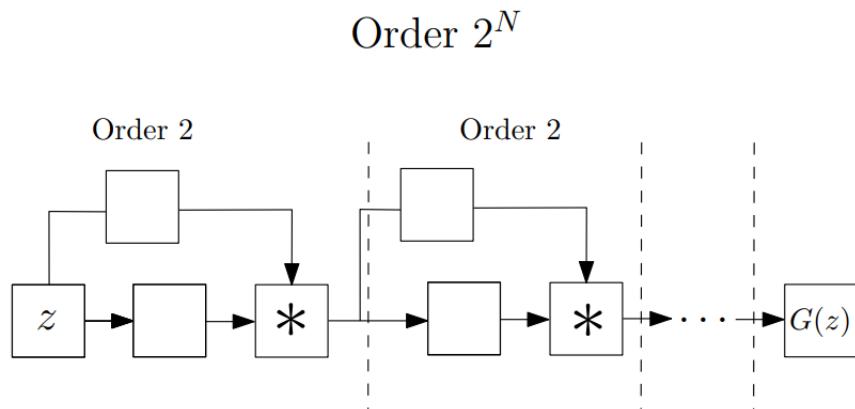


Figure 50: Cấu trúc mạng Prodpoly

Các thí nghiệm đã được thực hiện để kiểm tra độ hiệu quả của Prodpoly so với các phương pháp ResNet trước đó. Có thể thấy Prodpoly tối ưu tốt hơn hẳn so với các mô hình chỉ sử dụng ResNet.

Method	ResNet50	Prodpoly-ResNet50
LFW	$99.733 \pm 0.309$	<b><math>99.833 \pm 0.211</math></b> ( $\uparrow 0.100$ )
CFP-FF	$99.871 \pm 0.135$	<b><math>99.886 \pm 0.178</math></b> ( $\uparrow 0.015$ )
CFP-FP	$98.800 \pm 0.249$	<b><math>98.986 \pm 0.274</math></b> ( $\uparrow 0.186$ )
CPLFW	$92.433 \pm 1.245$	<b><math>93.317 \pm 1.343</math></b> ( $\uparrow 0.884$ )
AgeDB-30	$98.233 \pm 0.655$	<b><math>98.467 \pm 0.623</math></b> ( $\uparrow 0.234$ )
CALFW	$95.917 \pm 1.209$	<b><math>96.233 \pm 1.114</math></b> ( $\uparrow 0.316$ )
RFW-Caucasian	$99.333 \pm 0.307$	<b><math>99.700 \pm 0.100</math></b> ( $\uparrow 0.367$ )
RFW-Indian	$98.567 \pm 0.507$	<b><math>99.300 \pm 0.296</math></b> ( $\uparrow 0.733$ )
RFW-Asian	$98.333 \pm 0.435$	<b><math>98.950 \pm 0.350</math></b> ( $\uparrow 0.617$ )
RFW-African	$98.650 \pm 0.329$	<b><math>99.417 \pm 0.227</math></b> ( $\uparrow 0.767$ )

Figure 51: Bảng so sánh kết quả mô hình ResNet50 và Prodpoly-ResNet50

## 7 Tổng kết

Chúng ta đã cùng nhau đi từ các phương pháp kinh điển đến hiện đại trong bài toán nhận diện gương mặt. Các kỹ thuật mới được phát triển dựa trên các mô hình mạng học sâu ngày càng thể hiện được sức mạnh của mình trong việc giải các bài toán liên quan đến nhận diện. Cấu trúc mạng ResNet được phát hiện vào năm 2015 và đã được phổ biến khắp các mô hình mạng học sâu. ArcFace đạt được các thứ hạng cao trong cuộc thi về nhận dạng.

## 8 Tham khảo

1. Bài 27: Principal Component Analysis (machinelearningcoban)
2. Bài 29: Linear Discriminant Analysis(machinelearningcoban)
3. Bài 19: Support Vector Machine (machinelearningcoban)
4. Bài 21: Kernel Support Vector Machine (machinelearningcoban)
5. LFW Face Dataset
6. AgeDB Dataset
7. CFPW Dataset
8. MIT 6.S191 (2020): Convolutional Neural Networks
9. Convolutional Neural Networks (Course 4 of the Deep Learning Specialization)
10. Residual Network
11. ArcFace: Additive Angular Margin Loss for Deep Face Recognition
12. ArcFace explained video
13. ArcFace for face verification
14. Polynomial Neural Network