ĐẠI HỌC KHOA HỌC TỰ NHIÊN, ĐHQG-HCM
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN KHOA HỌC MÁY TÍNH

NHẬN DẠNG
BÁO CÁO ĐỒ ÁN CUỐI KỲ

Giảng viên lý thuyết
PGS. TS. Lê Hoàng Thái

Giảng viên hướng dẫn

Lê Thanh Phong, Nguyễn Ngọc Thảo

Sinh viên thực hiện

Nguyễn Hoàng Đức, Lê Nhựt Nam, Nguyễn Viết Dũng

Tháng 7 năm 2021

Lời cảm ơn

Trong quá trình thực hiện đồ án này, chúng em đã nhận được rất nhiều sự giúp đỡ cũng như hỗ trợ từ các thầy cô Trường Đại học Khoa học Tự nhiên, ĐHQG-HCM và các bạn bè trong lớp Nhận dạng. Chúng em xin bày tỏ lòng cảm ơn chân thành đến mọi người vì đã giúp đỡ hướng dẫn, chỉ bảo rất tận tình.

Đặc biệt, chúng em xin bày tỏ lòng biết ơn sâu sắc đến các thầy cô khoa Công nghệ Thông tin, cụ thể hơn là thầy Hoàng Thái, thầy Thanh Phong và cô Ngọc Thảo, dẫn đã giảng dạy rất nhiệt, cung cấp nhiều slides, tài nguyên học tập cần thiết, tạo điều kiện tốt nhất để chúng em có thể hoàn thành được đồ án này.

Trong lúc viết báo cáo này, chúng em không thể tránh khỏi nhiều thiếu sót, hy vọng mong nhận được góp ý từ thầy để chúng em tiếp tục hoàn thiện hơn đối với đồ án này, cũng như rút kinh nghiệm cho những đồ án, những báo cáo kế tiếp.

Đại học Khoa học Tự nhiên, ĐHQG-HCM.
Nguyễn Hoàng Đức, Lê Nhựt Nam, Nguyễn Viết Dũng
Tháng 5 năm 2021

Mục lục

A. Câu hỏi lý thuyết

A.1 Kiến thức về Local Binary Patterns (LBP)

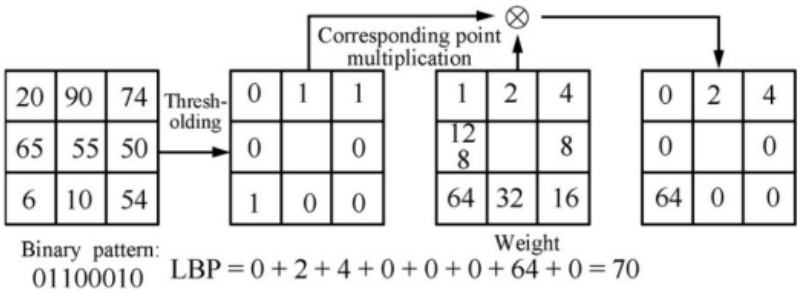
a. Trình bày đặc trưng mẫu nhị phân cục bộ $\mathbf{LBP_{P,R}}$

(1) Phương pháp xác định giá trị

Phương pháp xác định giá trị LBP sơ khai: Được tính toán bằng cách tại mỗi điểm ảnh, xét 8 điểm xung quanh nó

- Bước 01: Các điểm xung quanh có giá trị nhỏ hơn điểm đang xét sẽ được đánh dấu là 0, lớn hơn điểm đang xét sẽ được đánh dấu là 1.
- Bước 02: Sau đó, các giá trị sau khi tính toán phân ngưỡng ở trên sẽ được nhân với ma trận trọng số và được sử dụng để tính giá trị LBP của điểm đang xét.

$$\begin{bmatrix} 1 & 2 & 4 \\ 128 & & 8 \\ 64 & 32 & 16 \end{bmatrix}$$



Hình 1: Mô tả quá trình thực hiện LPB

Phương pháp xác định giá trị LBP cải tiến: Xét các điểm thuộc đường tròn với tâm là điểm đang xét. Ta gọi (P, R) là vùng lân cận gồm P điểm trên một đường tròn bán kinh R

$$T = t(g_c, g_0, ..., g_{p-1}) \tag{1}$$

Trong đó:

- g_c và $(g_0, ..., g_{p-1})$ là giá trị trên ảnh xám của điểm trung tâm và các điểm trên đường tròn bán kính R

Ta có thể xấp xỉ công thức trên bằng cách lấy từng điểm trên đường tròn bán kinh R trừ đi giá trị trung tâm g_c do các giá trị chỉ thể hiện cường độ sáng của điểm ảnh.

$$T \approx t(g_0 - g_c, g_1 - g_c, ..., g_{p-1} - g_c) \tag{2}$$

Để không bị ảnh hưởng bởi các giá trị của các điểm ảnh, chuẩn hoá công thức trên bằng việc xét dấu

$$T \approx t(sign(g_0 - g_c), sign(g_1 - g_c), ..., sign(g_{p-1} - g_c)) \tag{3}$$

Trong đó:

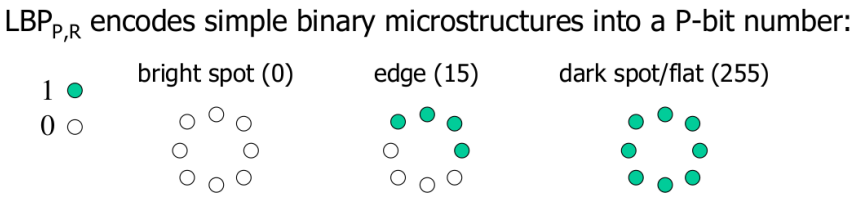
$$sign(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$$

Giá trị LBP được tính như sau:

$$LBP_{P,R} = \sum_{i=0}^{p-1} sign(g_i - g_c) \times 2^i \tag{4}$$

(2) Ưu điểm

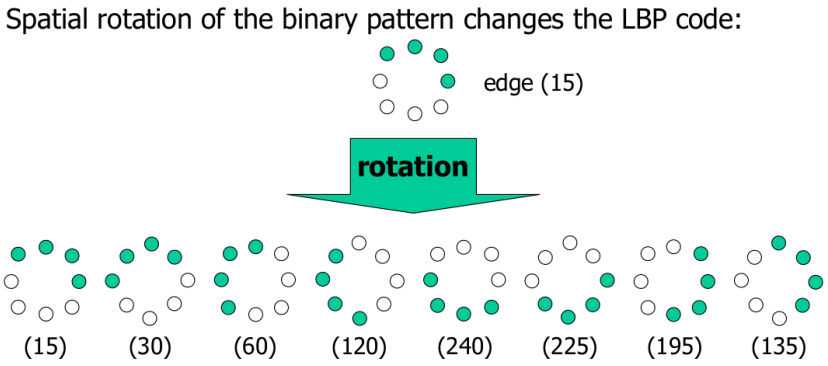
- Bất biến đối với bất kỳ phép biến đổi đơn điệu nào trên ảnh xám (grayscale) nên hiệu quả cho việc làm giảm tỷ lệ từ chối sai (FRR) cho mỗi đối tượng, khi đối tượng đó được chụp tại những điều kiện sáng/ tối khác nhau
- Lượng tử hoá vector là không cần thiết
- Tính toán một cách đơn giản



Hình 2: Mô tả quá trình thực hiện LPB

(3) Nhược điểm

Chưa quan tâm đến phép quay, đặc trưng LBP áp dụng trên hình ảnh xoay của cùng một đối tượng sẽ tạo ra các giá trị khác nhau.



Hình 3: Ảnh hưởng của phép xoay lên đặc trưng LBP

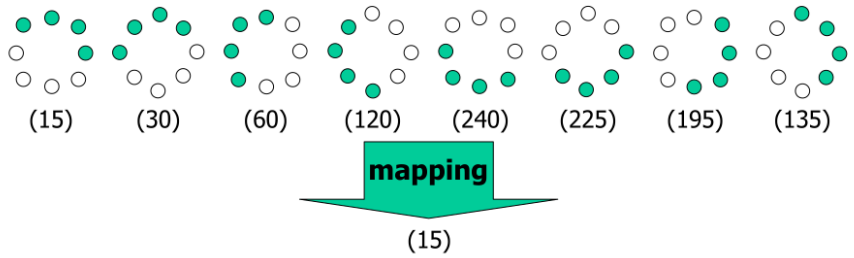
Có vấn đề khi số điểm lân cận tăng quá cao, giá trị LBP sẽ tăng lên rất cao

b. Sự khác biệt giữa đặc trưng $\mathbf{LBP_{P,R}}$ so với đặc trưng $\mathbf{LBP_{P,R^{ri}}}$

Đặc trưng $LBP_{(P,R)^{ri}}$ khắc phục nhược điểm của đặc trưng $LBP_{(P,R)}$: bất biến với phép xoay, chọ giá trị nhỏ nhất để đại diện cho tất cả các ảnh xoay (mỗi ảnh được xoay theo hướng của vòng tròn lượng giác một góc xác định cho trước)

Mã nhị phân phát sinh bởi LBP được xoay trong vòng tròn, theo hướng bên phải, một cách liên tục để đạt được giá trị nhỏ hơn cho mã nhị phân.

Ví dụ: 11101000 được dịch bit thành 01110100, sau đó thành 00111010, và cuối cùng là 00011101 sẽ là giá trị nhỏ nhất

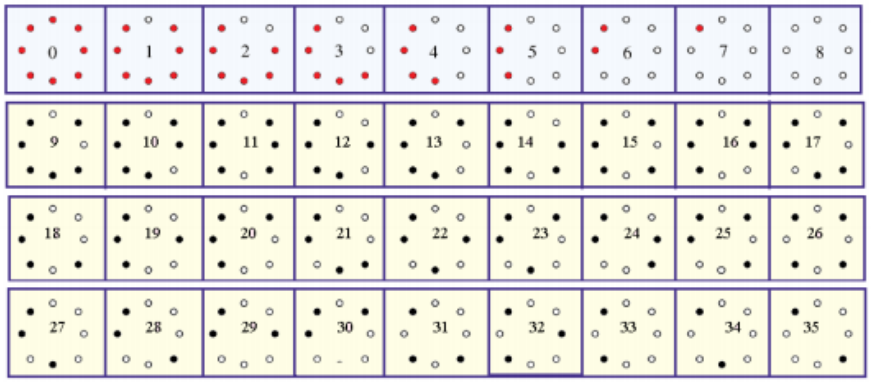


Hình 4: Đặc trưng LBP PR ri

Để đạt được bất biến với phép quay, một hàm bất biến xoay của LBP được định nghĩa bằng

$$LBP_{(P,R)^{ri}} = min(ROR(LBP_{(P,R)^{ri}}, i)i = 0, 1, ..., P-1) \tag{5}$$

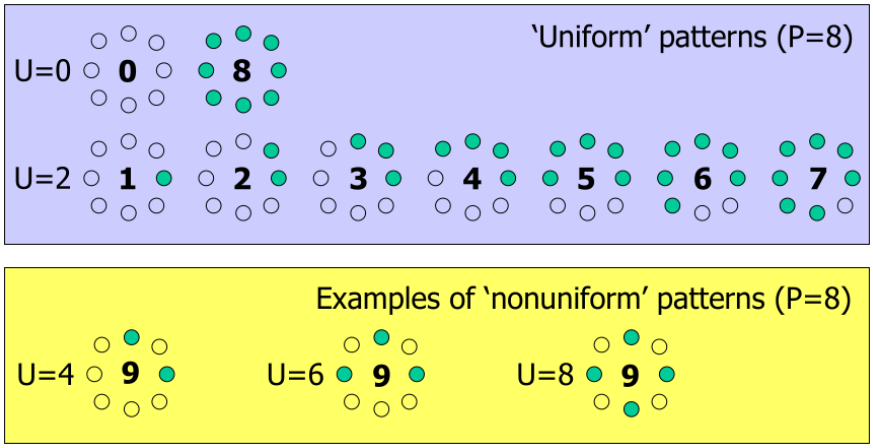
Trong đó, ROR là hàm xoay có tác dụng thay đổi chuỗi nhị phân thu được từ các điểm mẫu lần lượt. Sau khi tính toán hết giá trị LBP cho mỗi chuỗi, giá trị nhỏ nhất sẽ được chọn để biểu diễn mẫu kết cấu đó. Vì thế, thay vì phải xử lý 256 mẫu khác nhau cho $LBP_{8,1}$, ta chỉ cần 36 mẫu khác nhau



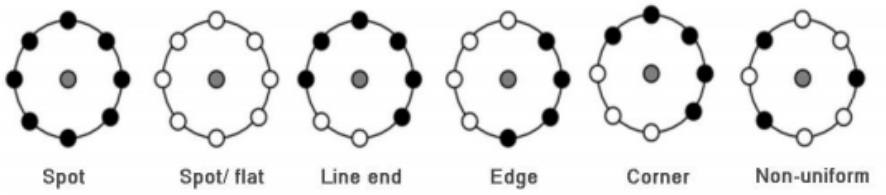
Hình 5: LBP Rotation Invariance

c. Sự khác biệt giữa đặc trưng $\mathbf{LBP_{P,R^{riu2}}}$ so với đặc trưng $\mathbf{LBP_{P,R^{ri}}}$

Đặc trưng $LBP_{P,R^{riu2}}$ cải tiến so với đặc trưng $LBP_{P,R^{ri}}$ bằng cách sử dụng khái niệm "mẫu đồng phục "Uniform Patterns"



Hình 6: Uniform Patterns



Hình 7: Hình ảnh nguyên thủy được nắm bắt bởi mẫu LBP

Để kiểm tra một mẫu kết cấu có phải là "Uniform Patterns"hay không, ta áp dụng công thức

$$U(LBP_{P,R}) = |sign(g_{p-1} - g_c) - sign(g_0 - g_c)| + \sum_{i=1}^{p-1} |sign(g_i - g_c) - sign(g_{i-1} - g_c)| \quad (6)$$

Khi $U \leq 2$, mẫu kết cấu gọi là "Uniform Patterns", ngược lại là "Non-Uniform Patterns"

Công thức tính "Uniform Patterns"

$$LBP^{riu2}_{P,R} = \begin{cases} \sum_{i=0}^{p-1}(g_i - g_c), \text{ if } U(LBP_{P,R}) \leq 2 \\ P+1, \text{otherwise} \end{cases} \quad (7)$$

Circular Binary Pattern	Bitwise transitions	Uniform pattern
11111111	0	Yes
00001111	1	Yes
11001110	3	No
10101010	8	No
11111101	2	Yes
01001011	6	No
11000000	1	Yes

Hình 8: Ví dụ Uniform và Non-Uniform LBP Patterns

d. Trình bày một biến thể của LBP

Trong phần này, chúng em xin được trình bày một biến thể khá kinh điển của LBP, ILBP hay Improved Local Binary Pattern được trình bày trong bài báo Face Detection Using Improved LBP Under Bayesian Framework bởi Hongliang Jin, Qingshan Liu, Hanqing Lu, Xiaofeng Tong đến từ National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, China được công bố tại hội nghị Third International Conference on Image and Graphics (ICIG'04) năm 2004

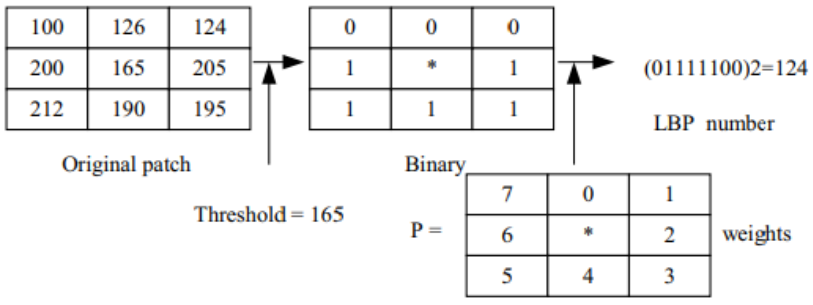
Đặc trưng ILBP là một cải tiến của đặc trưng LBP, xem xét cả dáng cục bộ (local shape) và thông tin kết cấu (texture information) thay vì thông tin xám thô (raw grayscale information) và nó cực kỳ tốt với sự thay đổi ánh sáng (illumination variation).

(1) Phương pháp xác định giá trị

Với một tập P lân cận và một đường tròn bán kính R, ta dễ dàng tính được hiệu số giữa pixel trung tâm (central pixel) g_c và lân cận của nó $\{g_0, ..., g_{p-1}\}$

$$LBP_{P,R} = \sum_{i=0}^{P-1} s(g_i - g_c)^2$$

$$s(x) = \begin{cases} 1, x > 0 \\ 0, x \leq 0 \end{cases}$$



Hình 9: Mô tả quá trình thực hiện LPB

Đặc trưng Improved Local Binary Pattern từ ý tưởng ánh xạ giá trị điểm ảnh sang một khoảng

giá trị mới, có thể được khai triển như sau:

$$LBP_{P,R} = \sum_{i=0}^{P-1} s(g_i - m)2^i + s(g_c - m)2^P$$

$$s(x) = \begin{cases} 1, x > 0 \\ 0, x \leq 0 \end{cases}$$

$$m = \frac{1}{P+1}\left(\sum_{i=0}^{P-1} g_i + g_c\right)$$

7	0	1
6	8	2
5	4	3

ILBP81

X	0	X
3	4	1
X	2	X

ILBP41

Figure 2. Mapping weights for $ILBP_{8,1}$ and $ILBP_{4,1}$ ("X" stands for arbitrary pixel value)

Hình 10: Ánh xạ trọng số cho ILBP 8,1 và ILBP 4,1

X	1	X
1	20	20
X	20	X

a. Original patch

X	0	X
0	*	0
X	0	X

b. Original LBP

X	0	X
0	1	1
X	1	X

c. improved LBP

Figure 3. Comparison of Original LBP and ILBP

Hình 11: So sánh giữa đặc trưng LBP và ILBP

(2) Ưu điểm và nhược điểm, trên cơ sở so sánh LBP

Về ưu điểm của ILBP:

- Giữa được cấu trúc cục bộ trong một số hoàn cảnh nhất định. Khi sử dụng LBP truyền thống, ta chỉ có thể sử dụng $2^8 = 256$ trong tổng số 511 patterns từ một patch 3×3
- ILBP xem xét nhiều hơn pixel trung tâm, trong đa số trường hợp, pixel trung tâm có nhiều thông tin hơn những lân cận của nó. Vì thế, trong ILBP, pixel trung tâm được mang trọng số lớn nhất

Về nhược điểm ILBP

- Những hình ảnh có chứa nguồn sáng cực mạnh khi sử dụng phương pháp này sẽ có kết quả không cao vì những thông tin cấu trúc đã bị mất mát

A.2 Kiến thức về Principal Component Analysis (PCA) và Linear Discriminant Analysis (LDA)

a. Các bước thực hiện của giải thuật phân tích thành phần chính PCA

Tư tưởng chính của PCA là đem chiếu dữ liệu sang một miền mới có số chiều d nhỏ hơn số chiều ở miền gốc là D, sao cho phương sai của dữ liệu sau khi được chiếu là lớn nhất có thể.
Thuật toán PCA:
Input: $D = x_1, x_2, ..., x_n, x_i \in R^D$
Output: W
Bước 1. Xây dựng vector trung bình μ

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Bước 2. Xây dựng ma trận hiệp phương sai S

$$S = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_i)(x_i - \mu_i)^T$$

Bước 3. Phân rã ma trận hiệp phương sai thành những cặp vector riêng và giá trị riêng

$$\{w_1, w_2, ..., w_D\}$$

và

$$\lambda_1, \lambda_2, ..., \lambda_D$$

Bước 4. Sắp xếp các giá trị riêng theo thứ tự giảm dần tương ứng với các vectors riêng

$$\{w_1, w_2, ..., w_D\}$$

và

$$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_D$$

Bước 5. Chọn k vector riêng mà tương ứng với k giá trị riêng lớn nhất, với k là số chiều đặc trưng mới ($k \leq D$). Ở đây mình sẽ có một cách chọn k sao cho hợp lý với cách dựa vào threshold

$$\{w_1, w_2, ..., w_k\}$$

và

$$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_k$$

Bước 6. Xây dưng ma trận hình chiếu W từ k vector riêng

$$W = [w_1, w_2, ..., w_k]^T$$

Bước 7: Ta có một phép biến đổi tuyến tính (linear transformation) $R^N \to R^k$ thực hiện giảm chiều (dimensionality reduction)

$$\begin{bmatrix} b_1 \\ b_2 \\ ... \\ b_k \end{bmatrix} = \begin{bmatrix} w_1^T \\ w_2^T \\ ... \\ w_k^T \end{bmatrix} (x - \bar{x}) = W^T(x - \bar{x})$$

Để chọn được giá trị k, chúng ta có thể sử dụng tiêu chí sau:

$$\frac{\sum_{i=1}^{k}\lambda_i}{\sum_{i=1}^{N}\lambda_i} > \text{threshold}$$

Trong đó: threshold là một người mà chúng ta muốn, có thể là 0.9, 0.95
Ví dụ: Cho dữ liệu

$$X = \begin{bmatrix} 7 & 4 & 6 & 8 & 8 & 7 & 5 & 9 & 7 & 8 \\ 4 & 1 & 3 & 6 & 5 & 2 & 3 & 5 & 4 & 2 \\ 3 & 8 & 5 & 1 & 7 & 9 & 3 & 8 & 5 & 2 \end{bmatrix}$$

Bước 01: Tính toán vector trung bình

$$\text{mean\_vector} = [6.9, 3.5, 5.1]$$

Bước 02: Xây dựng ma trận hiệp phương sai

$$C = \begin{bmatrix} 2.32 & 1.61 & -0.43 \\ 1.61 & 2.5 & -1.278 \\ -0.43 & -1.278 & 7.878 \end{bmatrix}$$

Bước 03: Phân rã ma trận hiệp phương sai thành những cặp vector riêng và giá trị riêng

$$w_1 = [-0.7012, 0.7075, 0.0841], \lambda_1 = 0.7499$$

$$w_2 = [0.699, 0.6609, 0.2731], \lambda_2 = 3.6761$$

$$w_3 = [-0.1376, -0.2505, 0.9583], \lambda_3 = 8.2739$$

Bước 04: Sắp xếp các giá trị riêng theo thứ tự giảm dần tương ứng với các vectors riêng

$$w_3 = [-0.1376, -0.2505, 0.9583], \lambda_3 = 8.2739$$

$$w_2 = [0.699, 0.6609, 0.2731], \lambda_2 = 3.6761$$

$$w_1 = [-0.7012, 0.7075, 0.0841], \lambda_1 = 0.7499$$

Bước 05: Chọn $k = 2$ vector riêng mà tương ứng với k giá trị riêng lớn nhất, với k là số chiều đặc trưng mới ($k \leq D$). Ở đây mình sẽ có một cách chọn k sao cho hợp lý với cách dựa vào threshold

$$w_3 = [-0.1376, -0.2505, 0.9583], \lambda_3 = 8.2739$$

$$w_2 = [0.699, 0.6609, 0.2731], \lambda_2 = 3.6761$$

Bước 06: Xây dưng ma trận hình chiếu W từ $k = 2$ vector riêng

$$W = \begin{bmatrix} -0.1376 & 0.699 \\ -0.2505 & 0.6609 \\ 0.9583 & 0.2731 \end{bmatrix}$$

Bước 7: Giảm chiều dữ liệu

$$\begin{bmatrix} b_1 \\ b_2 \\ ... \\ b_k \end{bmatrix} = \begin{bmatrix} w_1^T \\ w_2^T \\ ... \\ w_k^T \end{bmatrix} (x - \bar{x}) = W^T (X - \bar{X})$$

$$\begin{bmatrix} -0.1376 & -0.2505 & 0.9583 \\ 0.699 & 0.6609 & 0.2731 \end{bmatrix} \begin{bmatrix} 0.1 & -2.9 & -0.9 & 1.1 & 1.1 & 0.1 & -1.9 & 2.1 & 0.1 & 1.1 \\ 0.5 & -2.5 & -0.5 & 2.5 & 1.5 & -1.5 & -0.5 & 1.5 & 0.5 & -1.5 \\ -2.1 & 2.9 & -0.1 & -4.1 & -1.9 & -3.9 & -2.1 & 2.9 & -0.1 & -3.1 \end{bmatrix}$$

$$= \begin{bmatrix} -2.15 & 3.80 & 0.15 & -4.7 & 1.29 & 4.09 & -1.63 & 2.11 & -0.23 & -2.75 \\ -0.17 & -2.89 & -0.999 & 1.30 & 2.28 & 0.14 & -2.23 & 3.25 & 0.37 & -1.07 \end{bmatrix}$$

Tính $\hat{X}$

$$\hat{X} = W^T.Y + \bar{X} = \begin{bmatrix} 7.075 & 4.3582 & 6.1891 & 8.4573 & 8.3152 & 6.4364 & 5.5633 & 8.8818 & 7.1931 & 6.5306 \\ 3.9244 & 0.6388 & 2.8094 & 5.5389 & 4.6822 & 2.5682 & 2.4320 & 5.1129 & 3.8054 & 3.4814 \\ 2.9910 & 7.9570 & 4.9773 & 0.9451 & 6.9622 & 9.6076 & 2.9324 & 8.0142 & 4.9768 & 7.1762 \end{bmatrix}$$

$$MSE = \frac{1}{10} \sum_{i=1}^{10} (X_i - \hat{X}_i)^2 = 0.67493$$

b. Các bước chính của giải thuật tách lớp tuyến tính LDA

Tư tưởng chính của giải thuật tách lớp tuyến tính LDA là cực tiểu khoảng cách các phần tử trong cùng một lớp (distance within-class) và cực đại khoảng cách các phần tử nằm khác lớp với nhau (distance between-class). Giải thuật là có giám sát bởi vì dữ liệu đầu vào có dán nhãn trước. Sau đây là phần trình bày các bước chính của giải thuật.

Input - Đầu vào bài toán: Data labeled

$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}, x_i \in \mathcal{R}^D, y_i \in \{c_1, ..., c_k\}$$

Output - Đầu ra: ma trận chiếu W

Giả sử có k lớp. Đặt μ_i là vector trung bình của mỗi lớp i, với $i = 1, 2, ..., k$

Đặt N_i là số lượng mẫu trong mỗi lớp thứ i, với $i = 1, 2, ..., k$

Đặt $N = \sum_{i=1}^{k} N_i$ là tổng số lượng mẫu

Bước 01: Với mỗi lớp, tính toán vector trung bình D

Bước 02: Xây dựng ma trận phân tán between-class S_B và ma trận phân tán within-class S_W

Within-class scatter matrix

$$S_W = \sum_{i=1}^{k} \sum_{j=1}^{N_i} (x_j - \mu_i)(x_j - \mu_j)^T$$

Between-class scatter matrix

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

$$\mu = \frac{1}{k} \sum_{i=1}^{k} \mu_i$$

Bước 03: Tính toán vector riêng và giá trị tương ứng của ma trận $S_W^{-1} S_B$

$$S_b w_k = \lambda_k S_W w_k$$

Bước 04: Sắp xếp những giá trị riêng tương ứng với những vector riêng theo chiều giảm dần

Bước 05: Chọn n vector riêng tương ứng với n giá trị riêng lớn nhất để xây dựng ma trận biến đổi $D \times D$ chiều W, những vector riêng là những cột của ma trận này

$$\begin{bmatrix} b_1 \\ b_2 \\ ... \\ b_k \end{bmatrix} = \begin{bmatrix} w_1^T \\ w_2^T \\ ... \\ w_k^T \end{bmatrix} (x - \bar{x}) = W^T (x - \bar{x})$$

Bởi vì S_B có hạng lớn nhất là $k - 1$, số vector riêng lớn nhất khác 0 là $k - 1$

Trường hợp S_W^{-1} không tồn tại - Nếu S_W không là ma trận đơn (non-singular matrix)

$$S_W^{-1} S_B w_k = \lambda_k w_k$$

Trường hợp S_W^{-1} luôn luôn không tồn tại

- Dùng PCA trước

$$\begin{bmatrix} x_1 \\ x_2 \\ ... \\ x_N \end{bmatrix} \longrightarrow PCA \longrightarrow \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_K \end{bmatrix}$$

- Sau đó áp dụng LDA

$$\begin{bmatrix} x_1 \\ x_2 \\ ... \\ x_N \end{bmatrix} \longrightarrow LDA \longrightarrow \begin{bmatrix} y_1 \\ y_2 \\ ... \\ y_{k-1} \end{bmatrix}$$

Ví dụ bài toán phân lớp với LDA (Bài toán phân lớp với 2 lớp)

Cho hai lớp

- lớp 1 π_1: $X_1 = (x_1, x_2) = \{(4,2),(2,4),(2,3),(3,6),(4,4)\}$
- lớp 2 π_2: $X_2 = (x_1, x_2) = \{(9,10),(6,8),(9,5),(8,7),(10,8)\}$

Ta có $k = 2$ lớp
Đặt μ_i là vector trung bình của mỗi lớp i, với $i = 1, ..., k$
Đặt N_i là số lượng mẫu trong mỗi lớp thứ i, với $i = 1, ..., k$
Đặt $N = \sum_{i=1}^{k} N_i$ là tổng số lượng mẫu
Bước 01: Với mỗi lớp, tính toán vector trung bình D chiều
Gọi μ_1 và μ_2 là vector trung bình lớp thứ nhất và lớp thứ hai. Ta có:

$$\mu_1 = \frac{1}{N_1}\sum_{x\in\pi_1} x = \frac{1}{5}\left[\begin{pmatrix}4\\2\end{pmatrix} + \begin{pmatrix}2\\4\end{pmatrix} + \begin{pmatrix}2\\3\end{pmatrix} + \begin{pmatrix}3\\6\end{pmatrix} + \begin{pmatrix}4\\4\end{pmatrix}\right] = \begin{pmatrix}3\\3.8\end{pmatrix}$$

$$\mu_2 = \frac{1}{N_2}\sum_{x\in\pi_2} x = \frac{1}{5}\left[\begin{pmatrix}9\\10\end{pmatrix} + \begin{pmatrix}6\\8\end{pmatrix} + \begin{pmatrix}9\\5\end{pmatrix} + \begin{pmatrix}8\\7\end{pmatrix} + \begin{pmatrix}10\\8\end{pmatrix}\right] = \begin{pmatrix}8.4\\7.6\end{pmatrix}$$

Bước 02: Xây dựng ma trận phân tán between-class S_B và ma trận phân tán within-class S_W
Tính ma trận hiệp phương sai của lớp thứ nhất

$$S_1 = \sum_{x\in\pi_1}(x-\mu_1)(x-\mu_1)^T$$
$$= \left[\begin{pmatrix}4\\2\end{pmatrix} - \begin{pmatrix}3\\3.8\end{pmatrix}\right]^2 + \left[\begin{pmatrix}2\\4\end{pmatrix} - \begin{pmatrix}3\\3.8\end{pmatrix}\right]^2 + \left[\begin{pmatrix}2\\3\end{pmatrix} - \begin{pmatrix}3\\3.8\end{pmatrix}\right]^2 + \left[\begin{pmatrix}3\\6\end{pmatrix} - \begin{pmatrix}3\\3.8\end{pmatrix}\right]^2 + \left[\begin{pmatrix}4\\4\end{pmatrix} - \begin{pmatrix}3\\3.8\end{pmatrix}\right]^2$$
$$= \begin{pmatrix}1 & -0.25\\ -0.25 & 2\end{pmatrix}$$

Tính ma trận hiệp phương sai của lớp thứ hai

$$S_2 = \sum_{x\in\pi_2}(x-\mu_2)(x-\mu_2)^T$$
$$= \left[\begin{pmatrix}9\\10\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^2 + \left[\begin{pmatrix}6\\8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^2 + \left[\begin{pmatrix}9\\5\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^2 + \left[\begin{pmatrix}8\\7\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^2 + \left[\begin{pmatrix}10\\8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^2$$
$$= \begin{pmatrix}2.3 & -0.3\\ -0.3 & 3.3\end{pmatrix}$$

Ma trận phân tán between-class S_B

$$S_B = (\mu_1-\mu_2)(\mu_1-\mu_2)^T = \left[\begin{pmatrix}3\\3.8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]\left[\begin{pmatrix}3\\3.8\end{pmatrix} - \begin{pmatrix}8.4\\7.6\end{pmatrix}\right]^T = \begin{pmatrix}-5.4\\-3.8\end{pmatrix}\begin{pmatrix}-5.4 & -3.8\end{pmatrix} = \begin{pmatrix}29.16 & 20.52\\ 20.52 & 14.44\end{pmatrix}$$

Ma trận phân tán within-class S_W

$$S_W = S_1 S_2 = \begin{pmatrix}1 & -0.25\\ -0.25 & 2\end{pmatrix} + \begin{pmatrix}2.3 & -0.05\\ -0.05 & 3.3\end{pmatrix} = \begin{pmatrix}3.3 & -0.3\\ -0.3 & 5.5\end{pmatrix}$$

Bước 03: Tính toán vector riêng và giá trị tương ứng của ma trận $S_W^{-1}S_B$

$$S_W^{-1}S_B w_k = \lambda_k w_k$$

Ta có:

$$det(S_W^{-1}S_B - \lambda I_n) = 0$$
$$\Rightarrow det\left(\begin{pmatrix} 2.3 & -0.3 \\ -0.3 & 3.3 \end{pmatrix}^{-1}\begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix} - \lambda\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right) = 0$$
$$\Rightarrow det\left(\begin{pmatrix} 0.305 & 0.017 \\ 0.017 & 0.183 \end{pmatrix}\begin{pmatrix} 29.16 & 20.52 \\ 20.52 & 14.44 \end{pmatrix} - \lambda\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\right) = 0$$
$$\Rightarrow det\left(\begin{pmatrix} 9.2213 - \lambda & 6.489 \\ 4.2339 & 2.9794 - \lambda \end{pmatrix}\right) = 0$$
$$\Rightarrow (9.2213 - \lambda)(2.9794 - \lambda) - 6.489 \times 9.2213 = 0$$
$$\Rightarrow \lambda^2 - 12.2007\lambda = 0$$

Ta thu được hai giá trị riêng:

- $\lambda_1 = 0$
- $\lambda_2 = 12.2007$

Bước 04: Sắp xếp những giá trị riêng tương ứng với những vector riêng theo chiều giảm dần

- $\lambda_2 = 12.2007$
- $\lambda_1 = 0$

Tìm các vector riêng, ta có:

$$\begin{pmatrix} 9.2213 & 6.489 \\ 4.2339 & 2.9794 \end{pmatrix} W_1 = 0\begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$
$$\begin{pmatrix} 9.2213 & 6.489 \\ 4.2339 & 2.9794 \end{pmatrix} W_2 = 12.2007\begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

Tính được:

$$W_1 = \begin{pmatrix} -0.5755 \\ 0.8178 \end{pmatrix}$$
$$W_2 = \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix}$$

Chọn được ma trận chiếu là

$$W = \begin{pmatrix} 0.9088 \\ 0.4173 \end{pmatrix}$$

c. Điểm khác biệt giữa PCA và LDA

PCA	LDA
Kỹ thuật biến đổi tuyến tính, một kỹ thuật giảm chiều dữ liệu không giám sát	Kỹ thuật biến đổi tuyến tính, một kỹ thuật giảm chiều dữ liệu có giám sát
Không quan tâm về nhãn dữ liệu	Quan tâm về nhãn dữ liệu, dữ liệu đầu vào phải được gán nhãn trước khi chạy thuật toán LDA
PCA tập trung cực đại phương sai của dữ liệu	LDA tập trung cực tiểu khoảng cách các phần tử trong cùng một lớp (within classes distance) và cực đại khoảng cách giữa các lớp với nhau (between classes distance)

A.3 Kiến thức về Support Vector Machines (SVM)

a. Trình bày các bước thực hiện của giải thuật phân lớp dùng véc tơ hỗ trợ SVM. Phân tích cụ thể từng bước của giải thuật

Các bước thực hiện của giải thuật phân lớp dùng SVM

Bước 01: Chọn kernel function

Bước 02: Chọn một giá trị cho C

Bước 03: Giải bài toán quy hoạch toàn phương (Solve the quadratic programming problem)

Không gian để so sánh các siêu phẳng (hyperplane)

Xem xét phương trình siêu phẳng $w \cdot x + b = 0$. Nếu một điểm dữ liệu có tọa độ (x, y) trong không gian nằm trên siêu phẳng này, nó thõa mãn phương trình $w \cdot x + b = 0$. Nếu một điểm dữ liệu không nằm trên siêu phẳng, giá trị của biểu thức $w \cdot x + b$ có thể âm hoặc dương. Để biết được điểm dữ liệu nào nằm gần siêu phẳng nhất, ta có thể tính toán $\beta = |w \cdot x + b|$

Cho tập dữ liệu $\mathcal{D} = \{(x_i, y_i)|x_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\}_{i=1}^m$, với mỗi phần tử trong tập huấn luyên, tính toán được β, gọi B là giá trị nhỏ nhất của β

$$B = \min_{i=\{1...m\}} |w \cdot x + b|$$

Nếu ta có k hyperplane, mỗi hyperplane có một giá trị B_i tương ứng, ta chỉ cần chọn hyperplane với giá trị B_i lớn nhất

$$H = \max_{i=\{1...k\}} \{h_i|B_i\}$$

Vấn đề ở đây là ta có thể thất bại trong việc phân tách một siêu phẳng tốt với một siêu phẳng không tốt, vì có thể chúng có cùng giá trị.

Do đó, ta sử dụng thông tin về nhãn y. Định nghĩa hàm $f = y(w.x + b)$ và dấu của hàm f cho biết điểm dữ liệu được phân đúng lớp (dấu dương) và phân sai lớp (dấu âm)

Dùng tập dữ liệu $\mathcal{D} = \{(x_i, y_i)|x_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\}_{i=1}^m$, với mỗi phần tử trong tập huấn luyên, tính toán f và đặt F là giá trị f nhỏ nhất

$$F = \min_{i=\{1...m\}} y_i(w \cdot x + b)$$

Nếu ta có k hyperplane, hyperplane nào có giá trị F lớn nhất sẽ được chọn (F được gọi là function margin)

Nếu ta có những vectors có cùng vector cơ sở, ví dụ như $w_1 = (5, 6)$ và $w_2 = (50, 60)$, nó không bất biến với phép co dãn (scale).

Để khắc phục yếu điểm đó, ta chia f với độ lớn vector w. Ta định nghĩa được

$$\gamma = y\left(\frac{w}{||w||} \cdot x + \frac{b}{||w||}\right)$$

Dùng tập dữ liệu $\mathcal{D} = \{(x_i, y_i)|x_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\}_{i=1}^m$, với mỗi phần tử trong tập huấn luyện, tính toán được γ và ta gọi M là giá trị nhỏ nhất, M được gọi là geometric margin của tập dữ liệu

$$M = \underset{i=\{1...m\}}{min} y_i\left(\frac{w}{||w||} \cdot x + \frac{b}{||w||}\right)$$

Nếu ta có k hyperplane, ta sẽ chọn hyperplane với một giá trị M lớn nhất.

Bài toán của chúng ta là tối ưu siêu phẳng, tức là cần tìm giá trị w và b của một siêu phẳng tối ưu, với ràng buộc geometric margin của mỗi phần tử phải lớn hơn hoặc bằng M:

$$\max_{w,b} M$$
$$\text{subject to } \gamma_i \geq M, i = 1...m$$

Do $M = F/||w||$. Ta có thể viết lại bài toán ràng buộc như sau:

$$\max_{w,b} M$$
$$\text{subject to } f_i \geq F, i = 1...m$$

Khi co dãn w và b, bài toán không thay đổi do đó:

$$\max_{w,b} \frac{1}{||w||}$$
$$\text{subject to } f_i \geq 1, i = 1...m$$

$$\Leftrightarrow \min_{w,b} ||w||$$
$$\text{subject to } f_i \geq 1, i = 1...m$$

$$\Leftrightarrow \min_{w,b} \frac{1}{2}||w||^2$$
$$\text{subject to } y_i(w.x + b) - 1 \geq 0, i = 1...m$$

Giải bài toán tối ưu SVM (SVM optimization problem) - Hard Margin SVM
Lagrange phát biểu rằng nếu chúng ta tìm cực tiểu hàm f dưới một ràng buộc g, chúng ta chỉ cần giải:

$$\nabla f(x) - \alpha \nabla g(x) = 0$$

Trong đó: α được gọi là nhân tử Lagrange (Lagrange multiplier)

Với bài toán này, ta có hàm $f(w) = \frac{1}{2}||w||^2$, $g(w,b) = y_i(w.x+b-1, i = 1...m$. Hàm Lagrange được khai triển như sau:

$$\mathcal{L}(w,b,\alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i[y_i(w \cdot x + b - 1]$$

Để giải phương trình $\nabla\mathcal{L}(w,b,\alpha) = 0$, ta viết lại bài toán theo nguyên lý đối ngẫu (duality principle)

$$\min_{w,b} \ \max \ \mathcal{L}(w,b,\alpha)$$
$$\text{subject to } \alpha_i \geq F, i = 1...m$$

Một cách chính xác hơn, α nên là nhân tử Karush-Kuhn-Tucker bởi vì ta phải vấn đề ràng buộc bất công bằng.
Với Lagrangian function:

$$\mathcal{L}(w,b,\alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i[y_i(w \cdot x + b - 1]$$

Ta có

$$\nabla_w\mathcal{L}(w,b,\alpha) = w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0$$

$$\nabla_b\mathcal{L}(w,b,\alpha) = -\sum_{i=1}^{m} \alpha_i y_i = 0$$

Từ hai phương trình ở trên, ta có $w = \sum_{i=1}^{m} \alpha_i y_i x_i$ và $\sum_{i=1}^{m} \alpha_i y_i = 0$. Thay vào hàm Lagrangian

$$W(\alpha,b) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j x_i \cdot x_j$$

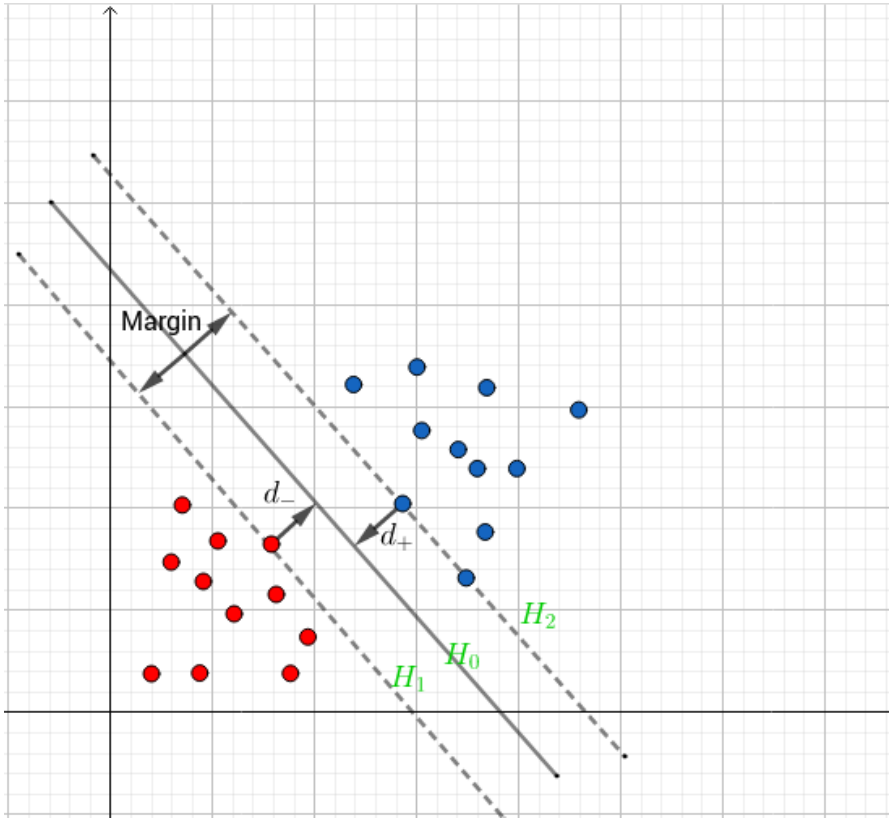
Bài toán được viết lại như sau:

$$\max_{\alpha} \ \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j x_i \cdot x_j$$
$$\text{subject to } \alpha_i \geq 0, i = 1...m, \sum_{i=1}^{m} \alpha_i y_i = 0$$

Mở rộng nhân tử Lagrange thành điều kiện (Karush-Kuhn-Tucker):

$$\alpha_i[y_i(w \cdot x^* + b) - 1] = 0$$

Trong đó, x^* là điểm mà đạt được tối ưu, α là giá trị tri dương cho những điểm tối ưu này, và có giá trị gần về 0 đối với những điểm còn lại. Do đó mà, $y_i(w \cdot x^* + b) - 1$ phải bằng 0, các điểm này gọi là support vectors, gần nhất với siêu phẳng.



Hình 12: Support Vector Machine

Và ta cần tính toán w và b để đưa ra quyết định về siêu phẳng tối ưu.

$$w - \sum_{i=1}^{m} \alpha_i y_i x_i = 0$$
$$\Leftrightarrow w = \sum_{i=1}^{m} \alpha_i y_i x_i$$

Để tính toán giá trị b

$$y_i(w \cdot x^* + b) - 1 = 0$$
$$\Leftrightarrow y_i^2(w \cdot x^* + b) - y_i = 0$$
$$\Leftrightarrow b = y_i - w \cdot x^*$$

Lưu ý: $y_i^2 = 1 \forall i$ Và giá trị b cũng có thể được tính

$$b = \frac{1}{S} \sum_{i=1}^{S} (y_i - w \cdot x)$$

Bước 04: Xây dựng hàm phân lớp từ những support vectors
Bộ phân lớp được định nghĩa để đưa ra dự đoán. Hypothesis function h:

$$h(x_i) = \begin{cases} +1, w \cdot x + b \geq 0 \\ -1, w \cdot x + b < 0 \end{cases}$$

Giải bài toán tối ưu SVM (SVM optimization problem) - Soft Margin SVM

Bài toán Hard Margin SVM giải quyết tốt cho dữ liệu phân hoạch tuyến tính. Tuy nhiên, điều này dường như hiếm gặp trong thực tế. Hầu hết các trường hợp dữ liệu sẽ chứa nhiễu và có thể không phân hoạch tuyến tính (phân bố dạng cầu, ...)

Soft Margin SVM sử dụng một trick rất đơn giản, thêm vào những biến chùng (slack variables) ς_i vào ràng buộc của bài toán tối ưu

$$y_i(w \cdot x_i + b) \geq 1 - \varsigma_i, i = 1...m$$

Nhờ sử dụng những biến chùng trong ràng buộc, khi cực tiểu hàm mục tiêu, nó có khả năng thõa mãn ràng buộc cho dù dữ liệu huấn luyện không thõa ràng buộc ban đầu. Và sử dụng L1 Regularization để giải quyết vấn đề của những giá trị biến chùng lớn. Ta có bài toán tối ưu như sau:

$$\Leftrightarrow \min_{w,b} \frac{1}{2}||w||^2 + \sum_{i=1}^{m} \varsigma_i$$
$$\text{subject to } y_i(w.x + b) \geq 1 - \varsigma_i, i = 1...m$$

Và để tránh việc phân lớp sai trong mỗi huấn luyện, ta thêm vào ràng buộc biến chùng không âm và tham số C gọi là regularization parameter để xác định giá trị biến chùng ς quan trọng như thế nào?

$$\Leftrightarrow \min_{w,b} \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} \varsigma_i$$
$$\text{subject to } y_i(w.x + b) \geq 1 - \varsigma_i, \varsigma_i \geq 0, i = 1...m$$

Và tiếp tục sử dụng nhân tử Lagrange, đưa bài toán về bài toán kép

$$\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j x_i \cdot x_j$$
$$\text{subject to } 0 \leq \alpha_i \leq C, i = 1...m, \sum_{i=1}^{m} \alpha_i y_i = 0$$

b. Liệt kê một số nhân (kernel) phổ biến được sử dụng trong SVM

Một số nhân (kernel) phổ biết được sử dụng trong SVM như

1) Linear kernel: $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j$

2) Polynomial kernel: $\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T\mathbf{x}_j)^p$

3) Gaussian (Radial-Basis Function (RBF)) kernel:

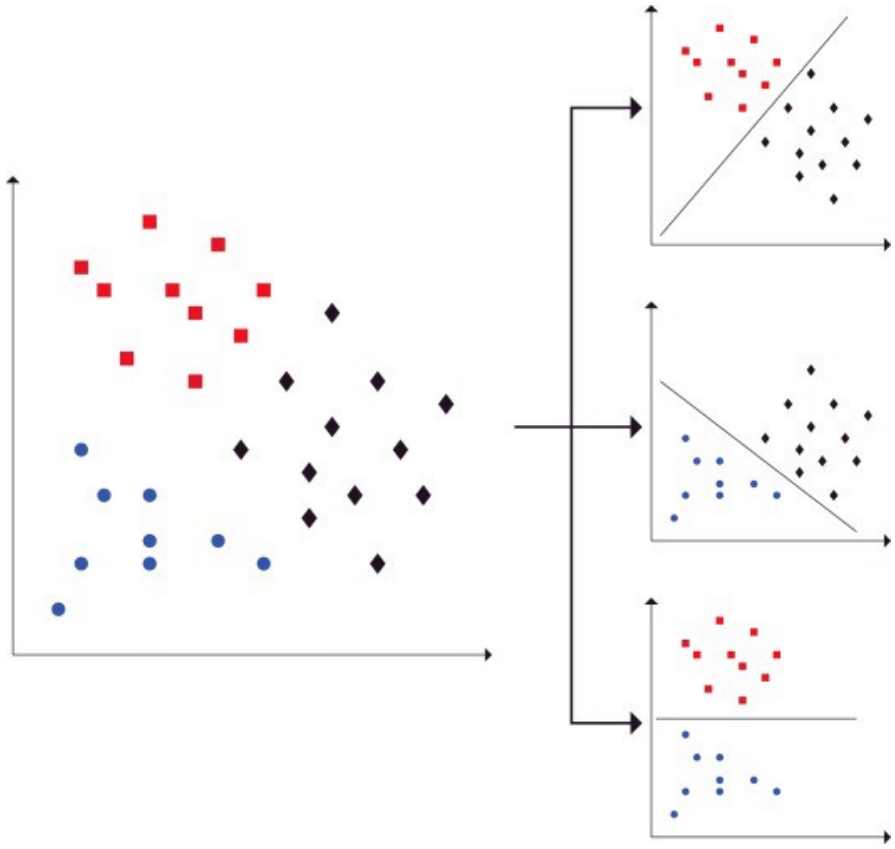
$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = exp\left(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}\right)$$

4) Sigmoid:

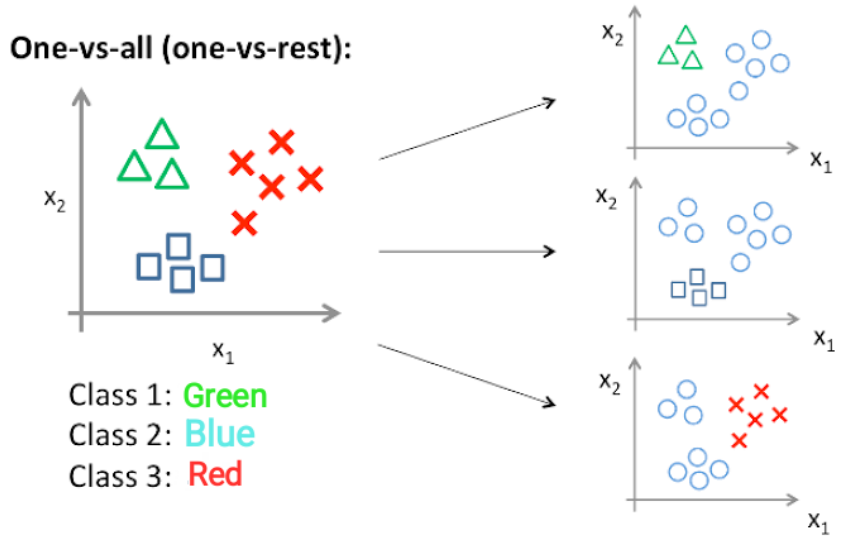
$$\mathbf{K}(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0\mathbf{x}_i^T\mathbf{x}_j + \beta_1)$$

c. Phân biệt các chiến lược phân đa lớp: one vs. all và one vs. one

	One vs one	One vs all
Ưu điểm	Dễ dàng sử dụng sau khi xử lí. Phù hợp khi bài toán có nhiều lớp.	Nhanh hơn khi hội tụ phù hợp khi bài toán có ít lớp.
Nhược điểm	Chậm hơn phân loại 2 lớp trong quá trình train.	Gần như không thể dùng khi có quá nhiều lớp.
Phương pháp	Thuật toán sử dụng cây. Neural network.	SVM. Các phương pháp tập hợp. Thuật toán sử dụng cây.



Hình 13: Chiến lược one vs. one



Hình 14: Chiến lược one vs. all

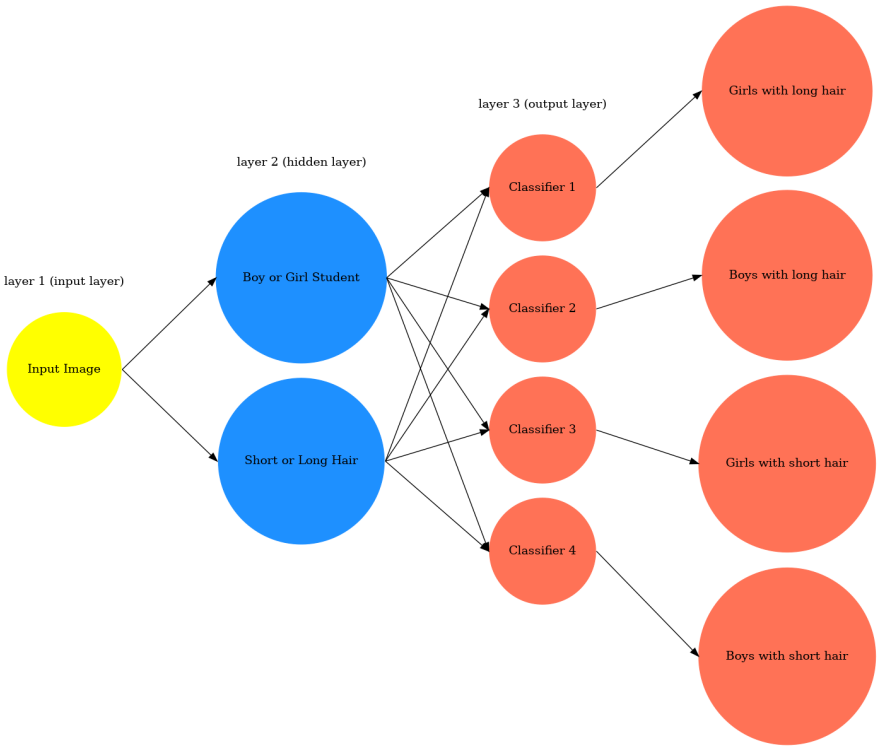
A.4 Kiến thức về mạng nơron cơ bản (ANN)

Cho trước một mạng neural với hai tầng: - Tầng 1: nhận vào một ảnh nhập - Tầng 2 là tầng xuất ứng với 4 bộ phân lớp được xuất ra (Bộ phân lớp 1 đại diện cho 'học sinh nữ có tóc dài'; Bộ phân lớp 2 đại diện cho 'học sinh nam có tóc dài'; Bộ phân lớp 3 đại diện cho 'học sinh nữ có tóc ngắn' và Bộ phân lớp 4 đại diện cho 'học sinh nam có tóc ngắn').

Tuy nhiên, dữ liệu huấn luyện mạng phân bố không đều: - 4 mẫu cho bộ phân lớp 1 - 4 mẫu cho bộ phân lớp 3 - 4 mẫu cho bộ phân lớp 4 - Riêng bộ phân lớp 2 chỉ có 1 mẫu huấn luyện

Như vậy, bộ phân lớp 2 là yếu. Học viên hãy đề nghị một cấu trúc mạng nơ ron mới sao cho nâng hiệu quả bộ lớp 2 mà vẫn sử dụng bộ mẫu huấn luyện nói trên? Nêu lên tính hiệu quả của cấu trúc mạng nơ ron đề nghị?

Đề nghị một cấu trúc mạng Neural tăng thêm một tầng cơ sở - Tầng 01: Tầng nhận ảnh đầu vào - Tầng 02 là tầng ẩn: Trong tầng này gồm hai bộ phân lớp - Bộ phân lớp 1 dùng để phân loại học sinh nam hay học sinh nữ gồm 8 mẫu huấn luyện cho nữ và 8 mẫu nam - Bộ phân lớp 2 dùng để phân loại tóc ngắn hay tóc dài gồm 5 mẫu huấn luyện tóc dài và 8 mẫu tóc ngắn - Tầng 03 là tập đầu ra với 4 bộ phân lớp: học sinh nữ có tóc dài, học sinh nam có tóc dài, học sinh nữ có tóc ngắn, học sinh nam có tóc ngắn



Hình 15: Kiến trúc mạng đề xuất

Tính hiệu quả của cấu trúc mạng Neural: - Xây dựng thêm một tầng cơ sở với 2 bộ phân lớp, tăng cường dữ liệu cho mỗi bộ phân lớp, nên sẽ trả về kết quả tốt - Tầng phân lớp (tầng xuất) chỉ có nhiệm vụ tổng hợp các kết quả nhận được từ tầng cơ sở (không cần phải huấn luyện)

A.5 Kiến thức về mạng nơron sâu (DNN)

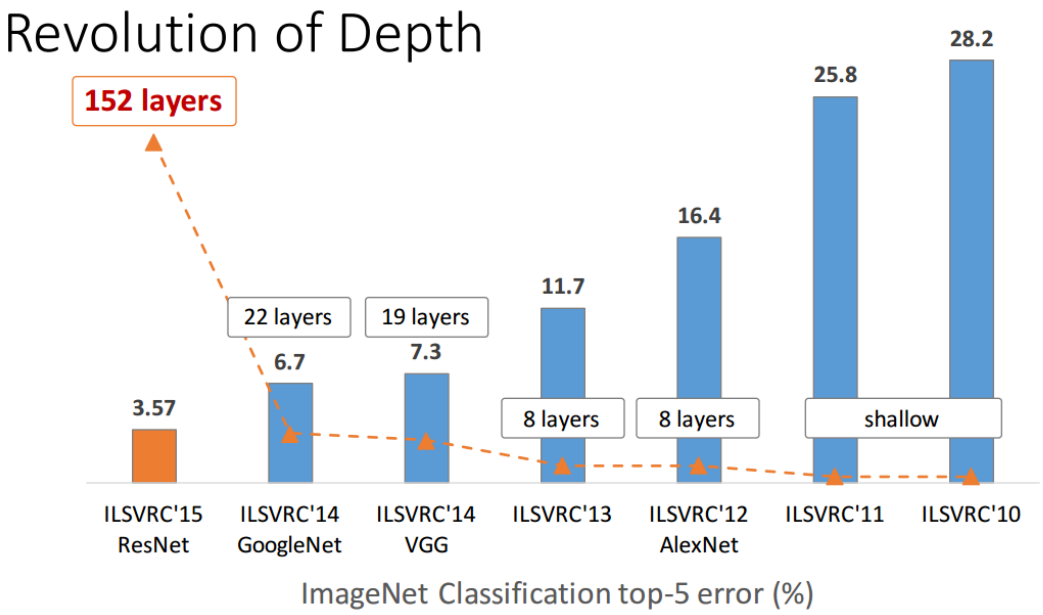
a. Giải thích vì sau mạng nơron sâu thường cho kết quả tốt hơn mạng nơron rộng với cùng số lượng nơron.

Mạng neural rộng thường cần nhiều neural hơn để có thể thực hiện các tác vụ giống như mạng neural sâu: số lượng đơn vị trong một mạng neural rộng tăng lên theo cấp số nhân với độ phức tạp của nhiệm vụ. Vì vậy, để trở nên hữu ích, một mạng neural rộng có thể cần phải rất lớn; có thể lớn hơn nhiều so với một mạng sâu. Điều này dựa trên một số tài liệu chứng minh rằng các mạng neural rộng trong một số trường hợp cần nhiều nơ-ron theo cấp số nhân.

Có thể một mạng neural rộng sẽ khó đào tạo hơn với các thuật toán hiện tại (ví dụ: khó xác định cực tiểu cục bộ hơn hoặc tốc độ hội tụ chậm hơn,. . .)

Nhiều lớp thực hiện chung 1 tác vụ, hoặc có thể mỗi lớp xử lý một tác vụ khác nhau. Linh hoạt trong nhiều nhiệm vụ khác nhau.

Kiến trúc của mạng neural rộng thường không phù hợp với các tác vụ hiện nay như nhận dạng, nhận diện.



Hình 16: Revolution of Depth

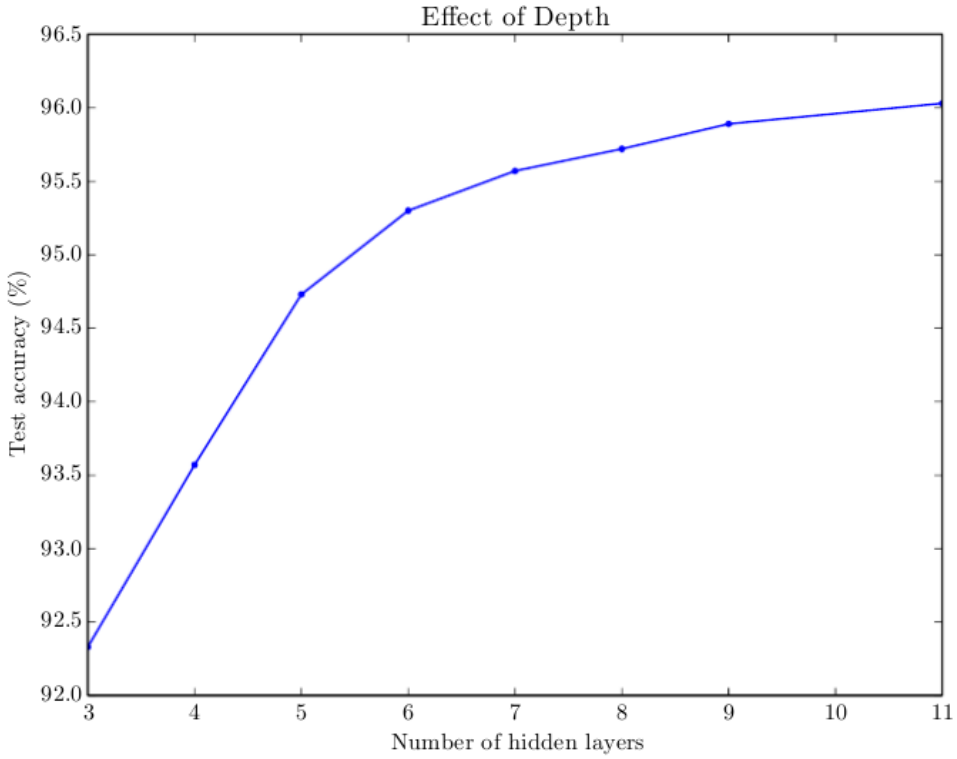
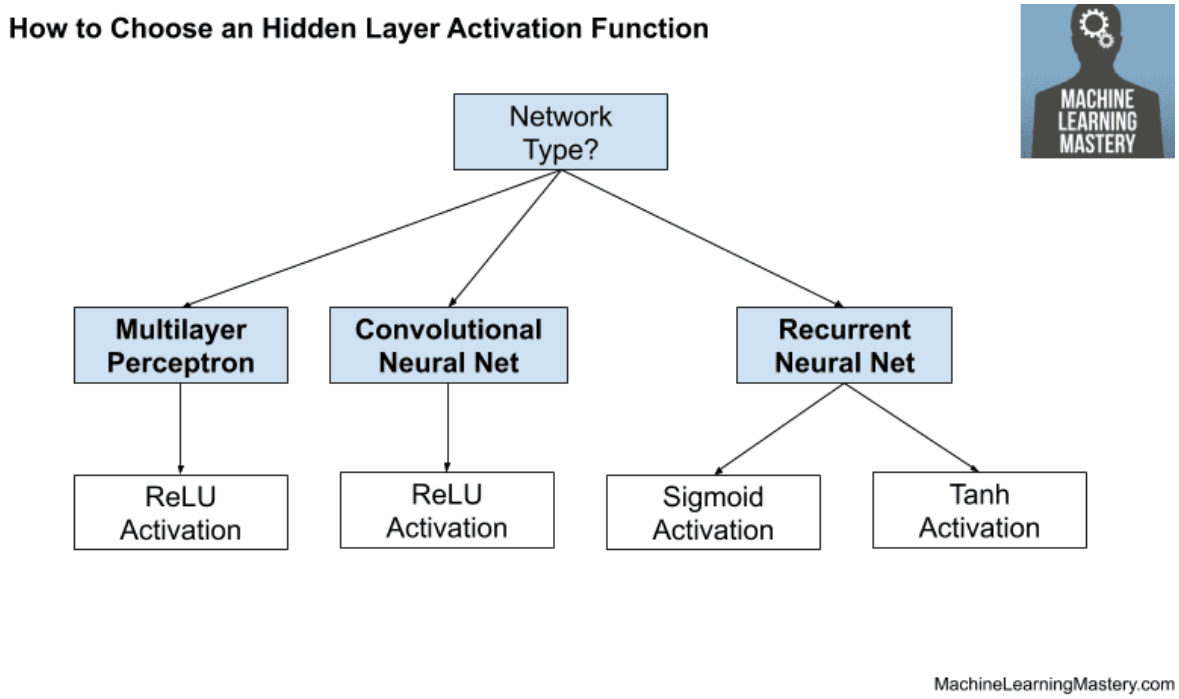


Figure 6.6: Empirical results showing that deeper networks generalize better when used to transcribe multi-digit numbers from photographs of addresses. Data from Goodfellow *et al.* (2014d). The test set accuracy consistently increases with increasing depth. See Fig. 6.7 for a control experiment demonstrating that other increases to the model size do not yield the same effect.

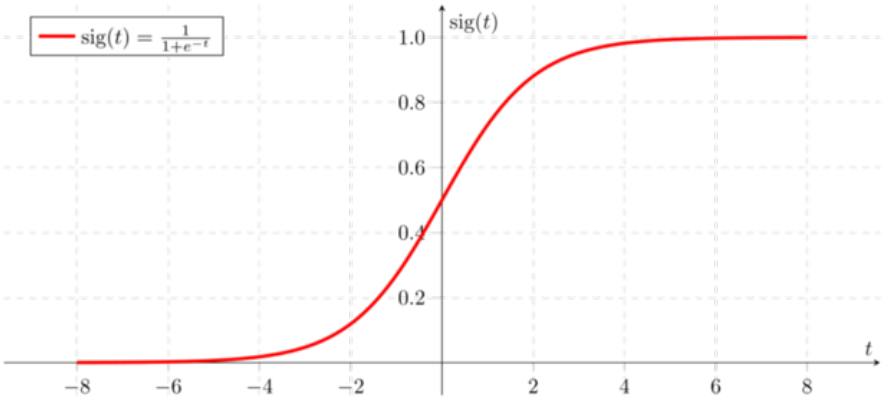
Hình 17: Revolution of Depth

b. Cho biết 3 dạng hàm kích hoạt (activation function) thường được sử dụng trong mạng sâu



Hình 18: Revolution of Depth

Sigmoid function



Hình 19: Trực quan hóa hàm Sigmoid Activation Function

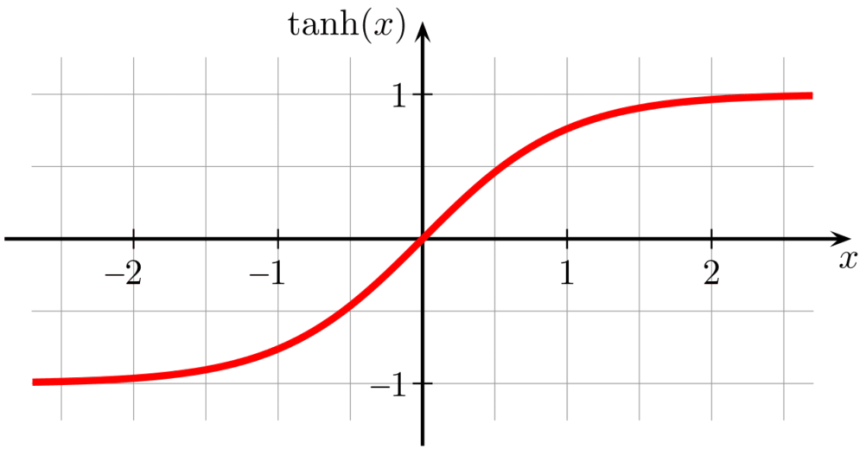
Biểu diễn toán học cho hàm như sau:

$$S(x) = \frac{1}{1 + e^{-x}}$$

Hàm kích hoạt tuyến tính tốt hơn trong việc cung cấp một loạt các kích hoạt và đường có độ dốc dương có thể làm tăng tốc độ bắn khi tốc độ đầu vào tăng.

- Ưu điểm: Đây là một loại hàm kích hoạt khá đơn giản và được sử dụng phổ biến vào những năm 1990.
- Khuyết điểm: độ bão hòa gradient, hội tụ chậm, gradient ẩm rõ nét(nguyên văn là sharp damp gradients) trong quá trình lan truyền ngược từ bên trong các lớp ẩn sâu hơn đến các lớp đầu vào, và đầu ra có điểm trung tâm khác 0 khiến các cập nhật gradient lan truyền theo các hướng khác nhau.

Tanh function



Hình 20: Trực quan hóa hàm Tanh Activation Function

Biểu diễn toán học của hàm ReLU

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

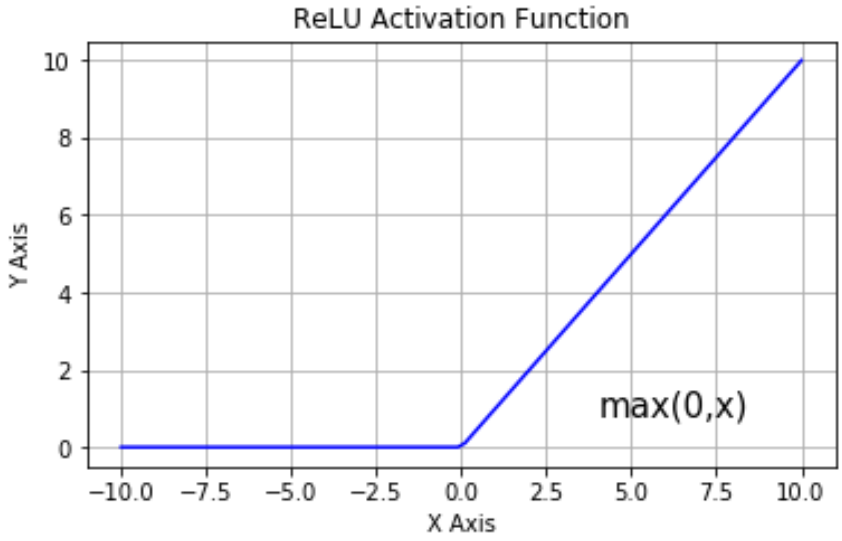
Ưu điểm

- Hàm tanh được sử dụng rộng rãi hơn nhiều so với hàm sigmoid vì nó mang lại hiệu suất đào tạo tốt hơn cho các mạng nơ-ron đa lớp. Ưu điểm lớn nhất của hàm tanh là nó tạo ra đầu ra với điểm 0 là trung tâm, do đó hỗ trợ quá trình lan truyền ngược. Hàm tanh chủ yếu được sử dụng trong các mạng nơ-ron hồi quy cho các tác vụ xử lý ngôn ngữ tự nhiên và nhận dạng giọng nói.

Nhược điểm

- Hàm tanh cũng có một hạn chế - giống như hàm sigmoid, nó không thể giải quyết vấn đề gradient biến mất. Ngoài ra, hàm tanh chỉ có thể đạt được độ dốc bằng 1 khi giá trị đầu vào là 0 (x bằng 0). Kết quả là, hàm có thể tạo ra một số tế bào thần kinh chết trong quá trình tính toán.

ReLU Activation Function



Hình 21: Trực quan hóa hàm ReLU Activation Function

Biểu diễn toán học của hàm ReLU

$$f(x) = max(0, x)$$

Ưu điểm của ReLU:

- Bằng cách ràng buộc các giá trị của đầu vào nhỏ hơn 0 và đặt chúng thành 0, và ngưỡng tối đa của hàm này ở vô cực - loại bỏ vấn đề gradient biến mất được quan sát trong các loại hàm kích hoạt trước đó (sigmoid và tanh). Ưu điểm đáng kể nhất của việc sử dụng hàm ReLU trong tính toán là nó đảm bảo tính toán nhanh hơn - nó không tính toán theo cấp số nhân và phép chia, do đó thúc đẩy tốc độ tính toán tổng thể. Một khía cạnh quan trọng khác của hàm ReLU là nó tạo ra sự thưa thớt trong các đơn vị ẩn bằng cách bình phương các giá trị từ 0 đến lớn nhất.

Nhược điểm của ReLU:

- ReLU có thể chết trong quá trình huấn luyện, nơi chúng ngừng học hoàn toàn, vì vậy chúng ta cần phải cẩn thận về phạm vi giá trị đầu vào.

ACTIVATION FUNCTION	SIGMOID	TANH	RELU
Range	0 to 1	-1 to 1	0 to Infinity
Vanishing Gradient Problem	Yes	Yes	No
Nature	Non Linear	Non Linear	Linear
Zero Centered Activation Function	No	Yes	No
Symmetric Function	No	Yes	No
Equation	y = 1/(1+e^(-x))	y = tanh(x)	{ xi if x >=0 0 if x <=0 }
Model Accuracy	Good	Very Good	Excellent

c. Cho biết 3 bộ tối ưu hóa (optimizer) thường được sử dụng trong mạng sâu

Mục tiêu chính của học máy là tạo ra một mô hình hoạt động tốt và đưa ra các dự đoán chính xác trong một số trường hợp cụ thể. Để đạt được điều đó, chúng ta cần tối ưu hóa học máy. Tối ưu hóa học máy là quá trình điều chỉnh các siêu tham số để giảm thiểu hàm chi phí bằng cách sử dụng

một trong các kỹ thuật tối ưu hóa. Điều quan trọng là phải giảm thiểu hàm chi phí vì nó mô tả sự khác biệt giữa giá trị thực của thông số ước tính và những gì mô hình đã dự đoán.

1) Batch normalization

Ý tưởng cơ bản: Chuẩn hóa là một kỹ thuật để thay đổi giá trị của các cột số trong tập dữ liệu thành một tỷ lệ chung mà không làm sai lệch sự khác biệt trong phạm vi giá trị. Kỹ thuật này thường được áp dụng như một phần của quá trình chuẩn bị dữ liệu cho học máy và cần thiết nếu các đặc trưng đầu vào khác nhau nằm trong một phạm vi giá trị khác nhau.

Ưu điểm

- Mạng huấn luyện nhanh hơn - Mỗi lần lặp lại việc huấn luyện sẽ chậm hơn do các tính toán bổ sung trong quá trình chuyển tiếp và các siêu tham số bổ sung để huấn luyện trong quá trình lan truyền ngược. Tuy nhiên, nó sẽ hội tụ nhanh hơn nhiều, vì vậy việc huấn luyện tổng thể trở nên nhanh hơn.
- Cho phép tỷ lệ học tập cao hơn - Giảm dần độ dốc thường yêu cầu tốc độ học tập nhỏ để dễ hội tụ. Và đối với mạng học sâu, độ dốc của chúng sẽ nhỏ dần trong quá trình lan truyền ngược, vì vậy chúng đòi hỏi nhiều lần lặp lại hơn. Sử dụng chuẩn hóa hàng loạt cho phép người dùng sử dụng tốc độ học tập cao hơn nhiều, điều này làm tăng thêm tốc độ huấn luyện mạng.
- Làm cho trọng số khởi tạo dễ dàng hơn - Khởi tạo trọng số có thể khó và càng khó hơn khi tạo các mạng học sâu. Chuẩn hóa hàng loạt dường như cho phép chúng ta bớt cẩn thận hơn trong việc chọn trọng số ban đầu.
- Làm cho nhiều hàm kích hoạt khả thi hơn - Một số hàm kích hoạt không hoạt động tốt trong một số trường hợp. Sigmoid mất gradient khá nhanh, có nghĩa là chúng không thể được sử dụng trong các mạng sâu. Và ReLU thường chết trong quá trình huấn luyện, nơi chúng ngừng học hoàn toàn, vì vậy chúng ta cần phải cẩn thận về phạm vi giá trị đầu vào. Chuẩn hóa hàng loạt, điều chỉnh các giá trị đi vào mỗi hàm kích hoạt, các yếu tố phi tuyến tính dường như không hoạt động tốt trong các mạng học sâu giờ đây đã có thể hoạt động tốt trở lại.
- Đơn giản hóa việc tạo các mạng học sâu - Do 4 ưu điểm được liệt kê ở trên.
- Có thể cho kết quả tổng quát tốt hơn - Một số thử nghiệm dường như cho thấy quá trình chuẩn hóa hàng loạt thực sự cải thiện kết quả huấn luyện.

Nhược điểm

- Khó ước tính giá trị trung bình và độ lệch chuẩn của đầu vào trong quá trình kiểm tra.
- Không thể sử dụng kích thước batch là 1 trong quá trình huấn luyện.
- Chi phí tính toán trong quá trình huấn luyện khá cao. bước chuyển tiếp lịch sử được mang theo mỗi bước, và những đi những bước lớn thì không phải lúc nào cũng tốt
- Cách tính khác nhau giữa việc huấn luyện và kiểm tra không phù hợp cho Mạng Neural Hồi quy (Recurrent Neural Network) và Mạng bộ nhớ ngắn hạn dài (Long Short-Term Memory Network)

2) Mini-batch gradient descentLà một biến thể của thuật toán gradient descent chia các tập dữ liệu huấn luyện thành các nhóm nhỏ được sử dụng để tính toán lỗi mô hình và cập nhật hệ số mô hình.

Ưu điểm

- Tần suất cập nhật mô hình cao hơn so với Batch gradient descent.
- Kiểm soát dữ liệu huấn luyện trong bộ nhớ tốt hơn giúp cho qua trình triển khai thuật toán hoạt động trơn tru.
- Cập nhật liên tục, cung cấp một quy trình tính toán hiệu quả hơn so với Stochastic gradient descent.

Nhược điểm

- Không đảm bảo sự hội tụ của một lỗi theo cách tốt hơn.

- Vì 50 mẫu dữ liệu được lấy, không đại diện cho các thuộc tính (hoặc phương sai) của toàn bộ tập dữ liệu. Do đó, đây là lý do mà chúng ta sẽ không thể có được sự hội tụ, tức là chúng ta sẽ không nhận được cực tiểu cục bộ hoặc toàn cục tại bất kỳ thời điểm nào.
- Khi sử dụng Mini-batch gradient descent, vì các mẫu được lấy theo từng nhóm, do đó, có thể xảy ra trường hợp trong một số nhóm xuất hiện lỗi này, một số lại xuất hiện lỗi khác. Vì vậy, chúng ta sẽ phải tự mình kiểm soát tốc độ học tập, bất cứ khi nào chúng ta sử dụng Mini-Batch gradient descent. Nếu tỷ lệ học tập rất thấp, tỷ lệ hội tụ cũng sẽ giảm. Nếu tỷ lệ học tập quá cao, ta sẽ không nhận được mức cực tiểu cục bộ hoặc toàn cục. Vì vậy chúng ta cần kiểm soát tốc độ học.

3) RMSProp optimization RMSprop là một kỹ thuật tối ưu hóa dựa trên gradient. Nó được đề xuất bởi cha đẻ của lan truyền ngược, Geoffrey Hinton. Gradient của các hàm phức tạp như mạng nơ-ron có xu hướng biến mất khi dữ liệu truyền qua các hàm đó. RMSprop được phát triển như một kỹ thuật ngẫu nhiên để học theo từng nhóm nhỏ.

RMSprop giải quyết vấn đề trên bằng cách sử dụng đường trung bình động của các bình phương gradient để chuẩn hóa gradient. Quá trình chuẩn hóa này cân bằng kích thước ở mỗi bước (bước nhảy), giảm bước cho các gradient lớn để tránh sự đột ngột tăng lên và tăng bước cho các gradient nhỏ để tránh biến mất.

Ưu điểm

- RMSProp hội tụ nhanh, trong nhiều cài đặt. Đặc biệt là đối với GAN, RL và mạng dựa trên sự chú ý. RMSProp có thể hội tụ các giải pháp với chất lượng tốt hơn (ví dụ: cực tiểu cục bộ tốt hơn). Ít biến thiên so với các thuật toán như Gradient Descent.

Nhược điểm

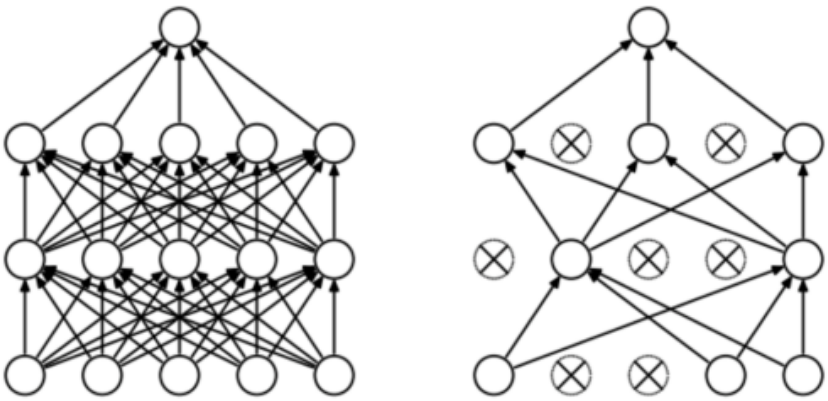
- Tỷ lệ học tập vẫn phải cài đặt thủ công, vì giá trị đề xuất không phải lúc nào cũng phù hợp cho mọi tác vụ. Triển khai RMSProp Descent cần có sự giám sát.

d. Trình bày kỹ thuật Dropout

Thuật ngữ "Dropout"đề cập đến việc bỏ qua các đơn vị (units) ẩn và hiện trong 1 mạng Neural.

Hiểu 1 cách đơn giản thì Dropout là việc bỏ qua các đơn vị (tức là 1 nút mạng) trong quá trình huấn luyện 1 cách ngẫu nhiên. Bằng việc bỏ qua này thì đơn vị đó sẽ không được xem xét trong quá trình lan truyền và lan truyền ngược.

Theo đó, p được gọi là xác suất giữ lại 1 nút mạng trong mỗi giai đoạn huấn luyện, vì thế xác suất nó bị loại bỏ là $(1^{\smile}p)$.



(a) Standard Neural Net (b) After applying dropout.

Công dụng: Làm đơn giản mô hình, tránh overfitting. Thêm vào đó, vì mỗi bước khi train model thì xác suất (1-p) các node bị loại bỏ nên model không thể phụ thuộc vào bất kì node nào của layer trước mà thay vào đó có xu hướng trải đều weights.

Một số tips: Hệ số p nên ở khoảng $[0.2, 0.5]$. Nếu p quá nhỏ thì không có tác dụng chống overfitting, tuy nhiên nếu p quá lớn thì gần như loại bỏ layer đấy và có dễ dẫn đến underfitting. Nên dùng model lớn, phức tạp hơn vì ta có dropout chống overfitting. Dropout chỉ nên dùng cho fully connected layer, ít khi được dùng cho ConvNet layer. Hệ số p ở các layer nên tỉ lệ với số lượng node trong FC layer đó.

Kỹ thuật này được sử dụng như sau:

- Trong pha train: với mỗi hidden layer, với mỗi trainning sample, với mỗi lần lặp, chọn ngẫu nhiên p phần trăm số node và bỏ qua nó (bỏ qua luôn hàm kích hoạt cho các node bị bỏ qua).
- Trong pha test: Sử dụng toàn bộ activations, nhưng giảm chúng với tỷ lệ p (do chúng ta bị miss $p\%$ hàm activation trong quá trình train).

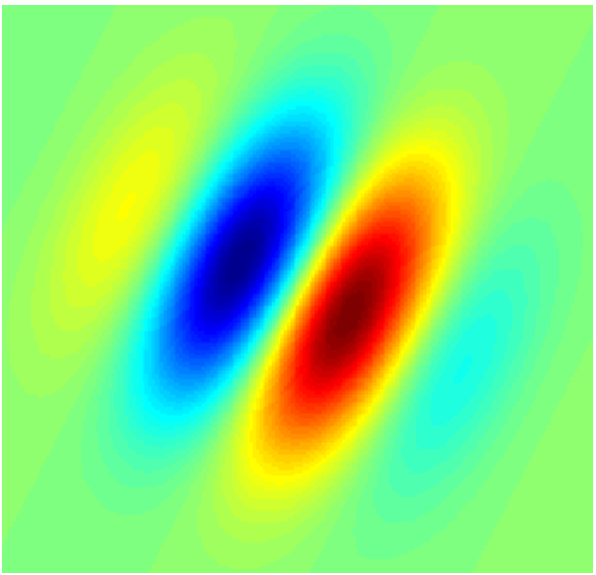
B. Bài tập thực tế

B.1 Mạng Neural Tích Chập và bài toán Nhận Dạng

a) Mô tả tổng quan về mạng nơron tích chập (CNN)

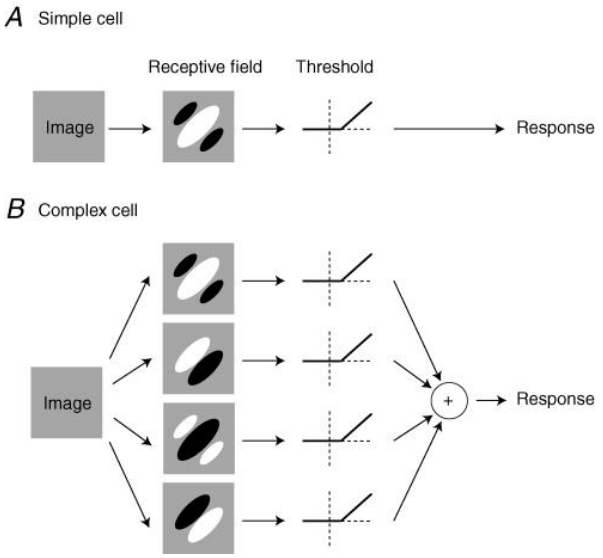
Những công trình đầu tiên vào năm 1959 của David Hubel và Torsten Wiesel đưa ra mô tả về "**simple cell**"và "**complex cell**"trong vỏ não thị giác của con người (human visual cortex) và được đề xuất sử dụng trong lĩnh vực Nhận dạng mẫu (Pattern Recognition)

Một "Simple cell"tương ứng với cạnh và đường trục trong một không gian cụ thể:



Hình 22: Simple Cell

Một "Complex cell"cũng tương ứng với cạnh và đường trục trong một không gian cụ thể nhưng nó khá khác so với "Simple cell"



Hình 23: Simple Cell

Với "simple cell"có thể chỉ ứng với một trục ngang ở cạnh dưới của một bức ảnh, còn với "complex cell"có thể ứng các trục ngang ở dưới (bottom), giữa (middle) hoặc trên cùng (top) của một bức ảnh.

Cho đến năm 1962, Hubel và Wiesel đề xuất một công trình cho thấy rằng Complex Cell đạt được bất biến không gian bằng cách "**summing** tổng hợp đầu ra của nhiều simple cell tham chiếu cùng hướng (cùng đường trục) nhưng khác nhau về vị trí tiếp nhận (có thể bottom, middle hoặc top). Khái niệm về một bộ nhận diện phức tạp có thể được tổng hợp từ nhiều bộ nhận diện đơn giản được phát hiện trong **hệ thống thị giác con người (human visual system)** và trở thành khái niệm cơ sở cho những **mô hình Convolution Neural Networks**.

Trong những năm 1980, Dr. Kunihiko Fukushima dựa trên ý tưởng công trình Simple cell và Complex cell của Hubel và Wiesel, đề xuất mô hình tên là "**neocognition**"trong bài báo "**Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position**". Mô hình bao gồm hai thuật ngữ quan trọng "S-cells"và "C-cells"định nghĩa toán tử toán tử. S-cells được đặt ở tầng đầu tiên của mô hình và được liên kết với C-cell nằm ở tầng hai của mô hình. Nhờ vào đây, người ta có thể biến nó thành mô hình tính toán cho Nhận dạng mẫu thị giác (Visual Pattern Recognition) với ý tưởng "**simple-to-complex**".

Công trình đầu tiên đặt nền móng cho Convolution Neural Network hiện đai xuất hiện vào những năm 1990, bởi Giáo sư Yann LeCun và đồng nghiệp công bố trong bài báo của họ "**Gradient-Based Learning Applied to Document Recognition**", và đến nay đã có hơn 38019 lượt citation cho bài báo này. Xuất phát từ ý tưởng của mô hình neocognition, công trình của giáo sư trình bày một mô hình CNN tổng hợp những đặc trưng đơn vào những đặc trưng phức tạp hơn có thể được sử dụng cho nhận dạng chữ viết tay (handwritten character recognition)

Một cách tổng quát, hầu hết các phương pháp dựa trên Neural Networks thường là những phương pháp có giám sát (supervised methods). Không chỉ các bài toán dự đoán hồi quy (regression) hay bài toán phân lớp (classification), nhưng có thể là không giám sát, do đó, tùy vào yêu cầu bài toán, với bài toán phân loại, chúng ta cần dữ liệu huấn luyện được gán nhãn trước, quá trình huấn luyện là giám sát, nhưng với bài toán gom cụ, ví dụ như gom cụm hình ảnh dựa trên độ tương đồng, thì các Neural Network đóng vai trò như bộ rút trích và được kết hợp với một số phương pháp không giám sát như K-means, ...

Với Convolution Neural Networks được sử dụng khá rộng rãi trong hầu hết các bài toán như phân lớp hình ảnh (Image Classification), nhận dạng vật thể (Object Recognition), các bài toán nhận dạng sinh trắc học (Biometric Recognition) như Nhận dạng khôn mặt, Nhận dạng vân tay, Nhận dạng dáng đi, Nhận dạng mẫu mắt, ... ngay cả Nhận dạng giọng nói/ người nói, nó chỉ dùng trong giai đoạn rút trích đặc trưng từ bức ảnh, tổng hợp và chồng chéo các khu vực nhỏ trên bức ảnh.

b) Các bài toán nhận dạng có thể được giải quyết bằng mạng nơron tích chập (CNN)

Một số bài toán nhận dạng có thể được giải quyết bằng mạng nơron tích chập (CNN)

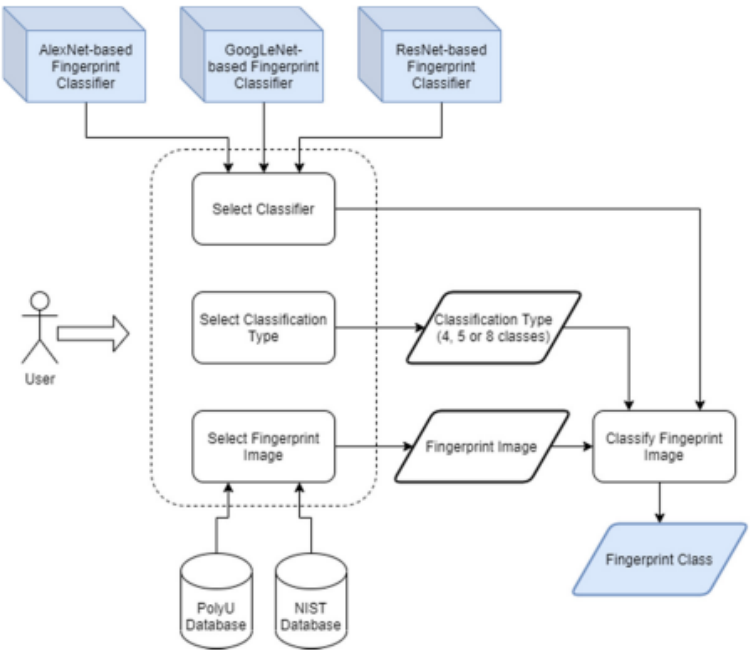
Bài toán 1: Nhận dạng vân tay dựa trên CNN

1) **Phát biểu bài toán**

- Đầu vào: Hình ảnh vân tay
- Đầu ra: Lớp của vân tay

2) **Phương pháp thực hiện**

Minh họa phương pháp được sử dụng ở [4]



Hình 24: Hệ thống nhận dạng vân tay

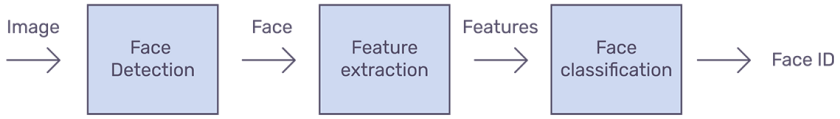
CNNs đóng vai trò như bộ phân lớp, dựa trên mạng học đã được huấn luyện từ trước với rất nhiều tham số như AlexNet, GoogleNet, ResNet.

Bài toán 2: Nhận dạng khuôn mặt dựa trên CNN

1) **Phát biểu bài toán**

- Đầu vào: Dữ liệu hình ảnh khuôn mặt
- Đầu ra: Lớp của hình ảnh khuôn mặt

2) **Phương pháp thực hiện**



Hình 25: Hệ thống nhận dạng khuôn mặt

- Face detection: Nhận diện một hay nhiều khuôn mặt trong một bức ảnh
- Feature extraction: Rút trích đặc trưng quan trọng từ một bức ảnh khuôn mặt
- Face classification: Phân lớp khuôn mặt dựa trên những đặc trưng được rút trích

Tham khảo từ: https://arsfutura.com/magazine/face-recognition-with-facenet-and-mtcnn/

Face Recognition Framework bao gồm những thành phần như MTCNN cho face detection, FaceNet cho việc phát sinh face embedding và dùng Softmax như một bộ phân lớp.

Dùng FaceNet để học:

- Chọn ngẫu nhiễn một anchor image
- Chọm ngẫu nhiên một ảnh cùng một người với ảnh anchor
- Chọn ngẫu nhiên một ảnh khác người với ảnh anchor
- Hiệu chỉnh tham số FaceNet để ảnh với ảnh anchor hơn là ảnh khác với anchor

Bài toán 3: Nhận dạng người nói dựa trên CNN

1) **Phát biểu bài toán**

Bài toán nhận dạng người nói thường được chia thành hai tác vụ con, gồm: xác minh người nói (speaker verification) và định danh người nói (speaker identification). Xét bài toán con xác minh người nói (speaker verification)

- Đầu vào: dữ liệu âm thanh giọng nói
- Đầu ra: Đồng ý (Accept)/ Từ chối (Deny)

2) **Phương pháp thực hiện**

Minh họa phương pháp được sử dụng ở [2]

Trong giai đoạn phát triển (development stage), lời nói của các người nói được sử dụng để tạo ra một mô hình nền cho việc đại diện người nói. Sử dụng kiến trúc DNN như một bộ rút trích đặc trưng người nói mạnh mẽ.

Trong giai đoạn đăng ký (Enrollment stage), một mô hình phân biệt sẽ được tạo cho mỗi định danh người nói. Những lời nói của người nói sẽ được sử dụng để phát sinh mô hình người nói. Trong DNN, lời nói của người nói sẽ là đầu vào của mô hình được tạo ra ở giai đoạn trước đó, và đầu ra được tích hợp với một số phương pháp để cho ra một mô hình người nói.

Trong giai đoạn đánh giá (Evaluation stage), những lời nói kiểm gia sẽ được đưa vào bộ rút trích đại diện người nói, truy vấn kiểm tra mẫu sẽ được so sánh với tất cả mô hình người nói bằng cách sử dụng một hàm tính điểm (score function) nào đó và điểm cao nhất được chọn là người nói được dự đó của mô hình.

CNN đóng vai trò như một bộ rút trích đặc trưng từ dữ liệu đầu vào, là những sóng âm thanh thô, được tinh chỉnh và dùng kỹ thuật cửa sổ (ví dụ hamming windows) trong miền tần số, làm CNN có thể hoạt động tương tự như xử lý với ảnh số.

c) Mô tả tư tưởng chủ đạo và các bước chính trong giải thuật của mạng nơron tích chập (CNN)

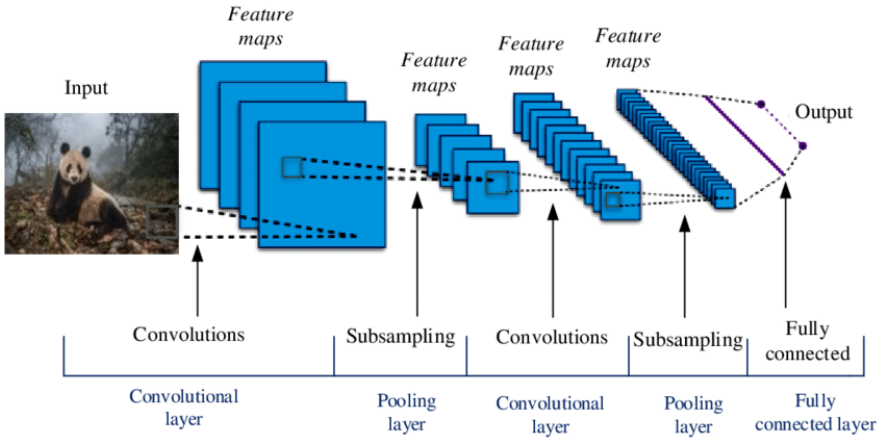
Mạng neural Tích Chập (hay được biết là Convolutional Neural Networks - CNNs) được hình thành từ ba ý tưởng chính:

- Tiếp nhận khu vực cục bộ (local receptive fields)
- Chia sẻ trọng số (Shared weights)
- Tổng hợp (Pooling)

Hai tính chất đặc trưng của CNNs:

- Những mẫu (patterns) mà CNNs học là bất biến với phép tịnh tiến
- CNNs có thể học những chiều phân cấp của mẫu

Kiến trúc cơ bản của CNN



Hình 26: Kiến trúc cơ bản mạng CNNs

Lớp Convolutional - Convolutional Layer Trong Xử lý ảnh số và Video số (Digital Image and Video Processing), một bức ảnh đầu vào được tích chập với một nhân (kernel) để tạo ra một ảnh đầu ra (output image)

- Input Image (Input feature map)
- Output Image (Output feature map)
- Filter (Bộ lọc) gồm ba tham số:
  - Kích thước nhân (kernel) F cho biết phạm vi không gian của kernel
  - Stride S cho biết khoảng cách giữa hai vị trí liên tiếp nhau của nhân (kernel)
  - Số lượng bộ đệm không (zero padding) P cho biết dố lượng các số không được thêm vào từ vị trí bắt đầu đến vị trí kết thúc của một trục

Ta có thể sử dụng hai toán tử cho việc tính tích chập gồm Convolution Operator và Cross-correlation operator

- Toán tử Convolution được sử dụng nhiều trong Xử lý ảnh số

$$\text{Output}(x,y) = (K * \text{Input})(x,y) = \sum_m \sum_n \text{Input}(x-m, y-m)K(m,n) \tag{8}$$

- Toán tử Cross-correlation được sử dụng trong mạng CNN

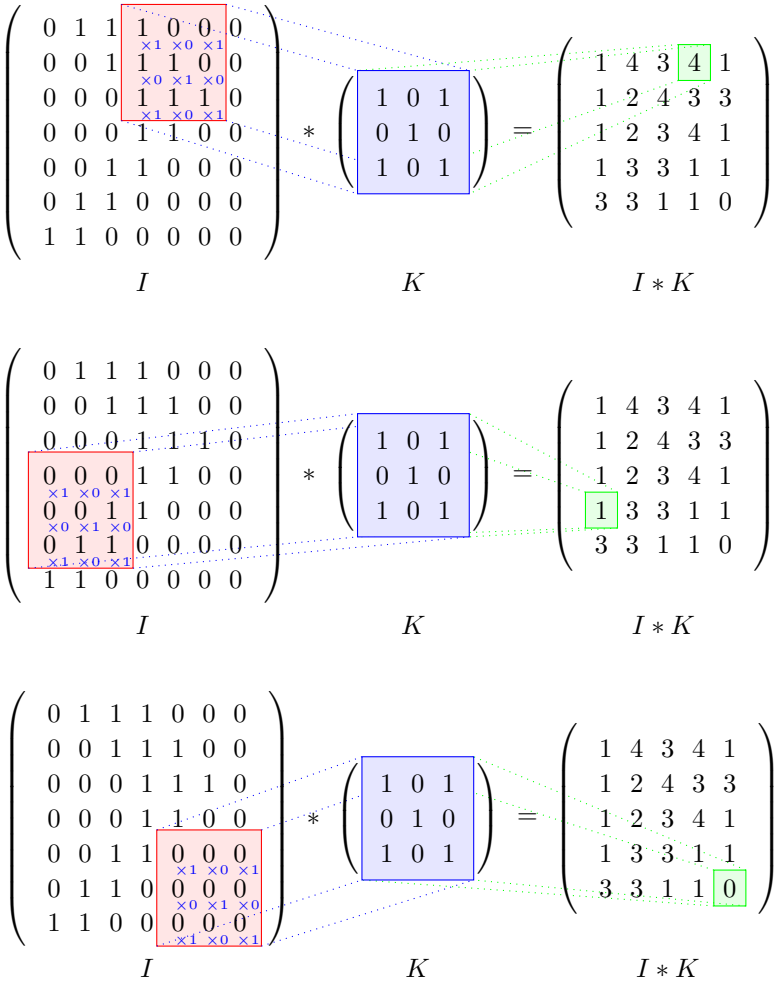
$$\text{Output}(x,y) = (K * \text{Input})(x,y) = \sum_m \sum_n \text{Input}(x+m, y+m) K(m,n) \tag{9}$$

Ví dụ:

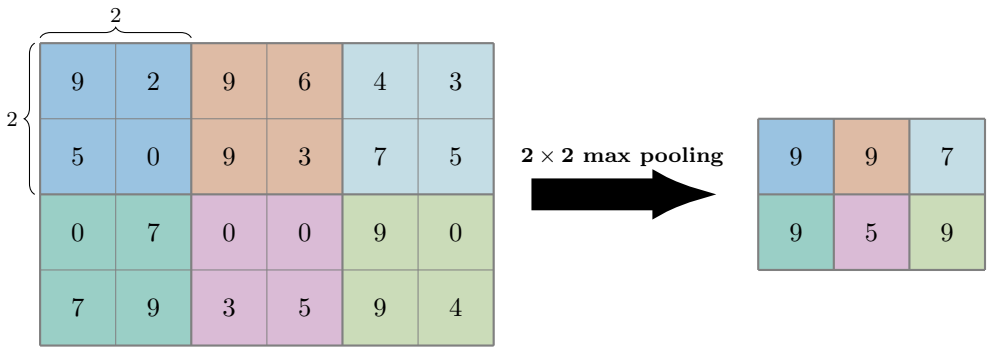
Ảnh đầu vào (Input image) có kích thước 7×7

Ảnh đầu ra (Output image) có kích thước 5×5

Nhân (Kernel) có kích thước $F = 3$, $S = 1$, $P = 0$

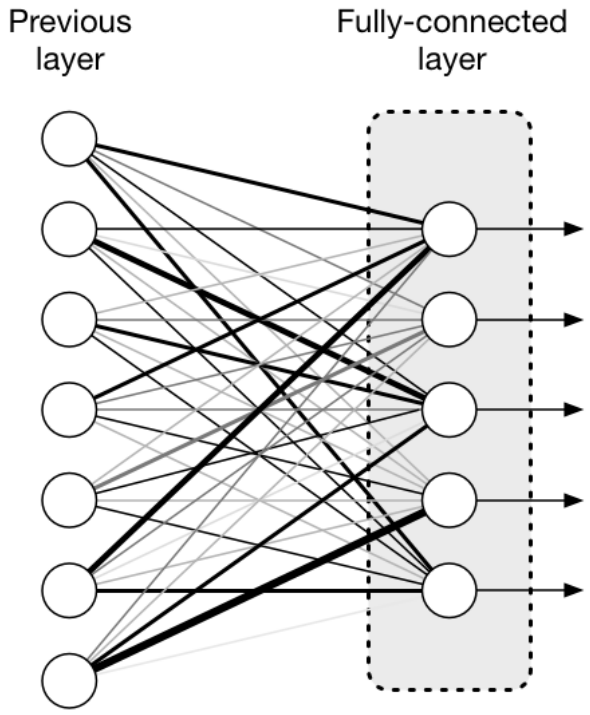


Lớp tổng hợp - Pooling layer nhiệm vụ chính của lớp này là sub-sampling các feature map, tác nó thành những feature map nhỏ hơn



Activation Function (non-linearity) ánh xạ đầu vào thành đầu ra, được coi là hàm cốt lõi (core function) của tất cả các hàm kích hoạt của tất cả các loại mạng Neural. Một hàm kích hoạt thường gặp như: Sigmoid, Tanh, ReLU, Leaky ReLU, Noisy ReLU, Parametric Linear Units

Lớp Fully-Connected Layers Fully connected layers còn được gọi là affine layers, tạo ra suy luận mức cao (high-level reasoning) trong DNN. Những neuron trong một fully connected layer kết nối đầy đủ với tất cả các kích hoạt trong layer trước đó.



Hình 27: Kiến trúc cơ bản mạng CNNs

Loss Function - Hàm mất mát là những hàm dùng để tính toán độ lỗi trong quá trình huấn luyện, nó tính toán độ lỗi giữa giá trị đúng (actual values) với giá trị dự đoán (predicted values) để tối ưu việc học trong mạng CNN. Một số hàm mất mát thường gặp như: Cross-Entropy hay Softmax Loss Function, Euclidean Loss Function, Hinge Loss Function

d) Đánh giá ưu điểm và khuyết điểm của mạng nơron tích chập (CNN) về nhiều phương diện

Ưu điểm của mạng nơron tích chập (CNN)

- CNN chia sẻ trọng số đặc trưng, mà làm giảm số lượng tham số huấn luyến có thể có và nó giúp mạng học tăng cường phát sinh và tránh overfitting
- Những tầng rút trích đặc trưng (feature extraction layers) và tầng phân lớp (classifcation layer) học một cách đồng thời vì đầu ra của mô hình vừa được tổ chức cao vừa có độ tin cậy cao vào các tính năng được trích xuất.
- Cài đặt mạng ở mức độ quy mô một cách dễ dàng với CNN hơn những mạng Neural khác.

Nhược điểm

- Cần nhiều sức mạnh tính toán, ví dụ như sử dụng GPU, TPU hay các máy chủ có cấu hình mạnh
- Hiệu năng phụ thuộc vào dữ liệu đầu vào, liệu nó có sạch hay không, có đủ nhiều hay không, có cân bằng hay không

- Quá nặng, việc nhúng vào các phần cứng bị hạn chế tài nguyên là một thách thức lớn.

B.2 YOLOX: Cải thiện YOLO (You Only Look Once)

a) Mô tả tổng quan về giải pháp nhận dạng được chọn

Với mục đích cải thiện YOLO, YOLOX là một thuật toán có khả năng phát hiện và xác minh các loại vật, người trong hình ảnh, từ đó xác minh trong video. Xác minh vật trong YOLO được thực hiện như 1 bài toán hồi quy và cung cấp xác suất phân lớp vật trong hình được dùng.

Thuật toán YOLO sử dụng CNN để xác minh vật theo thời gian thực. thuật toán này chỉ yêu cầu 1 lần lan tỏa trên mạng network để phát hiện vật.

Việc dự đoán vật trên hình chỉ qua 1 lần xử lí giúp cho YOLO trở thành 1 trong những thuật toán xử lí real-time tốt nhất hiện nay.

YOLO được phát triển qua thời gian với nhiều biến thể khác, YOLOX là 1 trong những biến thể đó.

Cá nhân/ tổ chức đang phát triển YOLO: Joseph Redmon, người đầu tiên và chính thức phát triển YOLO.

Cá nhân/ tập thể sử dụng tiềm năng:

- Chính phủ sử dụng cho quản lí về đường bộ.
- Các công ty bất kì để kiểm soát công viên làm việc.
- Camera giám sát trong những tòa nhà ở hay những cá nhân sử dụng bảo vệ căn nhà.

Tài liệu tham khảo đến bài báo:

- YOLO: https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/
- Link bài báo YOLO: https://arxiv.org/abs/1506.02640
- YOLOX: https://arxiv.org/abs/2107.08430
- Paper with code: https://paperswithcode.com/paper/yolox-exceeding-yolo-series-in-2021
- YOLO v5: http://www.xfkj.com.cn/EN/abstract/abstract510.shtml
- Code: https://github.com/Megvii-BaseDetection/YOLOX
- Trang chủ của Joseph Redmon: https://pjreddie.com/

b) Liệt kê các tính năng chủ đạo của giải pháp

- Tốc độ xử lí ảnh: thuật toán này cải thiện tốc độ phát hiện vật bởi nó có khả năng và sử dụng trong hệ thống thời gian thực.
- Độ chính xác cao: YOLO là 1 kĩ thuật dự đoán cho ra kết quả xác suất chính xác cao với tỉ lệ sai thấp.
- Khả năng học: thuật toán có khả năng học hoàn hảo cho phép nó học đại diện vật và áp dụng xác minh các vật tương đương.

Tài liệu tham khảo:

- https://arxiv.org/pdf/2107.08430.pdf (YOLOX)
- https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/ (YOLO)
- https://arxiv.org/ftp/arxiv/papers/2012/2012.07630.pdf (decoupled head)
- https://openaccess.thecvf.com/content_CVPR_2020/papers/Zhang_Bridging_the_Gap_Between_Anchor-Based_and_Anchor-Free_Detection_via_Adaptive_CVPR_2020_paper.pdf (Anchor-free)

Thuật toán trong YOLO:
Residual blocks:

- Bức ảnh được quét sẽ chia thành nhiều ô nhỏ, mạng lưới này có không gian từng lưới là S x S. như được thể hiện qua hình dưới đây.

Từ đó , xác minh đối tượng trên từng khung S x S. nhóm ô sẽ được cố định để tìm vật, qua đó xác minh danh tính của nhóm ô.

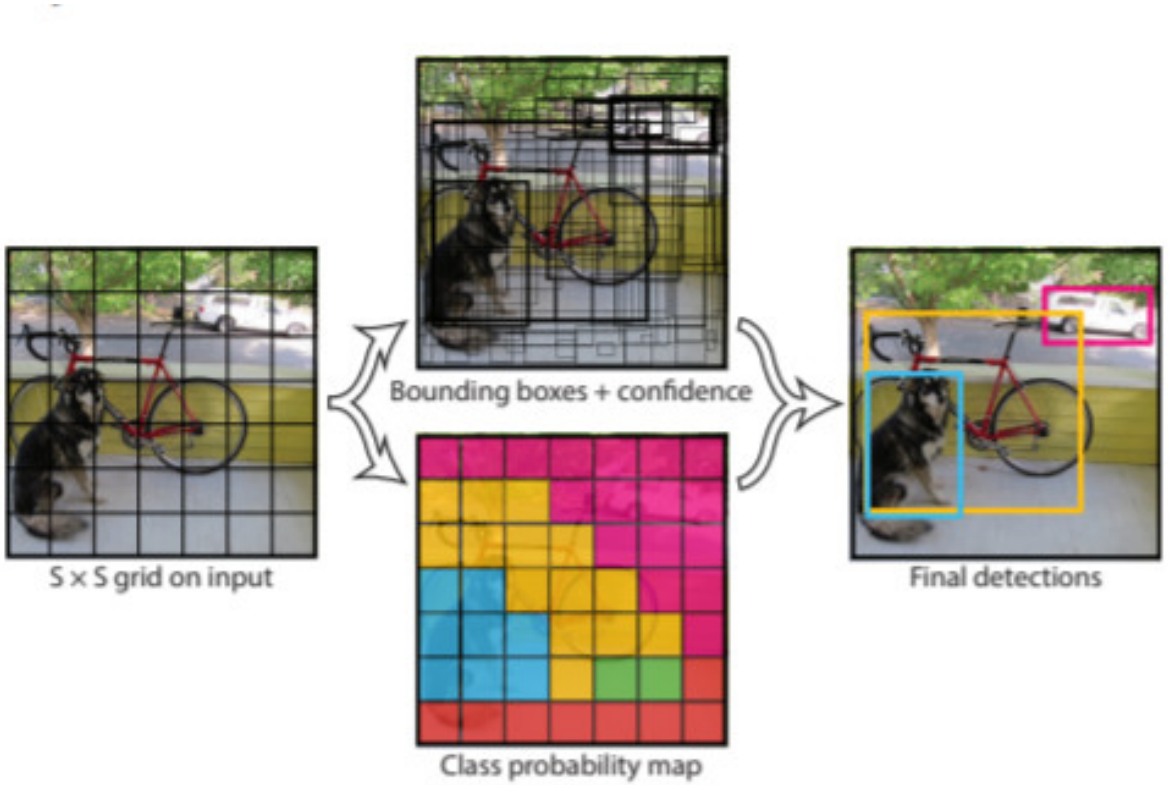
Bounding box regression:

- Các ô sau khi được nối liền sẽ trở thành 1 khối ô với đặc tính nhất định.
  - Độ rộng.
  - Độ cao
  - Lớp.
  - Tọa độ tâm.

  Khối ô sẽ được qua xử lí hồi quy để dự đoán các số liệu trên, từ đó xác định được vật (lớp).

Intersection over union (IOU):
IOU là một hiện tượng xảy ra khi các ô ở lưới bị overlap. YOLO sử dụng IOU để tạo ra 1 box ảo, box ảo này được tạo ra với mục đích làm cho việc dự đoán vật trong mỗi ô nhỏ trở nên dễ dàng hơn.

Kết hợp 3 phương pháp lại với nhau.



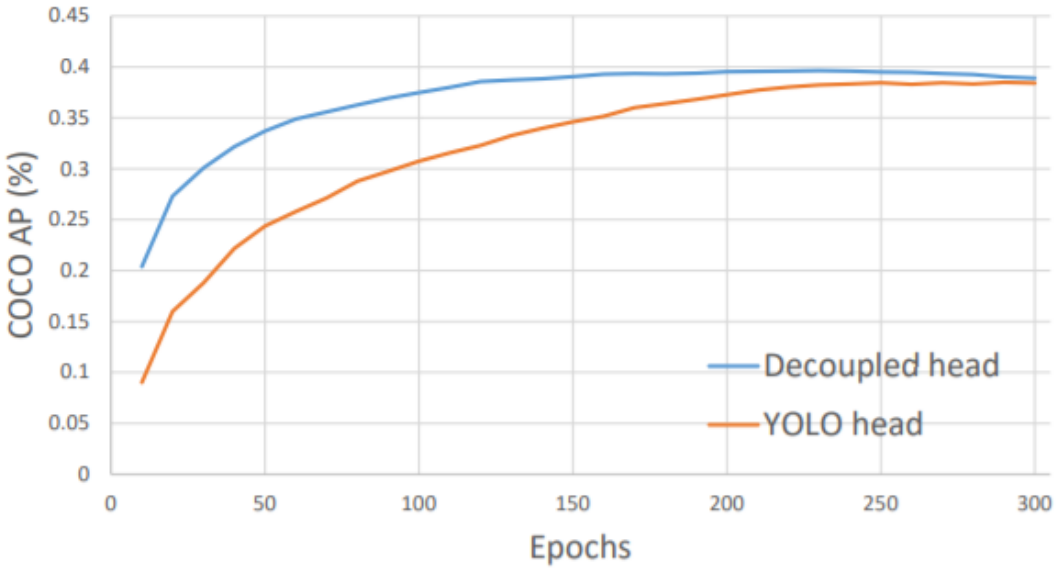
Đầu tiên ảnh được chia thành 1 lưới những ô S x S. trên mỗi ô đó thực hiện bounding box và dự đoán class của vật, với tỉ lệ chính xác được đặt trên từng ô. Tiếp theo, nhờ vào IOU, những ô này sẽ được quyết định tỉ lệ của vật khi cắt bỏ hoặc thêm vào những grid thiếu/ thừa bởi IOU. Giảm thiểu những ô thừa sẽ tăng tỉ lệ dự đoán chính xác cho vật cần xác minh.

Qua demo của bản thân và kết quả :

Việc dự đoán phụ thuộc vào nhiều yếu tố như: độ sáng của ảnh, tỉ lệ độ lớn vật trong khung hình hay kết cấu trên vật tạo ra grid.

YOLOX được phát triển từ YOLOv3 với công cụ Darknet53. YOLOX có các thay đổi so với YOLO về số liệu là chủ yếu, thay đổi về learning rate, IOU loss, BCE loss, ... Sử dụng decouple head hay vì couple head bởi:

- Thay thế decouple head vào YOLO làm cho tốc độ hội tụ khi training được tăng lên đáng kể.
- Decoupled head là cần thiết đối với phiên bản YOLO.



Sử dụng Anchor-free thay cho Anchor-based: Anchor-free xác định các điểm keypoints cố định và nới rộng vật theo các keypoint đó.

Hoặc xác định các ô cấu tạo vật, và bám theo điểm trung tâm của khối vật đó.

Thay vì sử dụng anchor based và trải qua nhiều bước để xử lí được vật xác minh, anchor-free thể hiện tương tác cao hơn và tốc độ xử lí nhanh hơn, là điều cần nhắm đến khi sử dụng trong ứng dụng real-time.

c) Đánh giá ưu điểm và khuyết điểm của giải pháp

YOLOX tập trung vào việc cải thiện thời gian xử lí để phù hợp với việc thực hiện real-time video. Bỏ qua các yếu tố như, độ chính xác, xử lí hình ảnh, xác định khu vực vật.

Những ưu điểm:

- Có độ chính xác tương đối cao, tốc độ xử lí hoàn hảo để sử dụng cho real-time video.
- Được phát triển với nhiều phiên bản phù hợp với yêu cầu đa dạng của người dùng.
- Là thuật toán mới được phát triển gần đây, sử dụng những phương pháp khá mới.
- Sẽ tiếp tục được phát triển nhiều hơn ở tương lai cũng như hiện tại, bởi sự đa dạng về phiên bản cũng như các bước thực hiện.

Những khuyết điểm:

- Độ chính xác không được tốt, chưa thực sự hoàn hảo

d) Nhận diện ít nhất hai giải pháp khác

Một số giải pháp khác

R-CNN, Faster R-CNN, Mask R-CNN:

Được phát triển bởi các nhà nghiên cứu làm việc tại Google, các phương pháp sử dụng CNN được phát triển đầu tiên và đang ngày càng được cải thiện qua từng ngày.

Input sẽ được đưa vào và cắt các mảnh được xem là vật cần xác định. Thực hiện lặp lại để ra được vật nhỏ trong vật lớn, từ đó đảm bảo được lấy được khung hình vừa vặn nhất so với vật cần tìm, và đảm bảo tính đúng đắn đối với các vật có trong hình.

Tương tự như YOLO, CNN cũng được phát triển qua nhiều phiên bản theo từng năm, với phiên bản mới nhất hiện nay là Mask R-CNN.

CenterNet:

Sử dụng keypoint để xác định điểm tâm, từ đó giãn ô và xác định vật.

Bằng cách xem như vật chỉ là 1 điểm tâm, bộ dự đoán sử dụng keypoint để để xác định được tính chất của vật, vị trí trên mô hình 3D, định hướng vật, hay dáng của vật.

CenterNet là phương pháp khác biệt, đơn giản hơn, nhanh hơn, và chính xác hơn so với việc sử dụng lưới hay chia ô.

CenterNet đạt được khả năng đánh đổi tốc độ-độ chính xác cao nhất trên dataset COCO. Với khả năng thực hiện real-time. Đây là 1 thuật toán khá tương đồng với YOLO.

B.3 Sử dụng CNN và RNN cho một bài toán nhận dạng

Trình bày một bài toán mà tồn tại cả giải pháp sử dụng CNN (hoặc biến thể của nó) và giải pháp sử dụng RNN (hoặc biến thể của nó)

a) Phát biểu bài toán

Bài toán: Nhận dạng giọng nói

Giới thiệu: Trong nhận dạng giọng nói hay nhận dạng chữ viết tay, đầu ra sẽ là một câu, nhưng chưa hoàn chỉnh vì có các ký tự lặp lại như "heelllo", "toooo", ... hay các chữ có những khoảng trống (blank) như "he l l oo", "t o o", ... Nguyên nhân dẫn tới những hiện tượng này là giọng nói dài (các đoạn ngân nga trong các bài hát, ...), giọng bị ngắt quãng, kích thước chữ viết tay lớn, nhỏ, ...

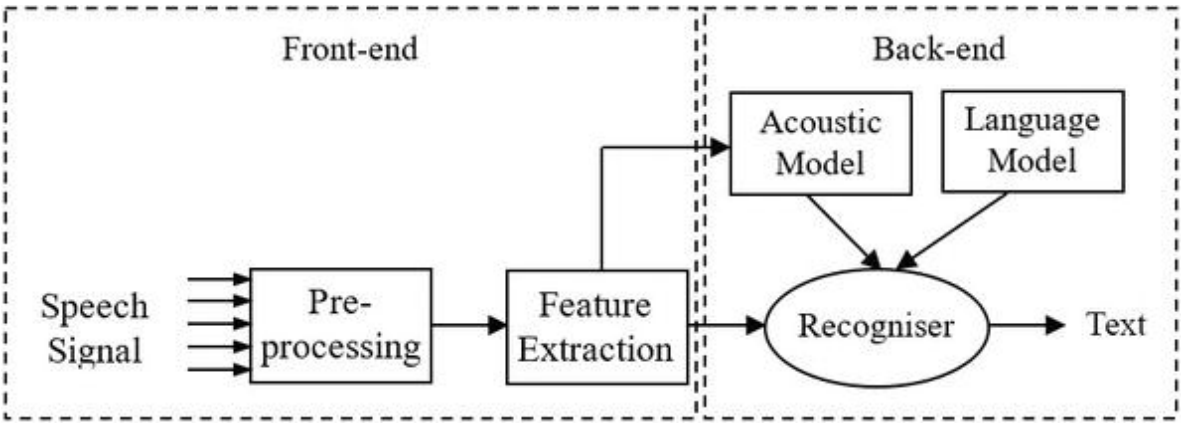
Do đó, để cho ra được một câu hoàn chỉnh thì ta cần phải căn chỉnh lại đầu ra ấy, loại bỏ các ký tự lặp lại và khoảng trống.

Bài toán này có nhiều ứng dụng thực tế như trợ giúp cho người khuyết tật về thị giác, tự động hóa, giáo dục,. . . .

Input: Dữ liệu âm thanh giọng nói thô.

Output: Đưa ra chuỗi ký tự được nói chính xác.

b) Mô tả tổng quan về giải pháp nhận dạng sử dụng mạng nơron tích chập (CNN)

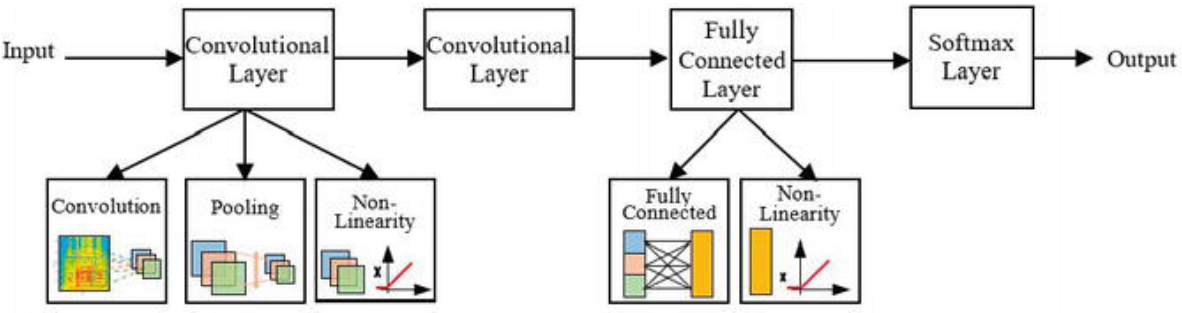


Hình 28: General framework of automatic speech recognition system

Mạng nơ ron tích chập hay convolutional neural network (CNN) là mạng dạng tiếp thuận (feedforward) trong đó thông tin chỉ đi theo một chiều từ đầu vào đến đầu ra. CNN là một deep neural network (DNN). Hiểu đơn giản, nó cũng chính là một dạng artificial neural network (ANN), một multi-layer perceptron (MLP) nhưng mang thêm 1 vài cải tiến, đó là convolution và pooling. Gồm một lớp đầu vào, các lớp ẩn và một lớp đầu ra. Các lớp ẩn thực hiện việc tích chập đầu vào mỗi lớp và đưa ra đầu ra, tương ứng với đầu vào của các lớp tiếp theo. Mỗi lớp tích chập được mô tả có ba trạng thái : tích chập, tổng hợp và phi tuyến tính.

CNN sử dụng các hàm phi tuyến để xử lý trực tiếp dữ liệu mức thấp. CNN có khả năng học các tính năng cấp cao với độ phức tạp và trừu tượng cao.

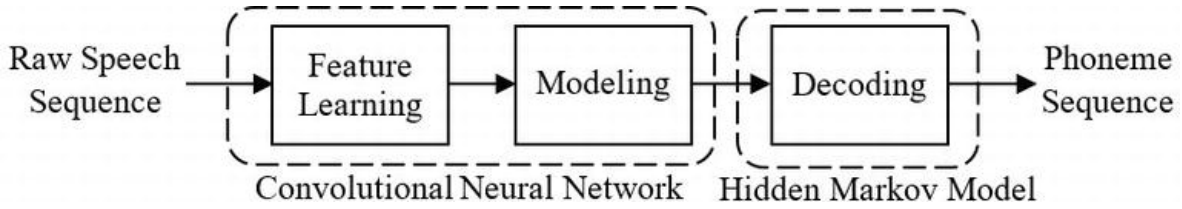
Pooling là phần quan trọng nhất của CNN giúp giảm kích thước của bản đồ đặc trưng. Pooling là một khái niệm biến biểu diễn đặc điểm chung thành thông tin có giá trị bằng cách giữ lại thông tin hữu ích và loại bỏ thông tin không quan trọng. Sự thay đổi tần số nhỏ thường gặp trong tín hiệu giọng nói được xử lý hiệu quả bằng cách sử dụng Pooling. Việc Pooling chung cũng giúp giảm phương sai phổ có trong lời nói đầu vào. Nó ánh xạ đầu vào từ p đơn vị liền kề thành đầu ra bằng cách áp dụng một hàm đặc biệt. Sau khi dùng hàm phi tuyến lên các phần tử chứa nhiều thông tin, các đặc trưng được chuyển qua lớp Pooling. Lớp này thực hiện việc lấy mẫu xuống trên các bản đồ đối tượng đến từ lớp trước và tạo ra các bản đồ đối tượng mới cô đọng. Lớp này làm giảm đáng kể kích thước và chiều của input. Nó phục vụ hai mục đích chính. Đầu tiên là giảm gần 65% kích thước của input, giảm chi phí tính toán. Ngoại ra, lớp này còn kiểm soát việc xảy ra overfitting.



Hình 29: CNN Architecture

CNN-based raw speech phoneme recognition system là một mô hình được để xuất bởi Palaz. Trong đó tín hiệu giọng nói thô đầu vào sẽ được chia thành các tín hiệu giọng nói đầu vào.

$$s_t^c = \{s_{t-c}, ..., s_t, s_{t+c}\}$$



Hình 30: CNN-based raw speech phoneme recognition systeme

Tương ứng với 2c frame, mỗi frame có cửa sổ kéo dài trong w_{in} milliseconds. Lớp tích chập đầu tiên học các đặc trưng có ích trong tín hiệu giọng nói thô. Các lớp tích chập còn lại thực hiện nhiệm vụ tiếp tục xử lý các đặc trưng thành thông tin có ích.

Sau khi xử lý tín hiệu giọng nói, CNN đánh giá xác suất có điều kiện của từng lớp Một số giai đoạn lọc có trong mạng trước giai đoạn phân loại. Giai đoạn lọc là sự kết hợp của lớp chập, lớp gộp và lớp phi tuyến. Việc huấn luyện chung của giai đoạn rút trích đặc trưng và giai đoạn phân loại được thực hiện bằng cách sử dụng thuật toán lan truyền ngược.

Cách tiếp cận của phương pháp này được mô tả như sau: Tín hiệu lời nói có bản chất là không cố định. Do đó, chúng được xử lý theo phương thức ngắn hạn. Các phương pháp trích xuất đặc trưng truyền thống thường sử dụng kích thước cửa sổ xử lý 20–40 ms. Trong phương pháp này, việc xử lý tín hiệu trong thời gian ngắn là bắt buộc. Do đó, kích thước của cửa sổ xử lý được coi là siêu tham số được xác định tự động trong quá trình huấn luyện.

Trích xuất đặc trưng có vai trò giống như một bộ lọc vì nó bao gồm: biến đổi Fourier, biến đổi cosin rời rạc, v.v. Trong các hệ thống truyền thống, lọc được áp dụng trên cả miền tần số và miền thời gian. Vì vậy, lọc cũng được xem xét trong việc xây dựng lớp tích chập trong hệ thống end-to-end. Do đó, số lượng bộ lọc và các tham số của chúng được xem như là các siêu tham số và được xác định tự động trong quá trình huấn luyện. Quá trình xử lý tín hiệu giọng nói ngắn hạn ẩn chứa thông tin theo thời gian. Trong các hệ thống truyền thống, thông tin lan truyền này được mô hình hóa bằng cách tính toán các dẫn xuất thời gian và thông tin theo ngữ cảnh. Do đó, biểu diễn trung gian được cung cấp cho bộ phân loại và được tính toán bằng cách lấy nhịp độ dài theo thời gian của tín hiệu giọng nói đầu vào. Từ đó, w_{in}, kích thước của cửa sổ nhập liệu, được coi là siêu tham số, được ước tính trong quá trình huấn luyện. Mô hình end-to-end ước tính P_i/s_t^c (đánh giá phân lớp) bằng cách xử lý tín hiệu giọng nói với các giả định tối thiểu hoặc tri thức trước đó.

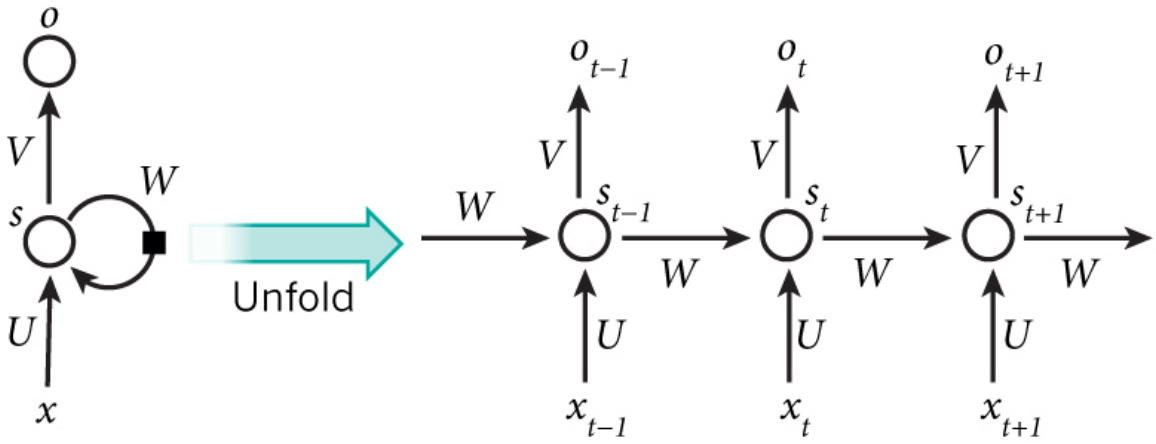
Đánh giá mô hình: Mô hình này trực tiếp học cách biểu diễn có liên quan từ tín hiệu tiếng nói theo hướng dữ liệu và tính toán xác suất có điều kiện cho mỗi lớp âm vị. Trong đó, CNN như một mô hình âm thanh bao gồm một giai đoạn đặc trưng và giai đoạn phân loại. Cả hai giai đoạn đều được huấn luyện chung. Lời nói thô được cung cấp làm đầu vào cho lớp chập đầu tiên và nó được xử lý thêm bởi một số lớp chập. Các bộ phân loại như ANN, CRF, MLP hoặc các lớp được kết nối đầy đủ sẽ tính toán xác suất có điều kiện cho mỗi lớp âm vị. Sau khi giải mã được thực hiện bằng HMM. Mô hình này cho thấy hiệu suất tương tự như các mô hình truyền thống dựa trên MFCC.

Độ chính xác cao nhưng độ phức tạp khá lớn.

c) Mô tả tổng quan về giải pháp nhận dạng sử dụng mạng hồi quy (RNN)

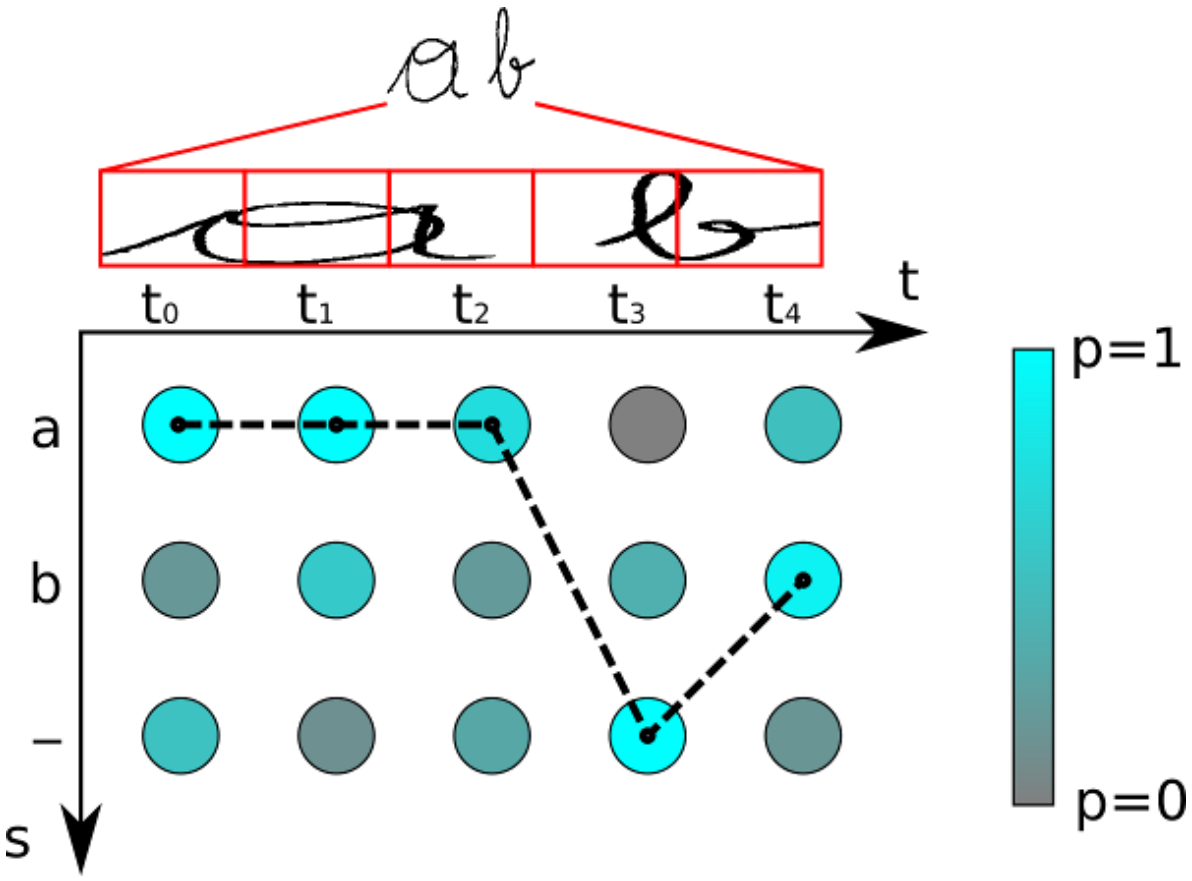
Ý tưởng chính của RNN (Recurrent Neural Network) là sử dụng chuỗi các thông tin. Trong các mạng nơ-ron truyền thống tất cả các đầu vào và cả đầu ra là độc lập với nhau. Tức là chúng không liên kết thành chuỗi với nhau. Nhưng các mô hình này không phù hợp trong rất nhiều bài toán. Ví dụ, nếu muốn đoán từ tiếp theo có thể xuất hiện trong một câu thì ta cũng cần biết các từ trước đó xuất hiện lần lượt thế nào chứ nhỉ? RNN được gọi là hồi quy (Recurrent) bởi lẽ chúng thực hiện cùng một

tác vụ cho tất cả các phần tử của một chuỗi với đầu ra phụ thuộc vào cả các phép tính trước đó. Nói cách khác, RNN có khả năng nhớ các thông tin được tính toán trước đó. Trên lý thuyết, RNN có thể sử dụng được thông tin của một văn bản rất dài, tuy nhiên thực tế thì nó chỉ có thể nhớ được một vài bước trước đó (ta cùng bàn cụ thể vấn đề này sau) mà thôi. Về cơ bản một mạng RNN có dạng như sau:



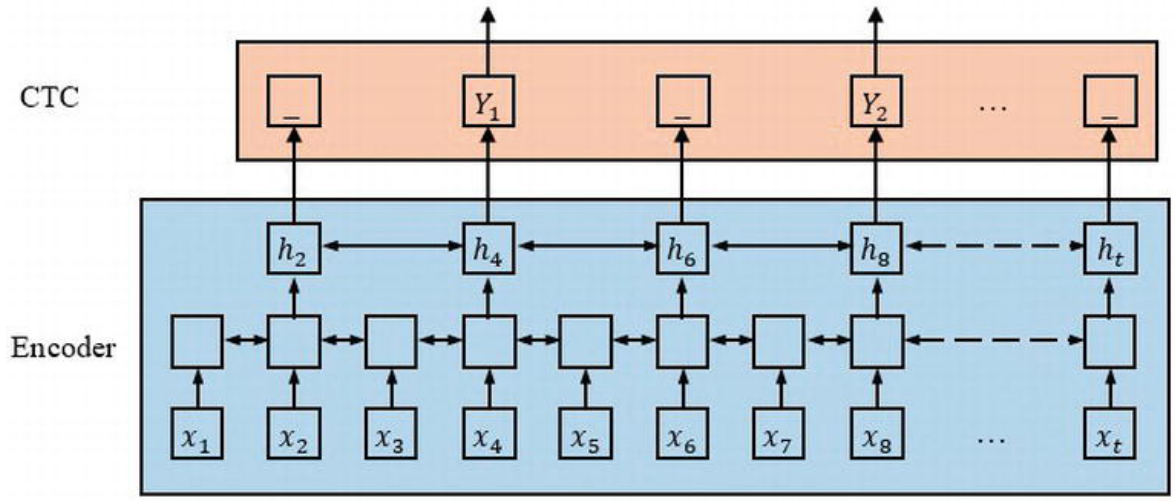
Mô hình trên mô tả phép triển khai nội dung của một RNN. Triển khai ở đây có thể hiểu đơn giản là ta vẽ ra một mạng nơ-ron chuỗi tuần tự. Ví dụ ta có một câu gồm 5 chữ "Đẹp trai lắm gái theo", thì mạng nơ-ron được triển khai sẽ gồm 5 tầng nơ-ron tương ứng với mỗi chữ một tầng.

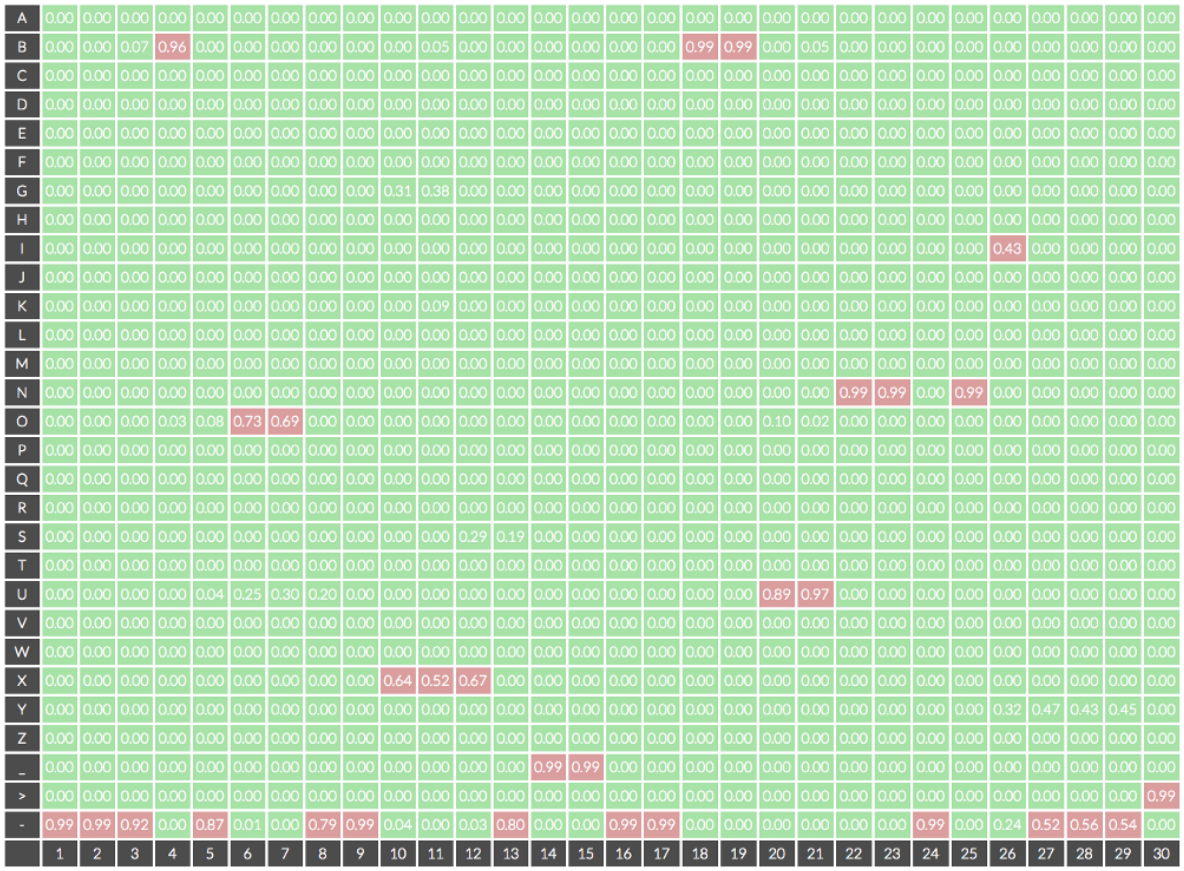
Đưa vào một chuỗi các tín hiệu âm thanh, ta có thể dự đoán được chuỗi các đoạn ký tự đi kèm với xác xuất của chúng. Ví dụ Hello Hellllooooo.



Tư tưởng chủ đạo của phương pháp này chính là tạo ra một mô hình được huấn luyện (CTC : RNN) với đầu vào là dữ liệu giọng nói thô và đầu ra là một bảng ký tự(có thể là ký tự tiếng anh, tiếng la tinh, còn nếu tiếng việt thì cần phải xác định thêm xác suất xuất hiện của dấu câu) kèm theo xác suất xuất hiện của nó theo thời gian.

Quá trình giải mã sau khi xử lý: Xác định những ký tự có khả năng xuất hiện cao nhất mỗi bước thời gian (có thể là 1s hoặc 2s), Xóa các ký tự bị trùng lặp đồng thời xóa những khoảng trống lặp lại. Phần văn bản còn lại đại diện cho đầu ra.





Quá trình huấn luyện mô hình này có thể được thực hiện theo các bước sau : tính toán xác suất của các alignments bằng cách connect các xác suất của output của các thời điểm lại với nhau và chọn alignment phù hợp nhất, tính sai số của nó và cho network học lại. Vì mô hình này có thể tính toán được hàm loss nên có thể huấn luyện dựa vào gradient descent.

Mô hình RNN được sử dụng cho phương pháp này là vì mô hình này cho đầu ra theo thời gian và mô hình chính trong phương pháp này là mô hình phân loại nào đó mang tính thời gian.

Đánh giá mô hình: Độ phức tạp đơn giản hơn nên thời gian thực hiện mô hình này ngắn hơn so với mô hình CNN ở trên nhưng đưa ra kết quả chưa được tốt. Ưu điểm hơn mô hình này chính là kết quả bám sát theo thời gian nói.

C. Nội dung phân công

STT	MSSV	Họ tên	Nội dung công việc	Hoàn thiện (%)
1	18120018	Nguyễn Hoàng Đức	A.5 Kiến thức về mạng nơron sâu (DNN), B-3. Trình bày bài toán thị giác máy tính sử dụng CNNs và RNN	100%
2	18120061	Lê Nhựt Nam	A-1. Kiến thức về Local Binary Patterns (LBP); A-2. Kiến thức về Principal Component Analysis (PCA) và Linear Discriminant Analysis (LDA); A-4. Kiến thức về mạng nơrồn cơ bản (ANN); B-1. Trình bày kỹ thuật nhận dạng	100%
3	18120167	Nguyễn Viết Dũng	A-3. Kiến thức về Support Vector Machines (SVM); B-2. Trình bày giải pháp nhận dạng trong khoảng thời gian gần đây	100%

Tài liệu

[1] Towardsdata science: An intuitive explanation of connectionist temporal classification.

[2] Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-dujaili, Y. Duan, Omran Al-Shamma, J. Santamaría, M. Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data*, 8, 2021.

[3] Yanzhang He, Tara N. Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, Qiao Liang, Deepti Bhatia, Yuan Shangguan, Bo Li, Golan Pundak, Khe Chai Sim, Tom Bagby, Shuo yiin Chang, Kanishka Rao, and Alexander Gruenstein. Streaming end-to-end speech recognition for mobile devices, 2018.

[4] Carmelo Militello, Leonardo Rundo, Salvatore Vitabile, and Vincenzo Conti. Fingerprint classification based on deep learning approaches: Experimental findings and comparisons. *Symmetry*, 13(5), 2021.

[5] Vishal Passricha and Rajesh Aggarwal. *Convolutional Neural Networks for Raw Speech Recognition.* 12 2018.

[6] Jie Wang and Zihao Li. Research on face recognition based on cnn. *IOP Conference Series: Earth and Environmental Science*, 170:032110, 07 2018.