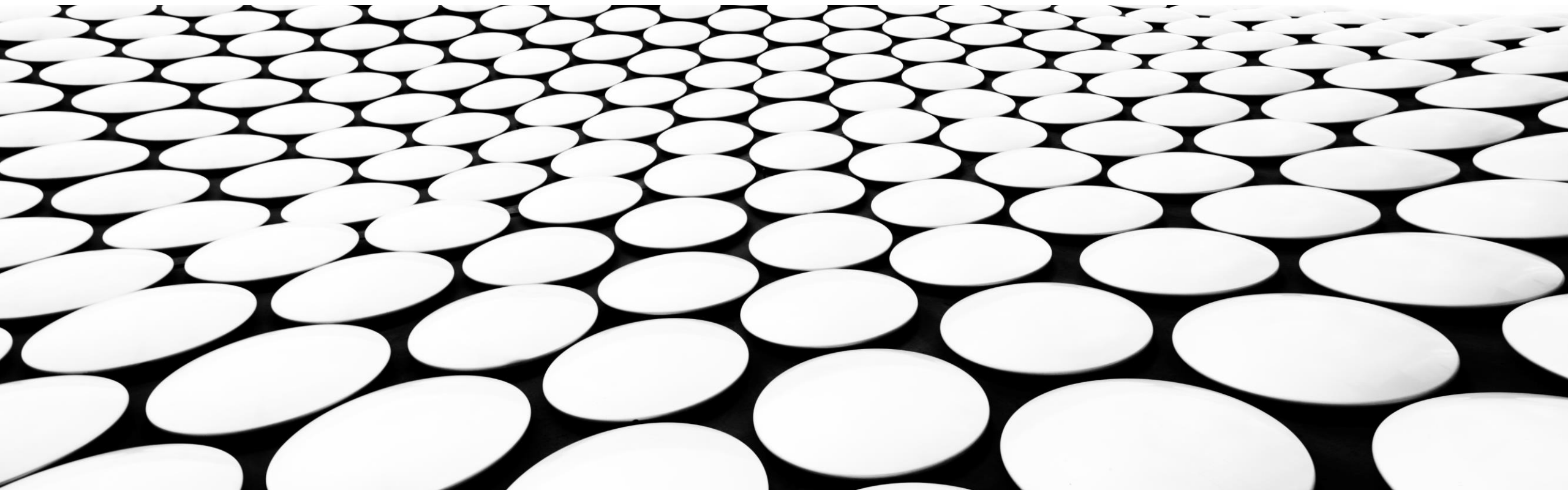

ALGORITHMS

SUPERVISED LEARNING & UNSUPERVISED LEARNING



CÁC THUẬT
TOÁN NHÓM
NGHIÊN CỨU:

- . *K nearest neighbor*
- . *K-means for clustering*
- . *Singular Value
Decomposition*
- . *Convolutional neural network*

PHÂN BIỆT HỌC CÓ GIÁM SÁT & HỌC KHÔNG GIÁM SÁT

Supervised Learning	Unsupervised Learning
Dữ liệu để huấn luyện mô hình	
Dữ liệu đầu vào được gán nhãn.	Dữ liệu đầu vào không được gán nhãn.
Chức năng chính	
Classification, Regression	Clustering
Độ chính xác của kết quả	
Chính xác và đáng tin cậy hơn.	Không chính xác và không đáng tin cậy.

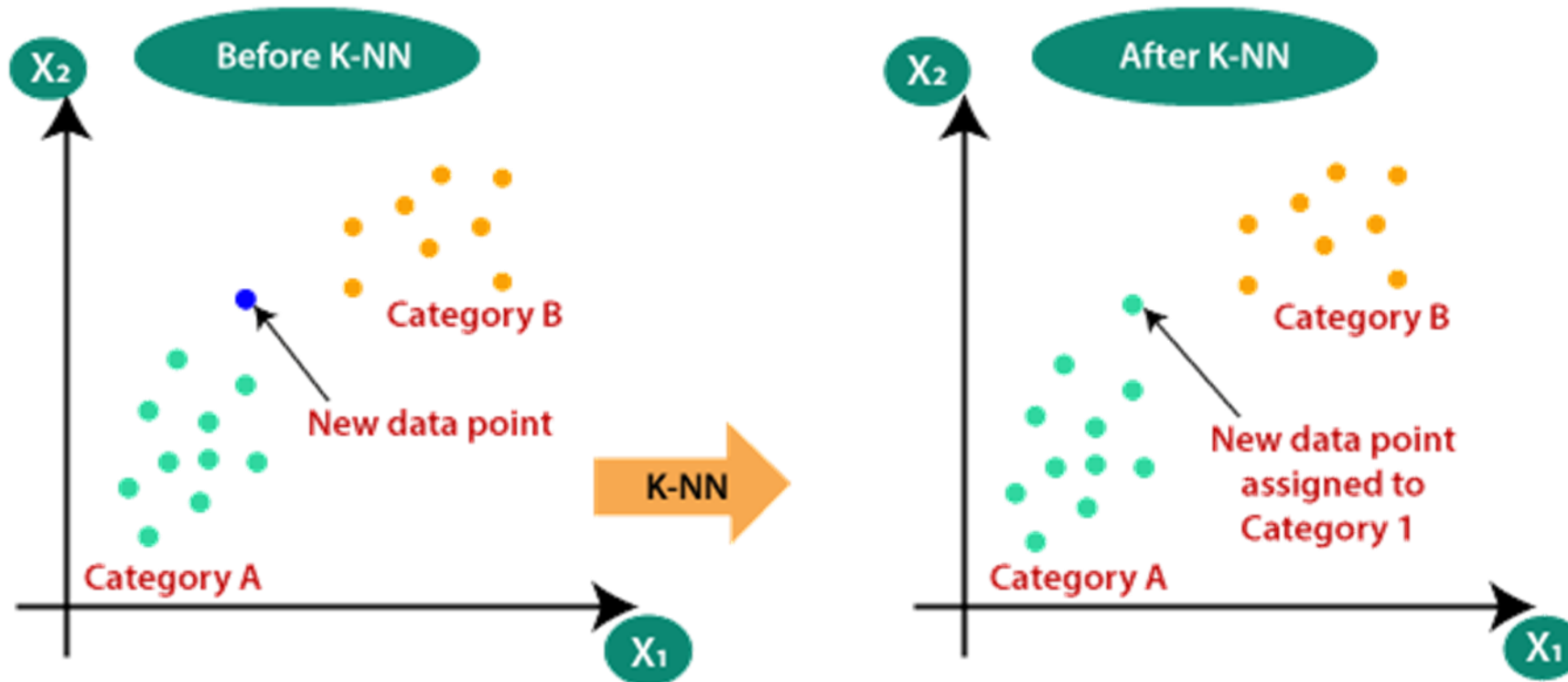
I. K-NEAREST NEIGHBOR:

1. Giới thiệu bài toán:

- KNN là một kĩ thuật máy học phân loại được sử dụng rộng rãi.
- KNN dựa vào số lượng những người hàng xóm gần nhất và nó được sử dụng cho các bài toán hồi quy .

I. K-NEAREST NEIGHBOR:

2. Cách thức hoạt động của KNN:



I. K-NEAREST NEIGHBOR:

2. Cách thức hoạt động của KNN:

- + Bước 1: Chọn số lượng K neighbors.
- + Bước 2: Lấy K neighbors của điểm data mới theo phương pháp *tính khoảng cách của hai điểm*.
- + Bước 3: Trong K neighbors đó , đếm số lượng data của mỗi cụm.
- + Bước 4: Cập nhật cho cái điểm data mới vào cụm mà có số lượng neighbors vừa đếm ở bước 3 là nhiều nhất.

I. K-NEAREST NEIGHBOR:

3. Công thức tính khoảng cách:

a) Euclidean distance:

- Là độ dài khoảng cách nối 2 điểm trong không gian
- Công thức :

$$d(x, y) = \sqrt{\sum_{i=1}^n (y_i - x_i)^2}$$

I. K-NEAREST NEIGHBOR:

3. Công thức tính khoảng cách:

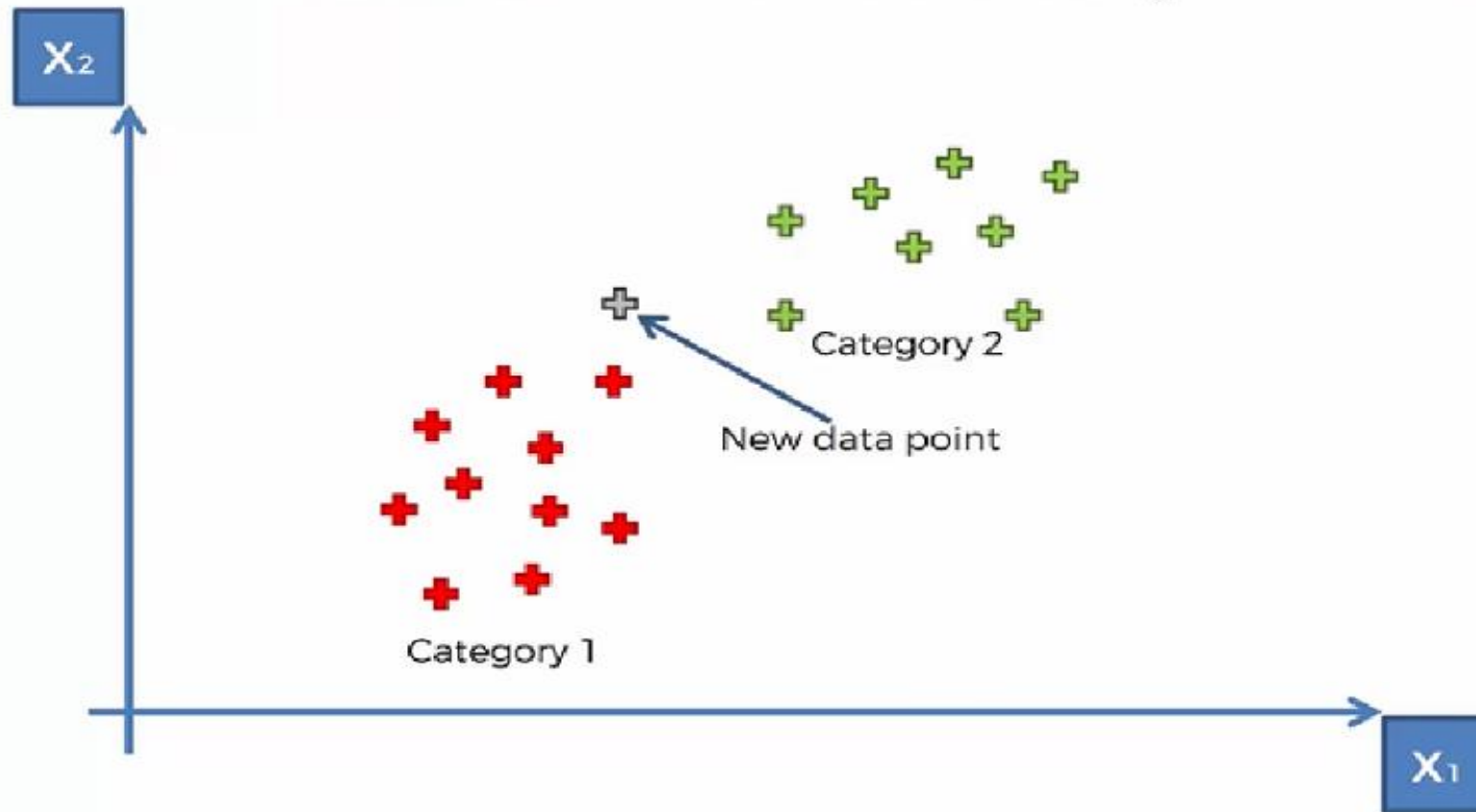
b) Manhattan distance

- Là khoảng cách L1, một dạng khoảng cách giữa hai điểm trong không gian Euclid với hệ tọa độ Descartes
- Công thức :

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

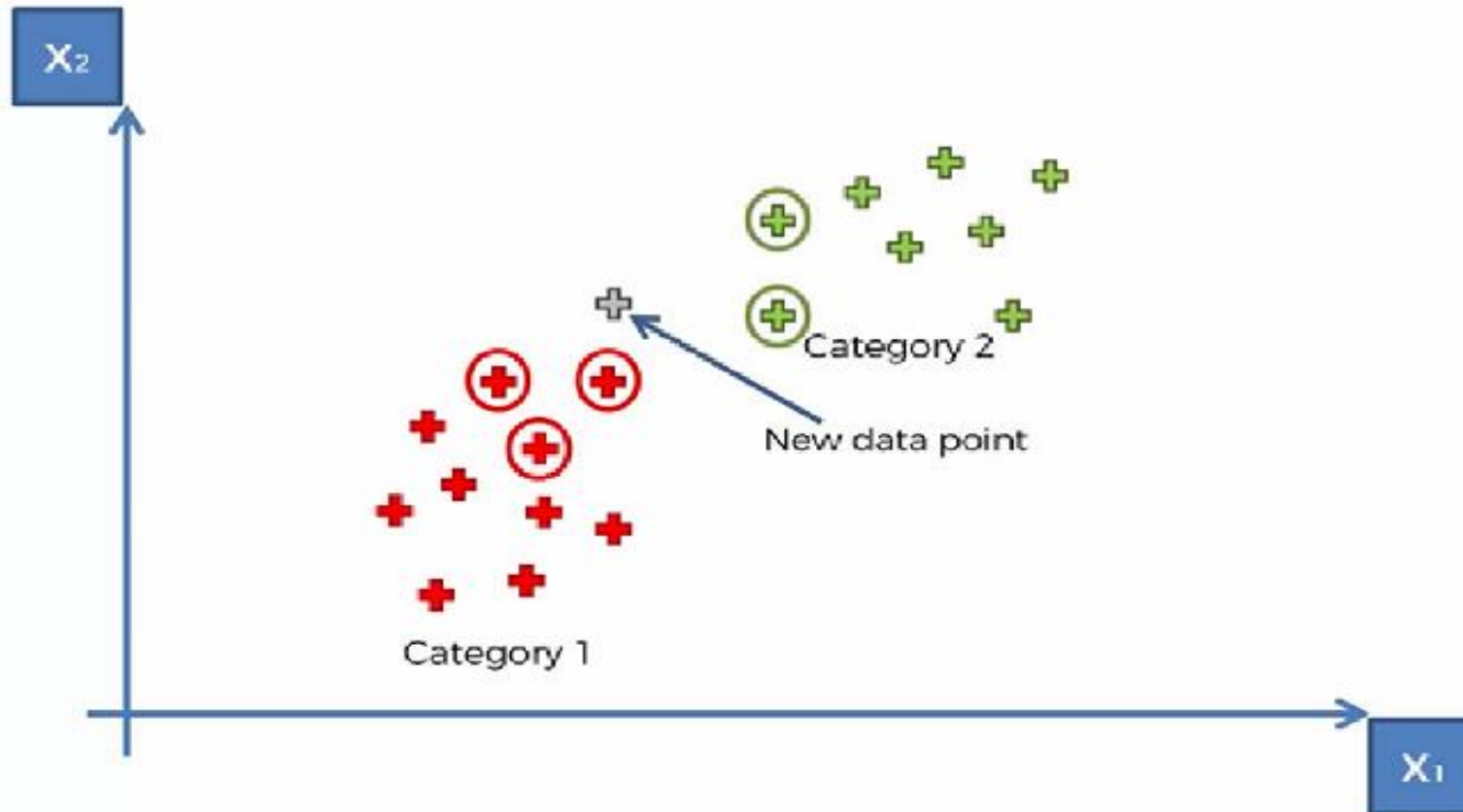
I. K-NEAREST NEIGHBOR:

STEP 1: Choose the number K of neighbors: $K = 5$



I. K-NEAREST NEIGHBOR:

STEP 2: Take the $K = 5$ nearest neighbors of the new data point, according to the Euclidean distance



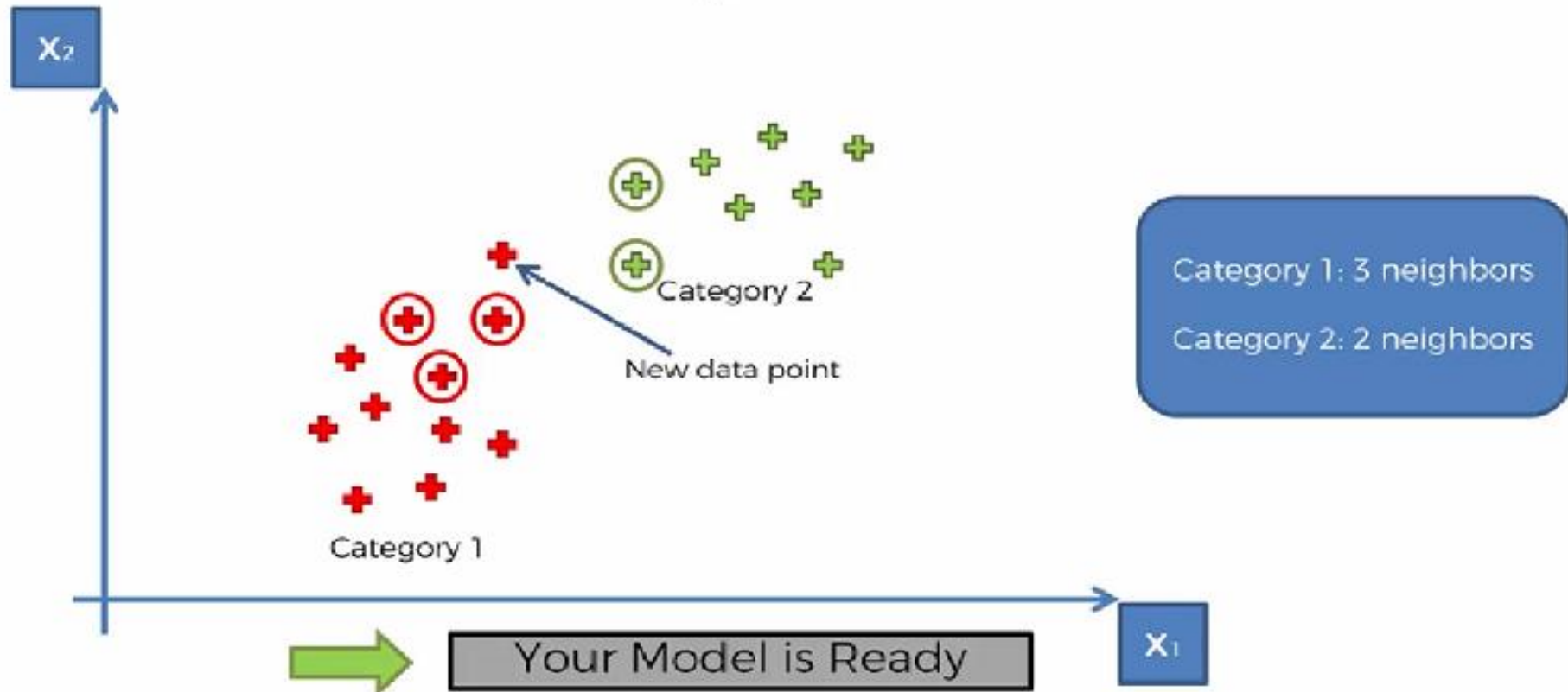
I. K-NEAREST NEIGHBOR:

STEP 3: Among these K neighbors, count the number of data points in each category



I. K-NEAREST NEIGHBOR:

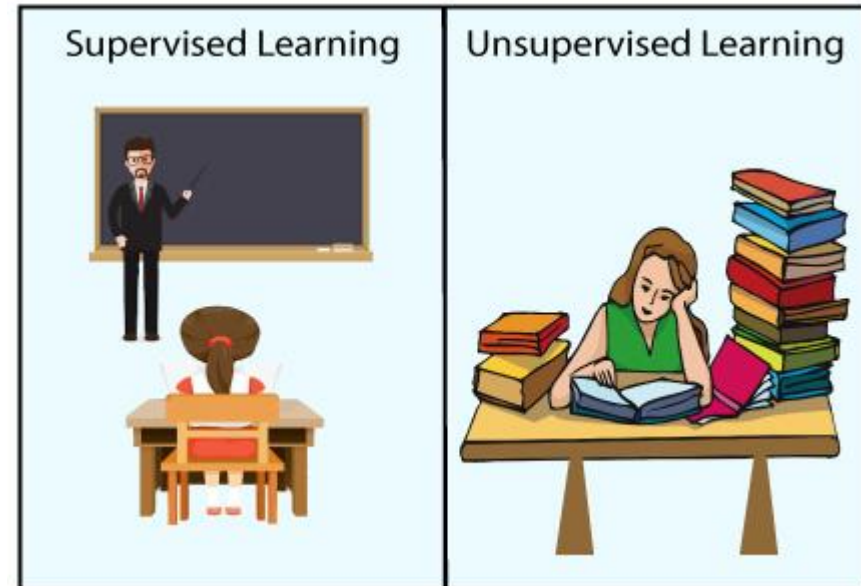
STEP 4: Assign the new data point to the category where you counted the most neighbors



II. K MEANS CLUSTERING:

1. Giới thiệu bài toán:

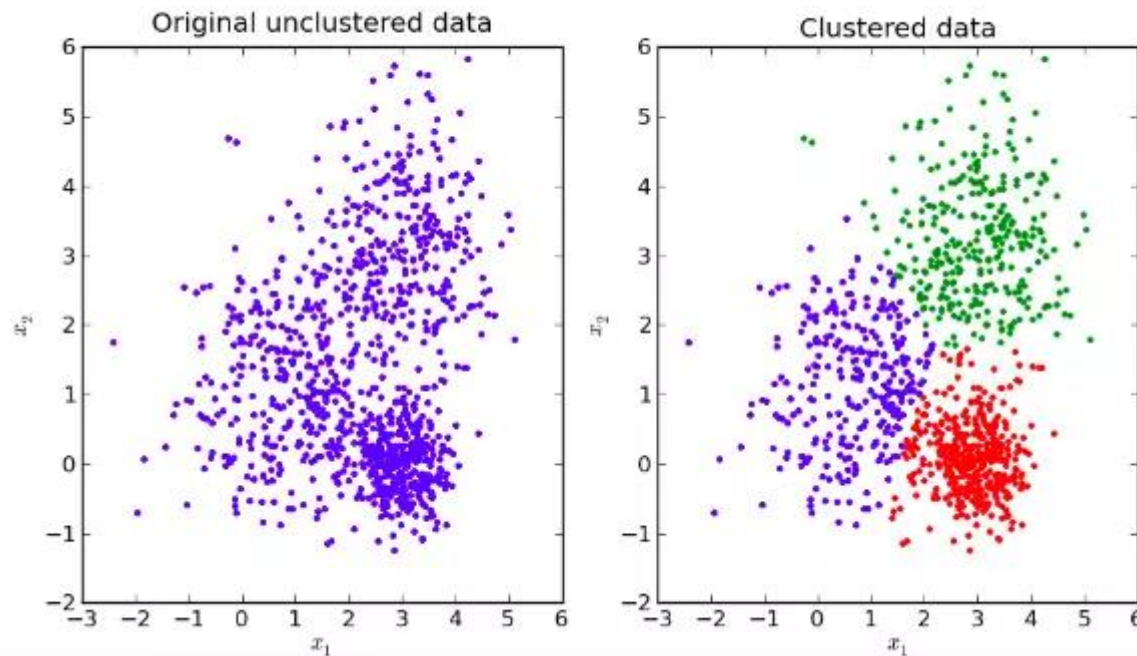
- K-means là thuật toán học không giám sát (Unsupervised Learning).
- Mục tiêu: Phân chia dữ liệu (Không được gán nhãn) thành các cụm sao cho dữ liệu trong cùng 1 cụm có tính chất giống nhau.
- Mô hình hóa toán học : Dữ liệu đầu vào là X (Không có gán nhãn) . Đầu ra là K cụm và nhãn cho từng điểm dữ liệu.



II. K MEANS CLUSTERING:

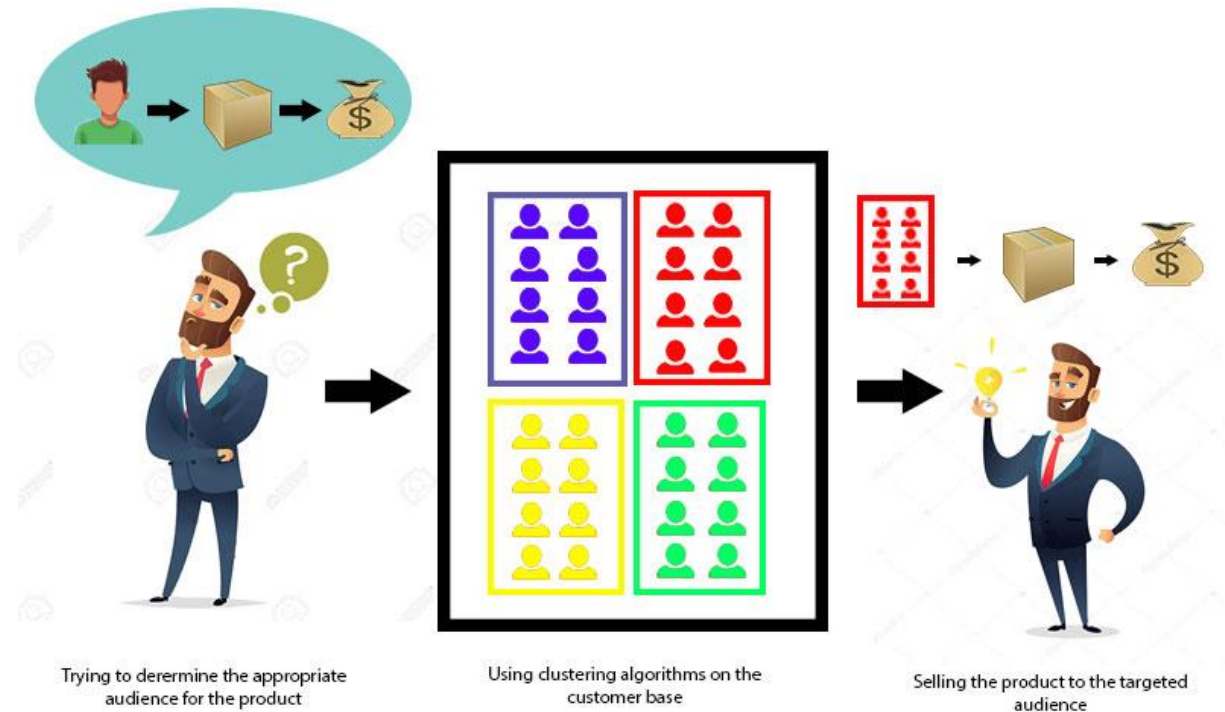
1. Giới thiệu về thuật toán:

- Ý tưởng về cụm : Tập hợp các điểm ở gần nhau trong cùng một không gian.
- Ý tưởng về K-means: tìm các điểm centers(điểm đại diện) -> tính khoảng cách của các điểm lân cận tới điểm centers -> so sánh khoảng cách -> cập nhật cụm cho các điểm lân cận.



II. K MEANS CLUSTERING:

- **Ứng dụng thực tế :**
 - Phân nhóm khách hàng trong kinh doanh.
 - Gán nhãn các thuật toán học có giám sát.



II. K MEANS CLUSTERING:

2. Tóm tắt cơ sở toán học:

- Input: N điểm dữ liệu $X = [x_1, x_2, \dots, x_N]$, k là số cluster mong muốn.
- Output: center của mỗi cluster (m_1, m_2, \dots, m_k) và label vector của mỗi điểm dữ liệu.
- Phân tích thuật toán:
 - + Độ phức tạp: Thấp \rightarrow Tương đối
 - + Ưu điểm : Độ phức tạp vừa phải, độ chính xác tương đối cao.
 - + Nhược điểm : Không phù hợp với non-convex data, dễ bị Outliers gây nhiễu, phụ thuộc vào số lượng cluster cần chia.

II. K MEANS CLUSTERING:

2. Tóm tắt cơ sở toán học

- Với mỗi điểm dữ liệu \mathbf{x}_i , đặt y_{ij} là vector one-hot của nó: $y_{ij} \in \{0, 1\} \quad \forall j; \quad \sum_{j=1}^K y_{ij} = 1$

- Hàm sai số giữa điểm dữ liệu và center của cluster: $\|\mathbf{x}_i - \mathbf{m}_k\|_2^2$

- Khi \mathbf{x}_i được phân cụm vào k thì hàm sai số giữa điểm và center được viết lại như sau:

$$y_{ik} \|\mathbf{x}_i - \mathbf{m}_k\|_2^2 = \sum_{j=1}^K y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2$$

II. K MEANS CLUSTERING:

2. Tóm tắt cơ sở toán học:

- Hàm mất mát:

$$\mathcal{L}(\mathbf{Y}, \mathbf{M}) = \sum_{i=1}^N \sum_{j=1}^K y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2$$

- Tối ưu hàm mất : Cố định M tìm Y
- Tìm label cho từng điểm dữ liệu : $\mathbf{y}_i = \arg \min_{\mathbf{y}_i} \sum_{j=1}^K y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2 \quad (3)$
- Có thể thu gọn lại tìm label cho từng điểm dữ liệu như sau : $j = \arg \min_j \|\mathbf{x}_i - \mathbf{m}_j\|_2^2$

II. K MEANS CLUSTERING:

2. Tóm tắt cơ sở toán học:

- Tối ưu hàm mất : Cố định Y tìm M

- Cập nhật center cho từng cụm K : $\mathbf{m}_j = \arg \min_{\mathbf{m}_j} \sum_{i=1}^N y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2.$

- Đạo hàm để tìm min thì ta sẽ công thức

cập nhật center như sau :

$$\mathbf{m}_j = \frac{\sum_{i=1}^N y_{ij} \mathbf{x}_i}{\sum_{i=1}^N y_{ij}}$$

II. K MEANS CLUSTERING:

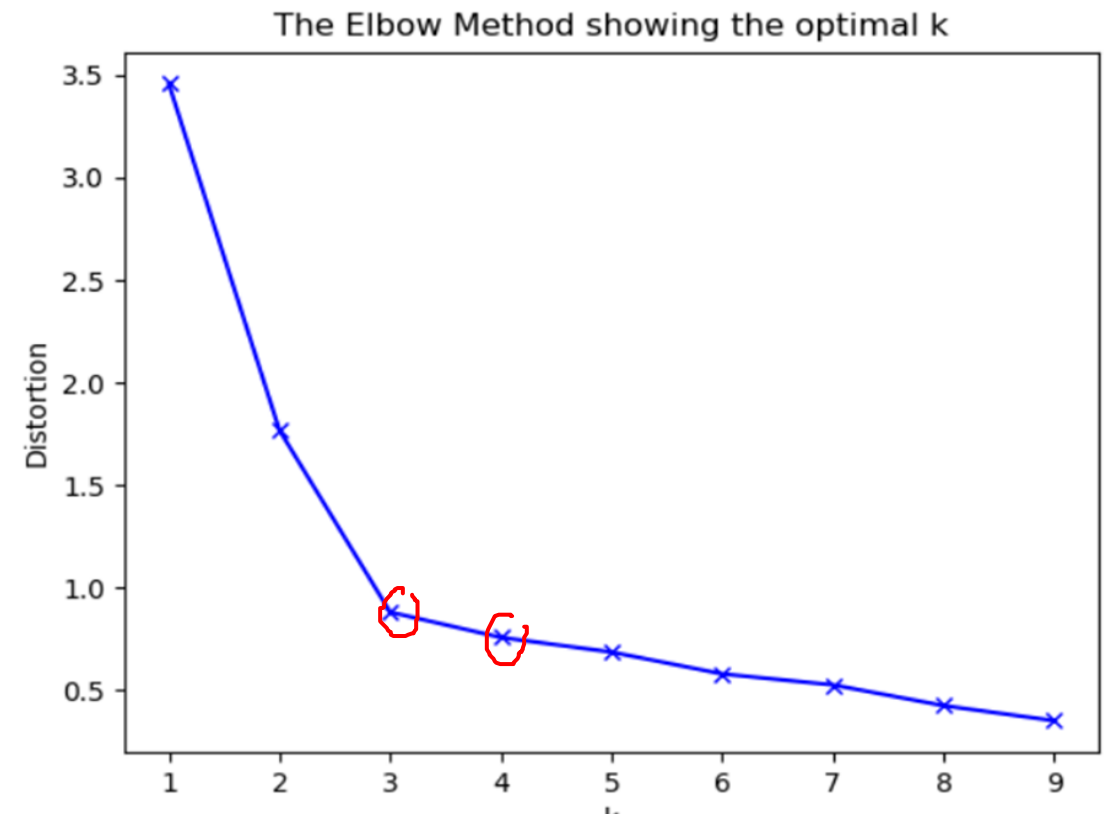
3. Tóm tắt thuật toán:

- **INPUT:** Dữ liệu X và số lượng cluster cần tìm K .
 - **OUTPUT:** Các center M và label vector cho từng điểm dữ liệu Y .
1. Chọn K điểm bất kỳ làm các center ban đầu.
 2. Phân mỗi điểm dữ liệu vào cluster có center gần nó nhất.
 3. Nếu việc gán dữ liệu vào từng cluster ở bước 2 không thay đổi so với vòng lặp trước nó thì ta dừng thuật toán.
 4. Cập nhật center cho từng cluster bằng cách lấy trung bình cộng của tất cả các điểm dữ liệu đã được gán vào cluster đó sau bước 2.
 5. Quay lại bước 2.

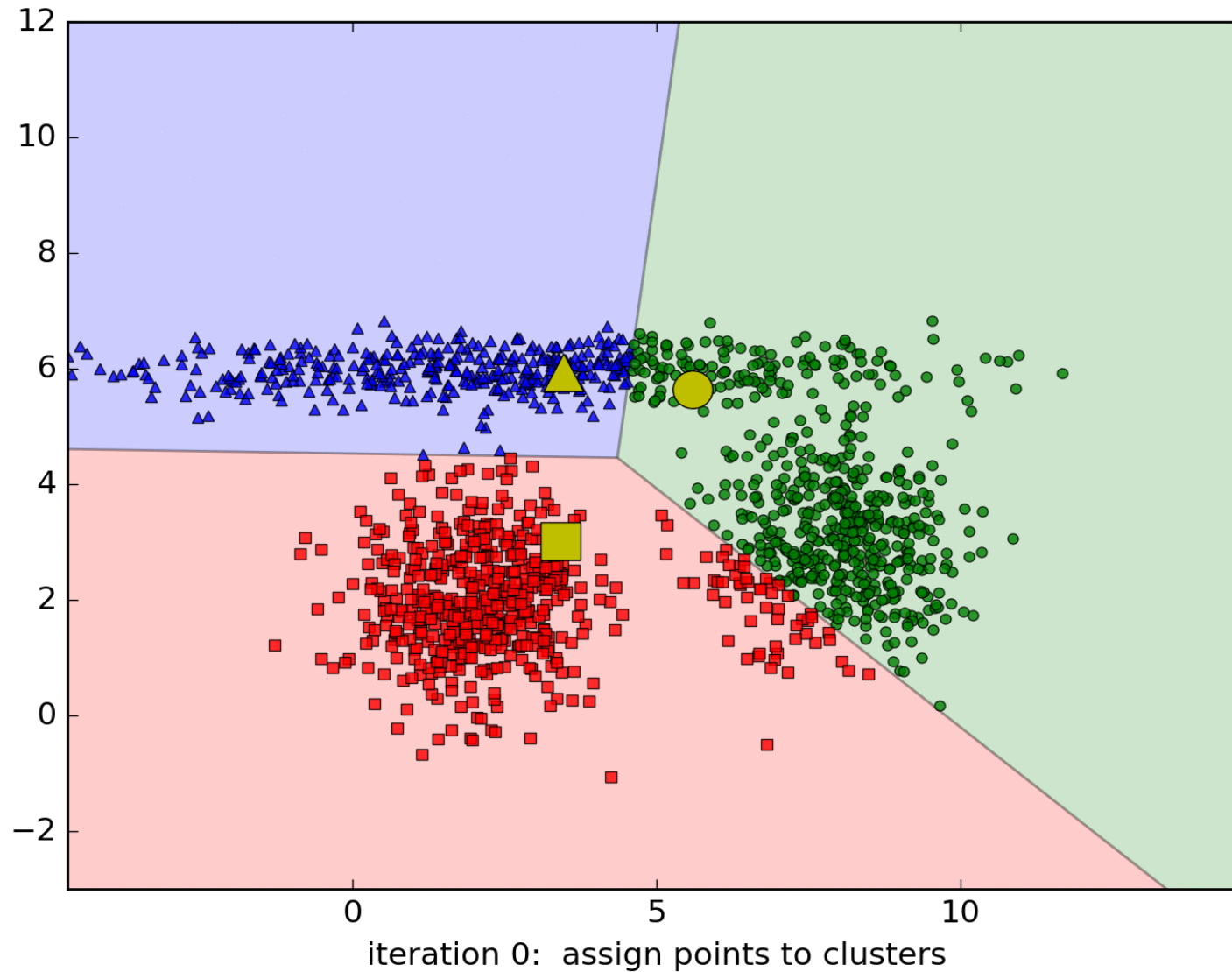
II. K MEANS CLUSTERING:

4. Đánh giá bài toán K-means bằng Elbow graph:

- Mục đích: Chọn số cluster K phù hợp.
- Cấu trúc Graph:
 - Trục hoành : Số lượng cluster.
 - Trục tung : Trung bình cộng bình phương khoảng cách giữa tâm cluster đến các điểm còn lại.



II. K MEANS CLUSTERING:



III. SINGULAR VALUE DECOMPOSITION:

1. Giới thiệu thuật toán:

- Phương pháp SVD đã được phát triển dựa trên những tính chất của ma trận trực giao và ma trận đường chéo để tìm ra một ma trận xấp xỉ với ma trận gốc.

III. SINGULAR VALUE DECOMPOSITION:

2. Một số kiến thức về đại số tuyến tính:

a) Trị riêng và vector riêng

- Cho một ma trận vuông A cấp n .
- Số λ được gọi là *trị riêng* của A nếu tồn tại vectơ $x \in \mathbb{R}^n$, $x \neq \theta$ thoả mãn:

$$Ax = \lambda x$$

Khi đó vector $x \neq \theta$ được gọi là *véctơ riêng* của A ứng với trị riêng λ .

III. SINGULAR VALUE DECOMPOSITION:

2. Một số kiến thức về đại số tuyến tính:

b) Hệ vector trực giao, trực chuẩn

- Xét hệ cơ sở $u = \{u_1, u_2, u_3, \dots, u_m\}$ trong \mathbb{R}^m .

• Nếu u thỏa:
$$\begin{cases} u_i \neq \theta \\ \langle u_i, u_j \rangle = 0 \quad \forall 1 \leq i, j \leq m; i \neq j \end{cases} \quad (*)$$

Thì khi đó u là một hệ trực giao.

• Nếu u thỏa (*) và chuẩn (norm) $\|u_i\|_2 = 1$. Khi đó u là một hệ trực chuẩn.

III. SINGULAR VALUE DECOMPOSITION:

3. Thuật toán SVD:

1	2	3	4
2	4	6	8
10	20	30	40
20	40	60	80

=

1
2
10
20

\otimes

1	2	3	4
---	---	---	---

1	2	3	4
5	6	99	55
23	255	120	34
75	42	60	0

=

?

III. SINGULAR VALUE DECOMPOSITION:

3. Thuật toán SVD:

$$A_{m \times n} = U_{m \times m} \Sigma_{m \times n} (V_{n \times n})^T$$

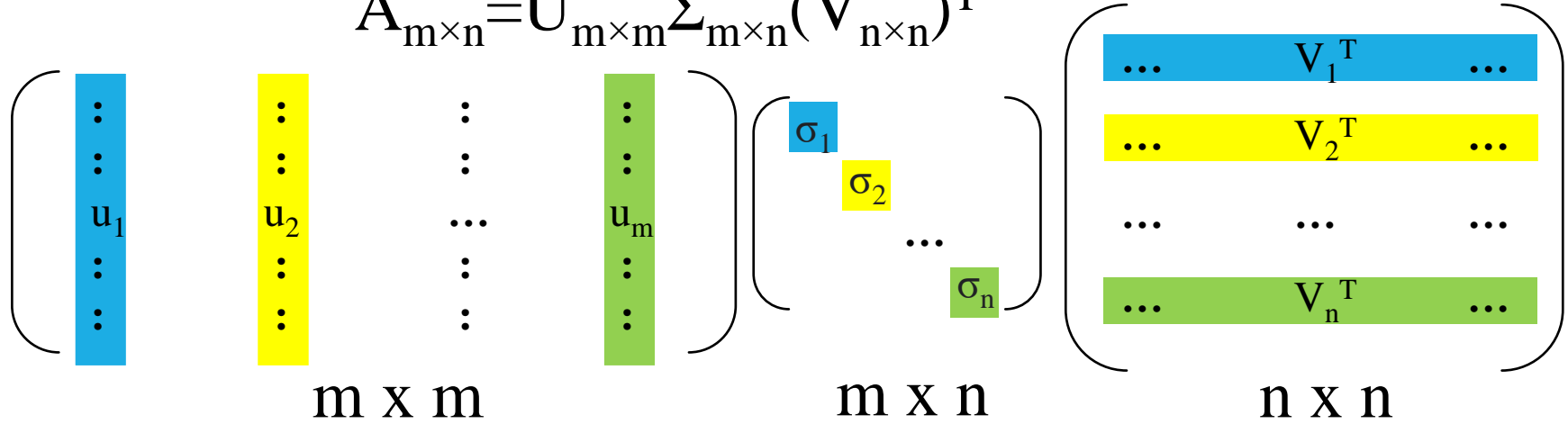
The diagram illustrates the SVD decomposition $A_{m \times n} = U_{m \times m} \Sigma_{m \times n} (V_{n \times n})^T$. The matrices are represented as follows:

- $U_{m \times m}$ is a matrix with columns u_1, u_2, \dots, u_m , shown as a blue vertical bar.
- $\Sigma_{m \times n}$ is a diagonal matrix with singular values $\sigma_1, \sigma_2, \dots, \sigma_n$, shown as a yellow vertical bar.
- $(V_{n \times n})^T$ is a matrix with rows $V_1^T, V_2^T, \dots, V_n^T$, shown as a green vertical bar.

The dimensions of the matrices are indicated below them: $m \times m$ for U , $m \times n$ for Σ , and $n \times n$ for V^T .

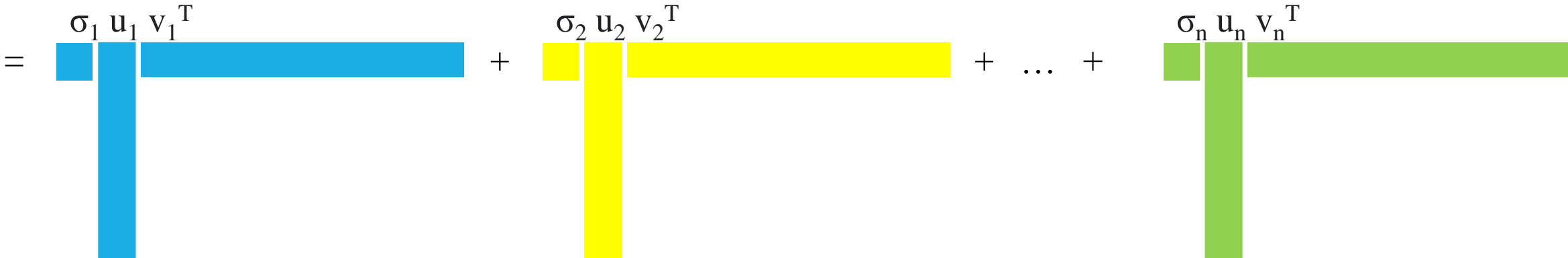
III. SINGULAR VALUE DECOMPOSITION:

3. Thuật toán SVD:

$$A_{m \times n} = U_{m \times m} \Sigma_{m \times n} (V_{n \times n})^T$$


The diagram illustrates the SVD decomposition of matrix A into three components: U , Σ , and V^T .

- U is an $m \times m$ matrix with columns u_1, u_2, \dots, u_m .
- Σ is an $m \times n$ matrix with singular values $\sigma_1, \sigma_2, \dots, \sigma_n$.
- V^T is an $n \times n$ matrix with rows $V_1^T, V_2^T, \dots, V_n^T$.

$$= \sigma_1 \cdot u_1 \cdot v_1^T + \sigma_2 \cdot u_2 \cdot v_2^T + \dots + \sigma_n \cdot u_n \cdot v_n^T$$


The diagram shows the decomposition of A as a sum of rank-1 matrices:

$$= \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \dots + \sigma_n u_n v_n^T$$

III. SINGULAR VALUE DECOMPOSITION:

3. Thuật toán SVD:

$$A_{m \times n} = U_{m \times m} \Sigma_{m \times n} (V_{n \times n})^T$$

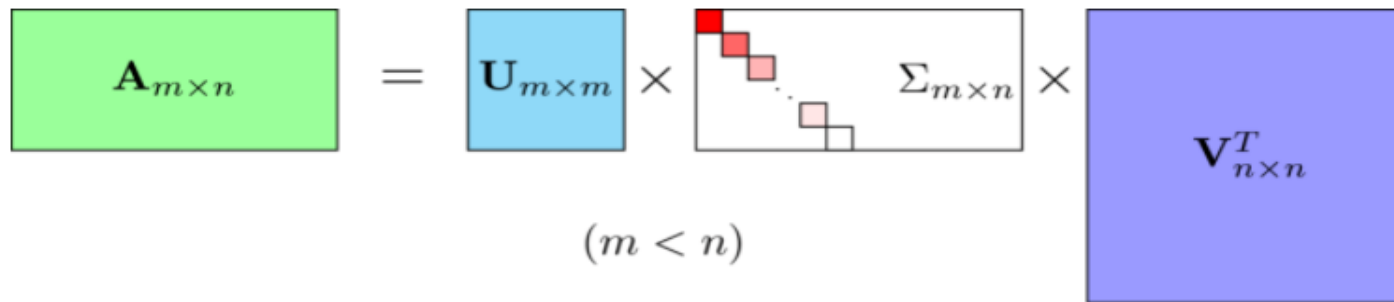


Diagram illustrating the SVD decomposition for the case $m < n$. A green rectangle labeled $A_{m \times n}$ is equal to a blue rectangle labeled $U_{m \times m}$ multiplied by a white rectangle labeled $\Sigma_{m \times n}$ (containing a diagonal of red squares) multiplied by a blue rectangle labeled $V_{n \times n}^T$. The condition $(m < n)$ is noted below the Σ matrix.



Diagram illustrating the SVD decomposition for the case $m > n$. A green rectangle labeled $A_{m \times n}$ is equal to a blue rectangle labeled $U_{m \times m}$ multiplied by a white rectangle labeled $\Sigma_{m \times n}$ (containing a diagonal of red squares) multiplied by a blue rectangle labeled $V_{n \times n}^T$. The condition $(m > n)$ is noted below the Σ matrix.

III. SINGULAR VALUE DECOMPOSITION:

3. Thuật toán SVD:

Tính U, Σ, V

- Bước 1: Tính $A^T A$
- Bước 2: Tìm trị riêng và vectơ riêng của $A^T A$
 - + $(A^T A)x = \lambda x \Leftrightarrow (A^T A - \lambda I)x = 0$
 - + $T = (A^T A - \lambda I)$. Giải $\det(T) = 0 \rightarrow \lambda = \text{????}$
- Bước 3: Tìm hệ vectơ trực giao từ λ thay vào $T \rightarrow$ Tìm được hệ trực chuẩn $\rightarrow V$

III. SINGULAR VALUE DECOMPOSITION:

3. Thuật toán SVD:

Tính U , Σ , V

- Bước 4: Tìm Σ

- + Điền các giá trị căn bậc hai các λ tìm được vào ma trận theo đường chéo, giảm dần trên ma trận.

- Bước 5: Tìm U

- + $A = U\Sigma V^T \Leftrightarrow AV = US \Leftrightarrow AVS^{-1} = U \rightarrow U$

$$U, \Sigma, V \Rightarrow A' = U'_{m \times k} \Sigma'_{k \times k} (V'_{k \times n})^T (\simeq A)$$

III. SINGULAR VALUE DECOMPOSITION:

$k = 5$: error = 0.1492



TỔNG QUAN

- **Lịch Sử**
- **Tổng quan về cấu trúc trong CNN**
- **Ví dụ về cách thức hoạt động của CNN và giới thiệu về các thuật ngữ quan trọng**
- **CNNs for NLP**

LỊCH SỬ

- Yann LeCun

New York University

Facebook Artificial Intelligence Research



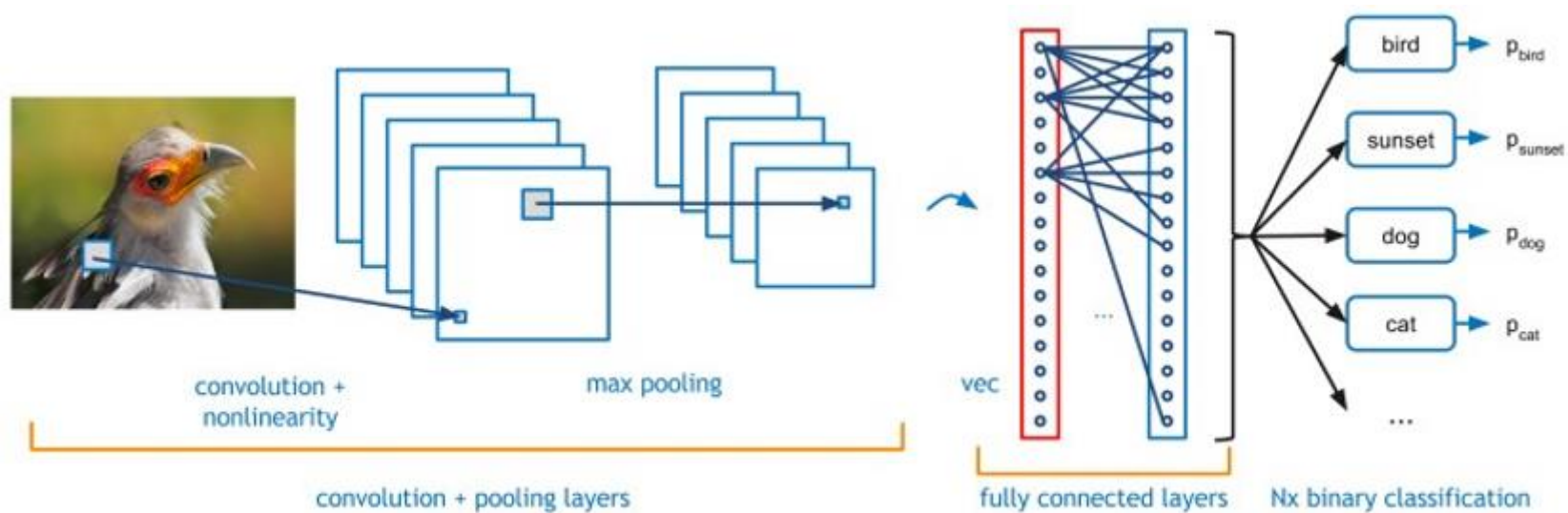
- Yoshua Bengio

Université de Montréal



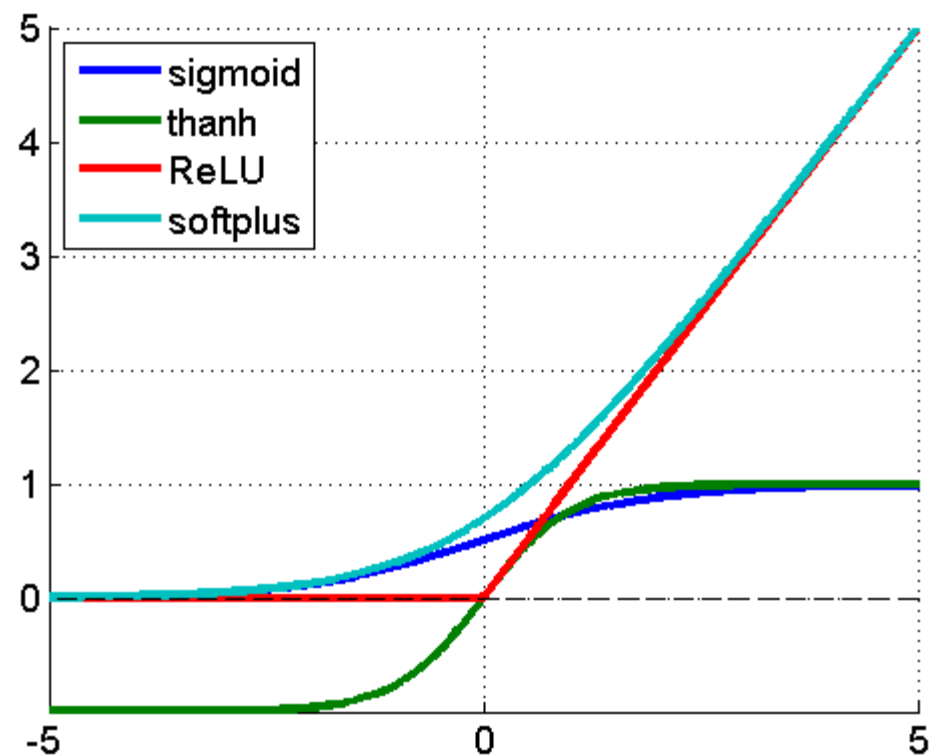
- Vào năm 1995, **Yann LeCun** và **Yoshua Bengio** đã giới thiệu khái niệm về convolutional neural networks.

CẤU TRÚC TRONG CNN



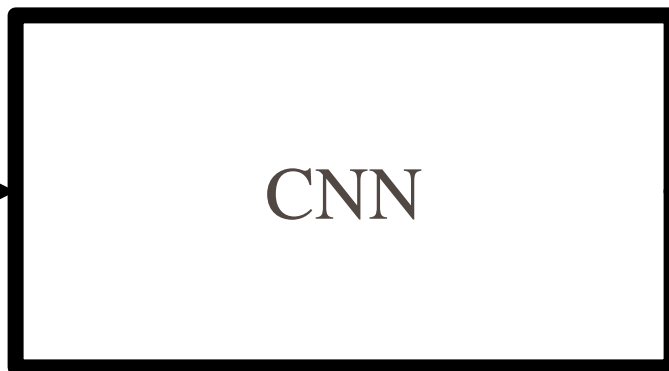
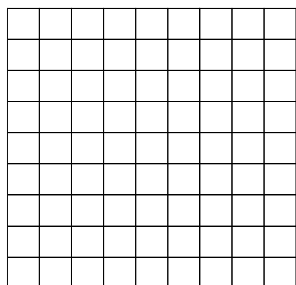
NON-LINEARITY?

- Một mạng neural chỉ là phi tuyến tính nếu ‘squash’ tín hiệu đầu ra từ các node có chức năng kích hoạt phi tuyến tính
 - => Xử dụng hàm tích lũy
- Việc diễn giải tín hiệu đầu ra ‘squashed’ có thể được hiểu là cường độ của tín hiệu này



VÍ DỤ

Một mảng 2 chiều
của các điểm ảnh

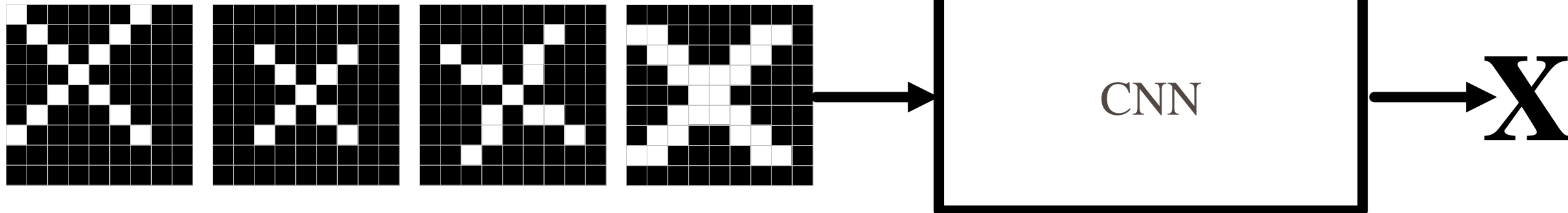


X or **0**

VÍ DỤ



MỘT SỐ HÌNH ẢNH KHÁC

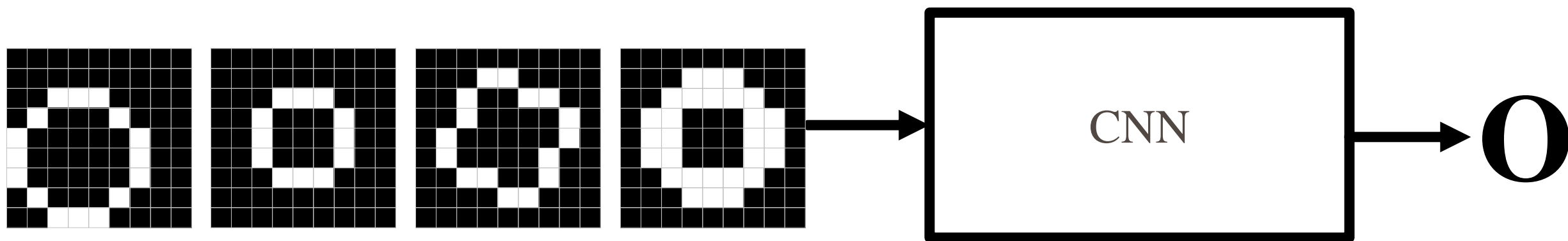


translation

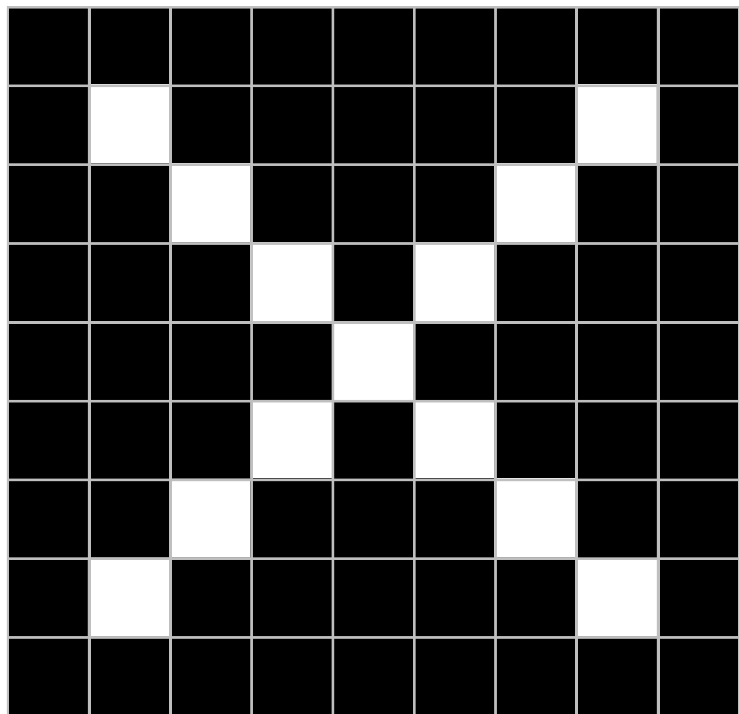
scaling

rotation

weight

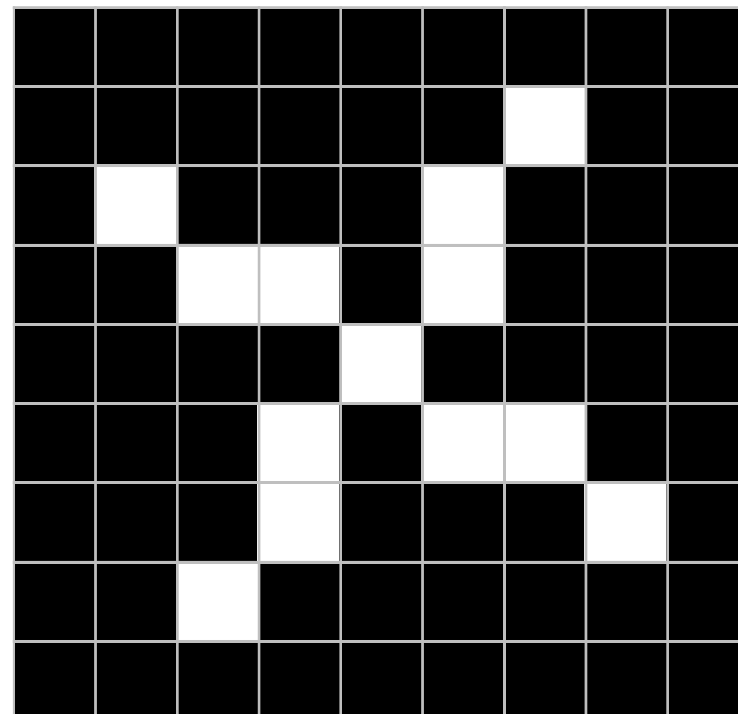


LÀM SAO ĐỂ SO SÁNH ?

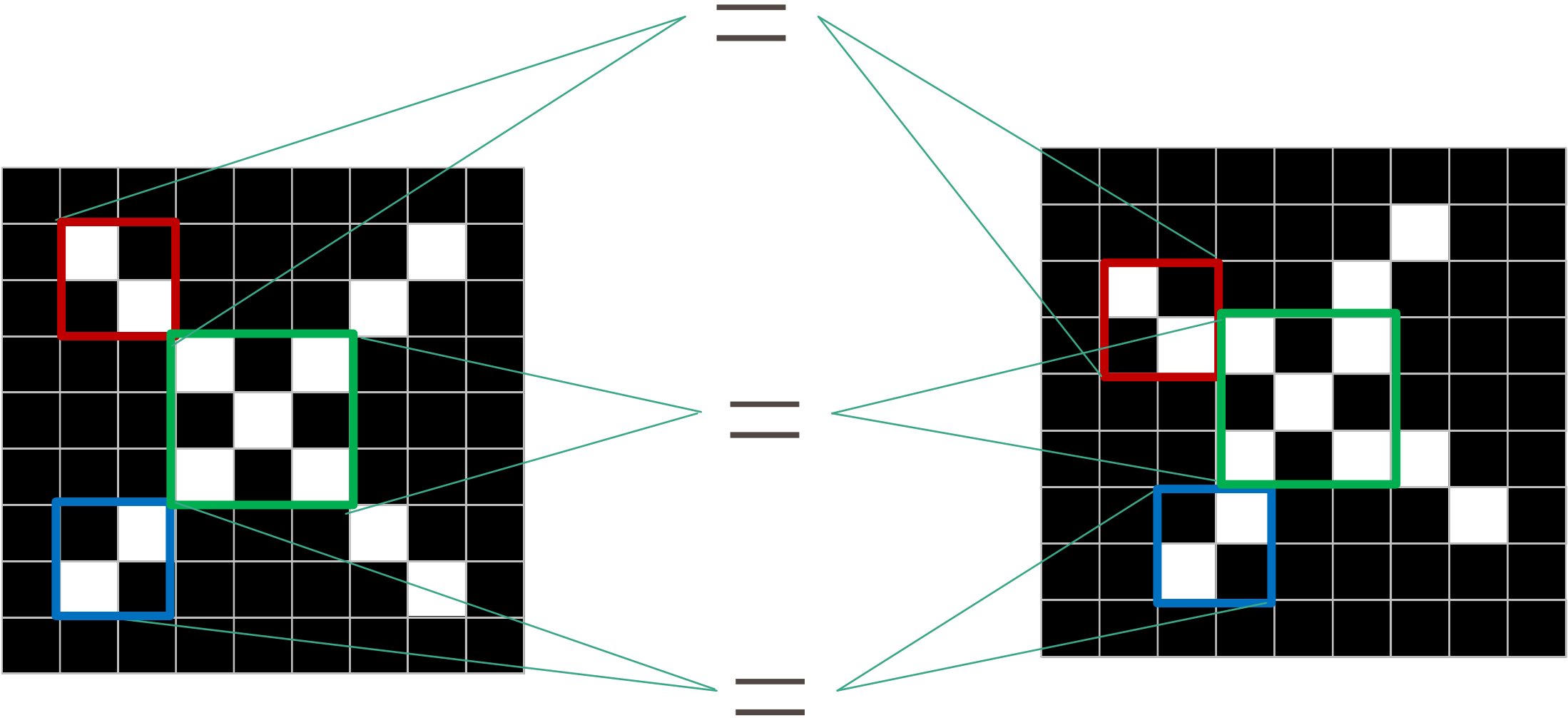


?

=



SO SÁNH VÙNG NHỎ



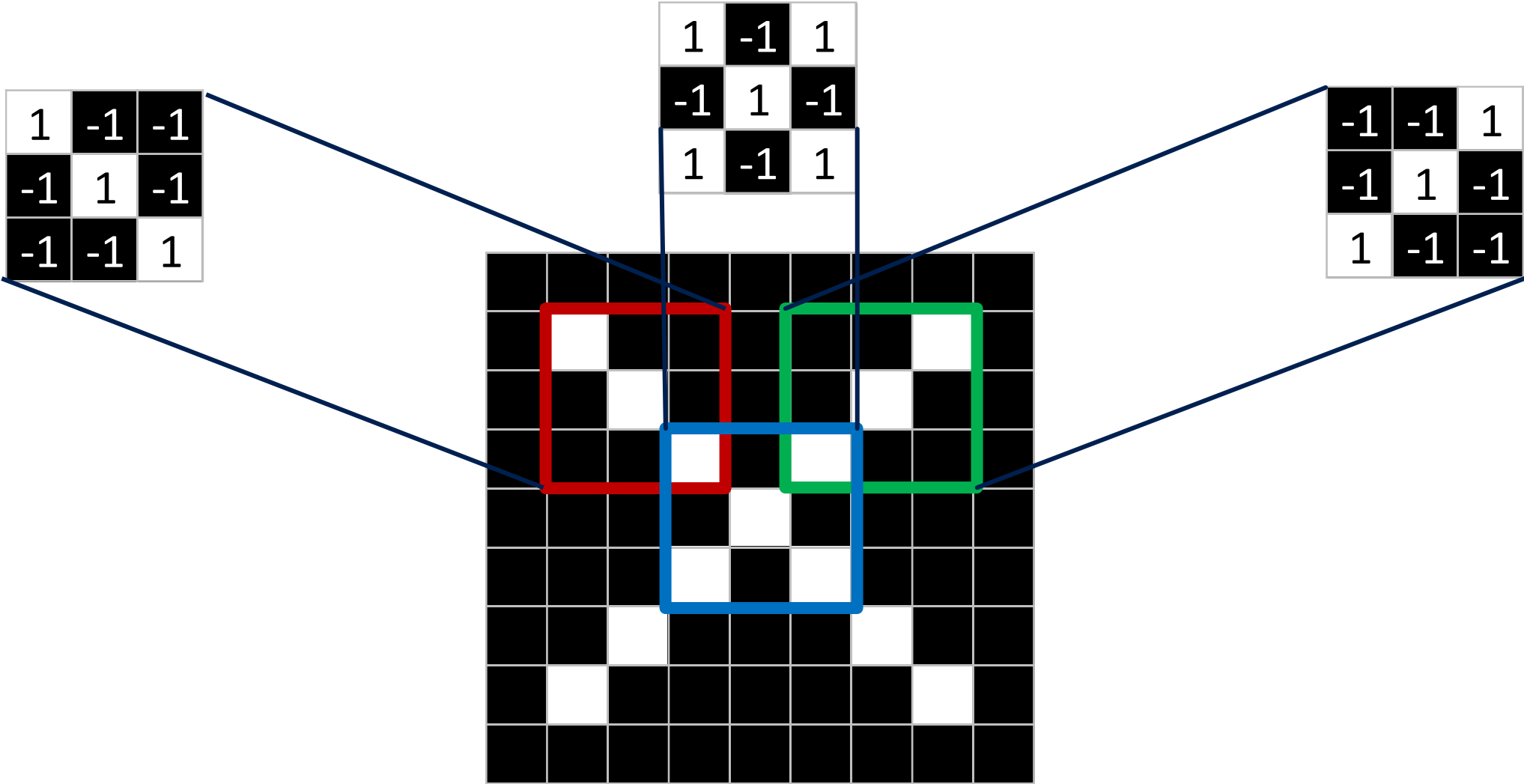
TERM: FILTER

1	-1	-1
-1	1	-1
-1	-1	1

1	-1	1
-1	1	-1
1	-1	1

-1	-1	1
-1	1	-1
1	-1	-1

TERM: FILTER



TERM: CONVOLUTION

1 _{x1}	1 _{x0}	1 _{x1}	0	0
0 _{x0}	1 _{x1}	1 _{x0}	1	0
0 _{x1}	0 _{x0}	1 _{x1}	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved
Feature

Các filters sử dụng để tạo một filter layer dựa trên dữ liệu đầu vào

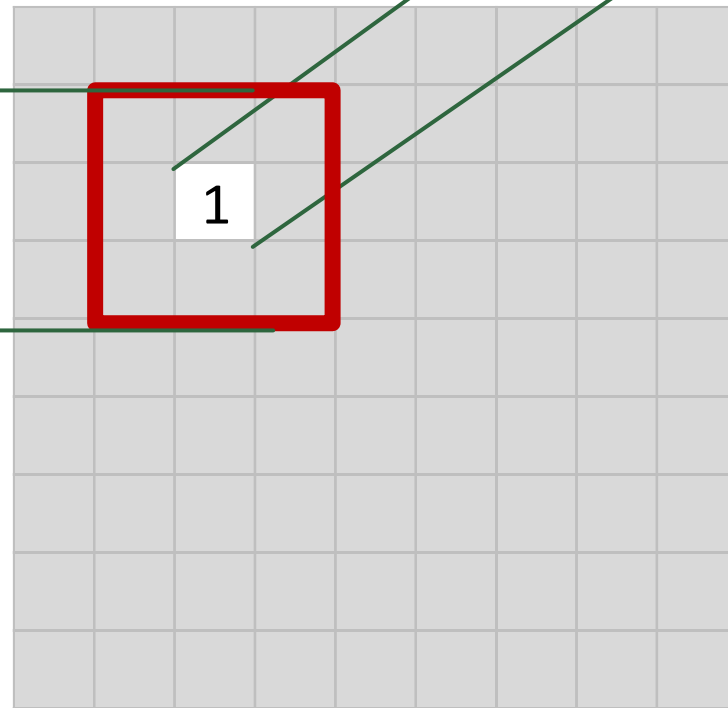
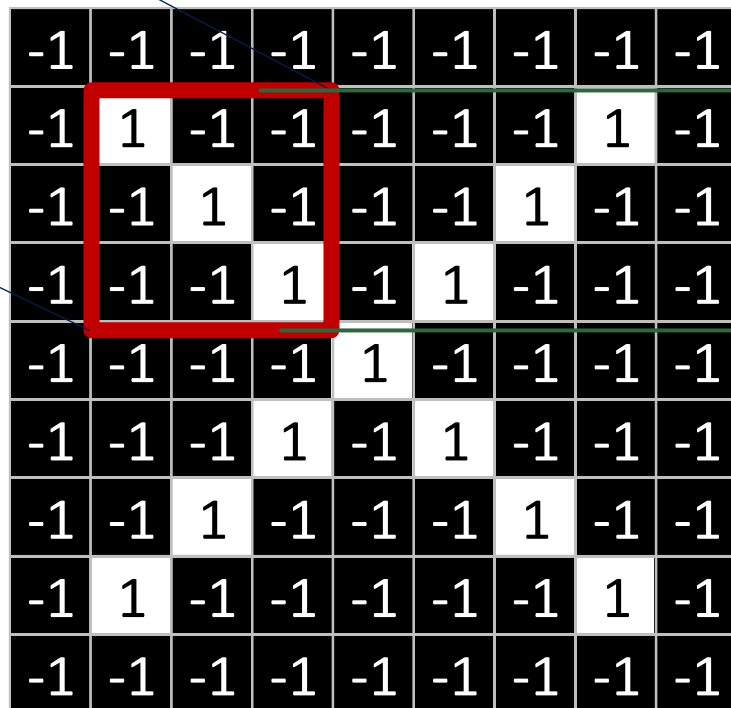
- Vùng màu vàng: đếm tất cả những cái nằm trong đường chéo của nó
- Khu vực màu xanh lá cây: các tính năng đầu vào

BUILDING THE FILTER LAYER

1	-1	-1
-1	1	-1
-1	-1	1

1	1	1
1	1	1
1	1	1

$$\frac{1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1}{9} = 1$$



-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1



1	-1	-1
-1	1	-1
-1	-1	1

=

0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1



1	-1	1
-1	1	-1
1	-1	1

=

0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.11	0.33	-0.77	1.00	-0.77	0.33	-0.11
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

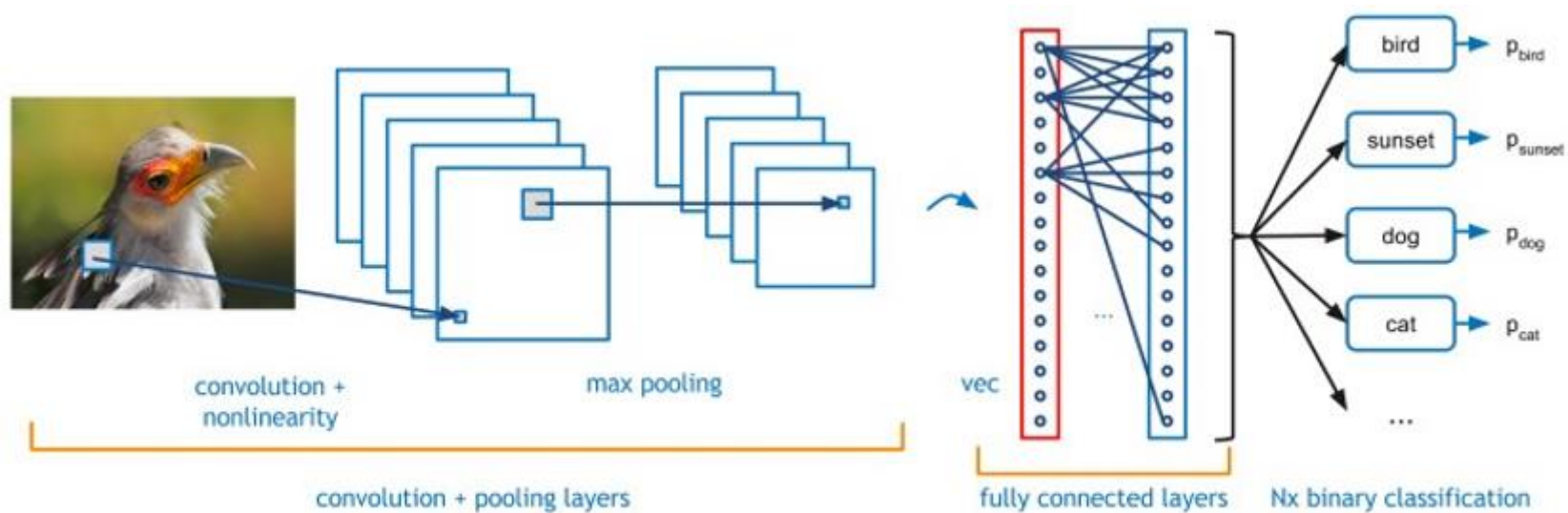


-1	-1	1
-1	1	-1
1	-1	-1

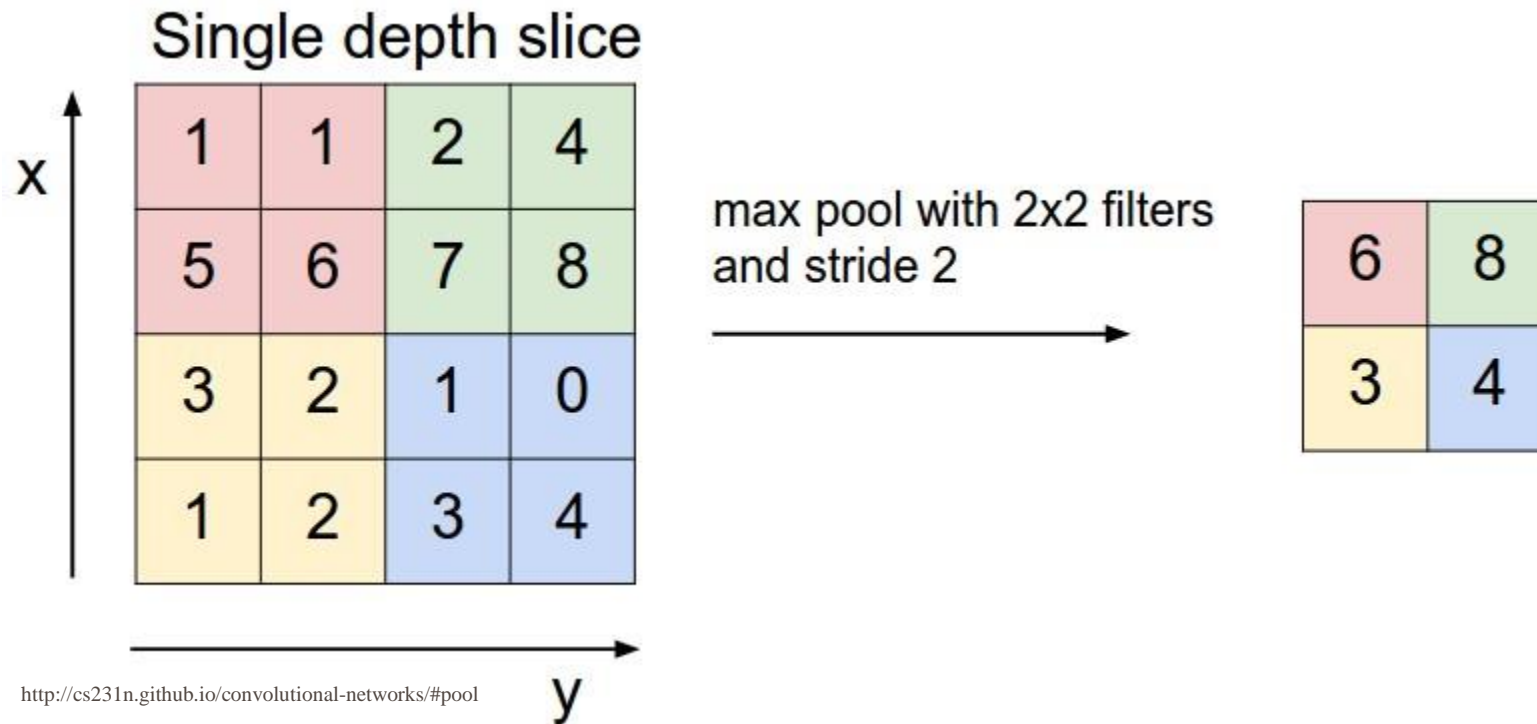
=

0.33	-0.11	0.55	0.33	0.11	-0.11	0.77
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33

CẤU TRÚC TRONG CNN



TERM: POOLING LAYERS



0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.11	0.33	-0.77	1.00	-0.77	0.33	-0.11
0.11	-0.55	0.55	-0.77	0.55	-0.55	0.11
-0.55	0.55	-0.55	0.33	-0.55	0.55	-0.55
0.33	-0.55	0.11	-0.11	0.11	-0.55	0.33

0.33	-0.11	0.55	0.33	0.11	-0.11	0.77
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.11	-0.11	1.00	-0.33	0.11	-0.11	0.55
-0.11	1.00	-0.11	0.33	-0.11	0.11	-0.11
0.77	-0.11	0.11	0.33	0.55	-0.11	0.33



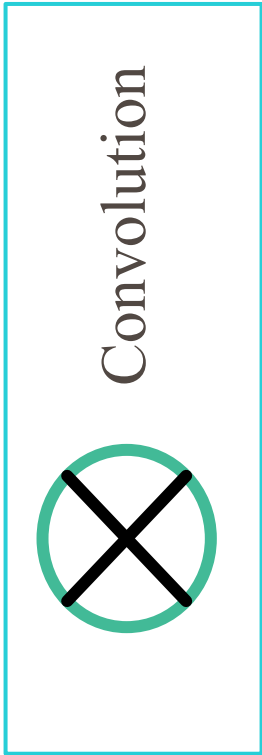
1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

0.55	0.33	0.55	0.33
0.33	1.00	0.55	0.11
0.55	0.55	0.55	0.11
0.33	0.11	0.11	0.33

0.33	0.55	1.00	0.77
0.55	0.55	1.00	0.33
1.00	1.00	0.11	0.55
0.77	0.33	0.55	0.33

STACKING

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

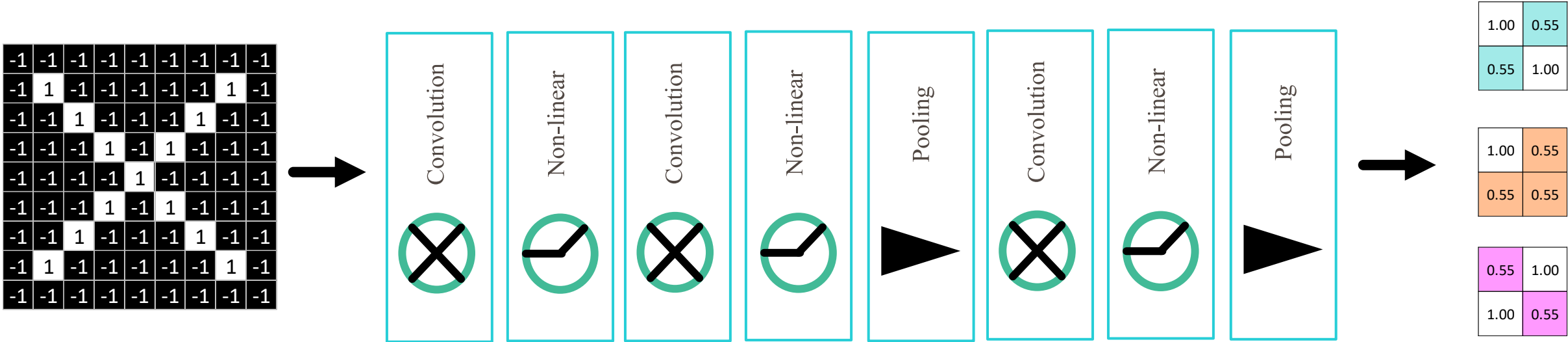


1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

0.55	0.33	0.55	0.33
0.33	1.00	0.55	0.11
0.55	0.55	0.55	0.11
0.33	0.11	0.11	0.33

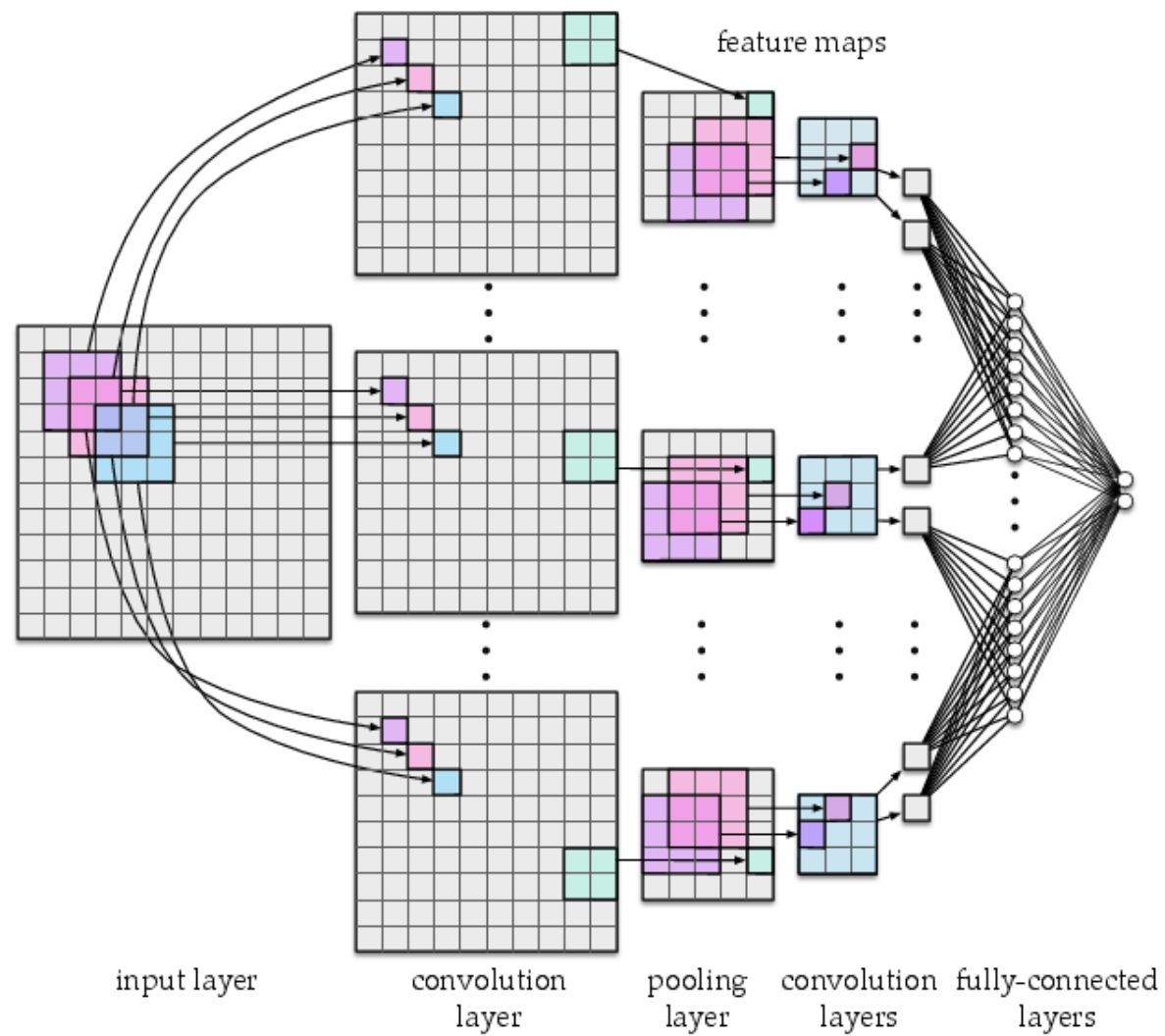
0.33	0.55	1.00	0.77
0.55	0.55	1.00	0.33
1.00	1.00	0.11	0.55
0.77	0.33	0.55	0.33

STACKING

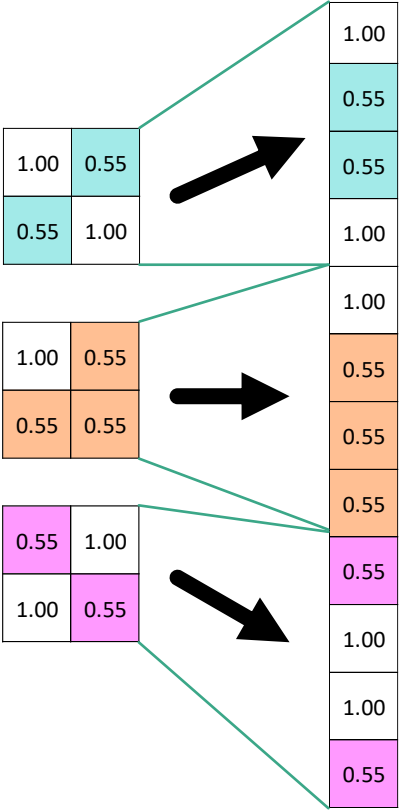


TERM: FULLY-CONNECTED

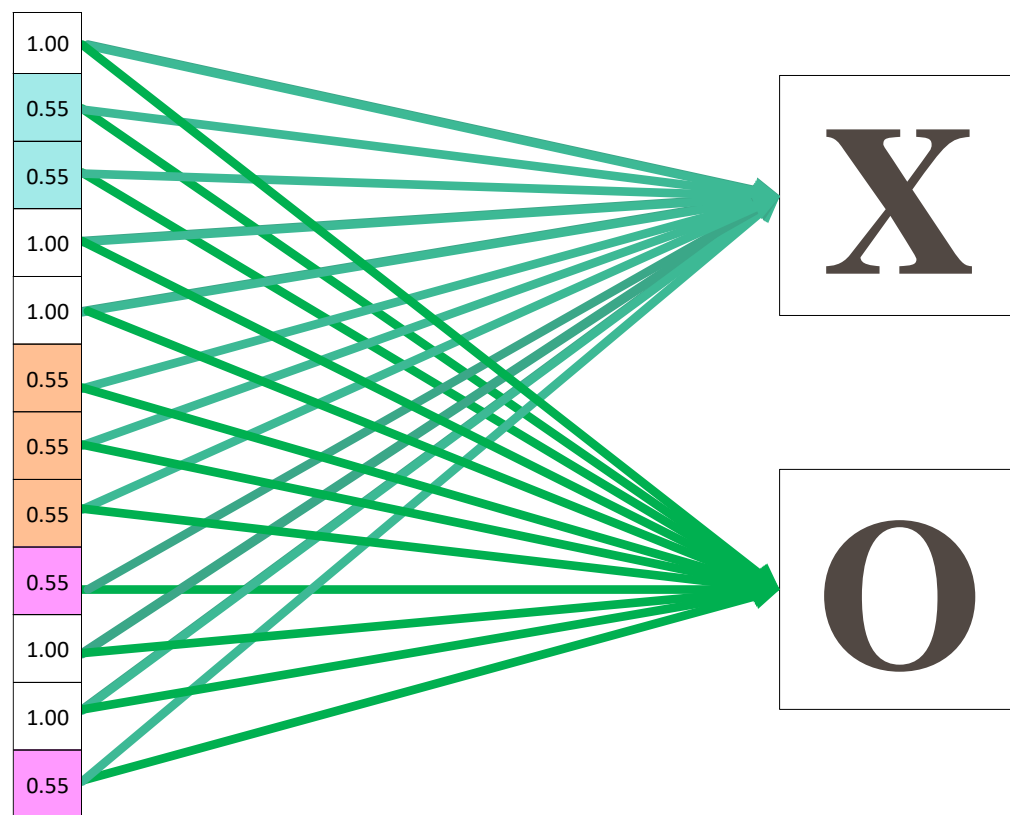
- Lấy một số lượng đầu vào và xuất ra một vectơ N chiều



VÍ DỤ: FULLY-CONNECTED LAYER

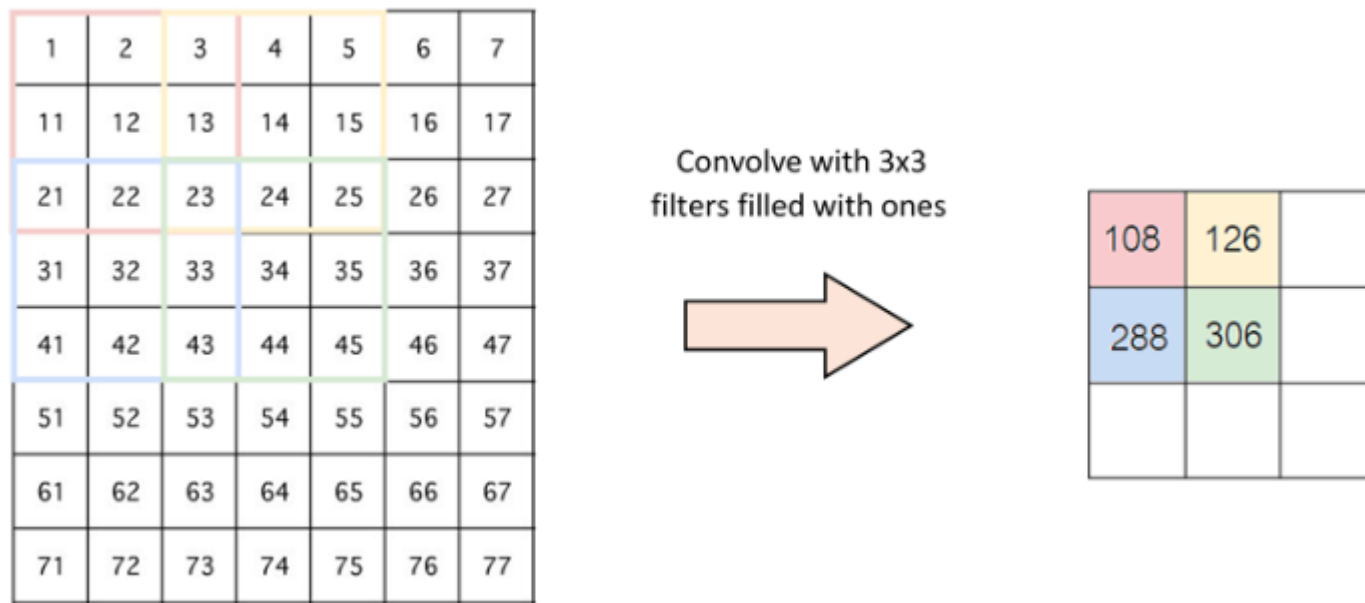


FULLY-CONNECTED LAYER => TRẢ VỀ OUTPUT

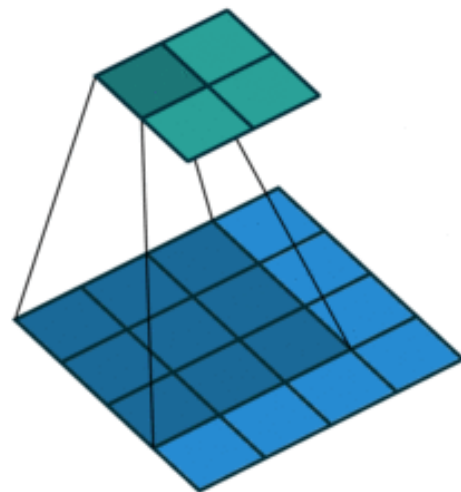


TERM: STRIDES

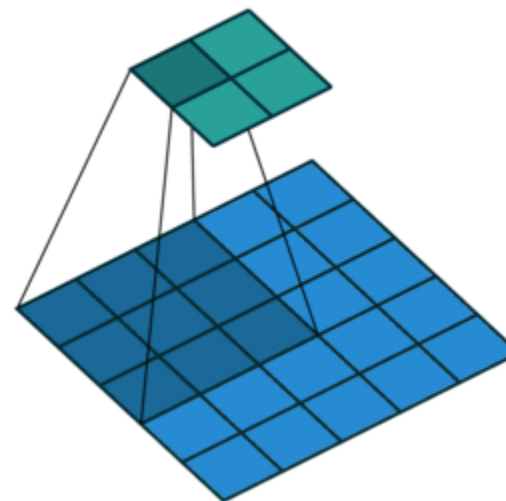
- Stride là số pixel thay đổi trên ma trận đầu vào. Khi stride là 1 thì ta di chuyển các kernel 1 pixel. Khi stride là 2 thì ta di chuyển các kernel đi 2 pixel và tiếp tục như vậy. Hình dưới là lớp tích chập hoạt động với stride là 2.



TERM: STRIDES

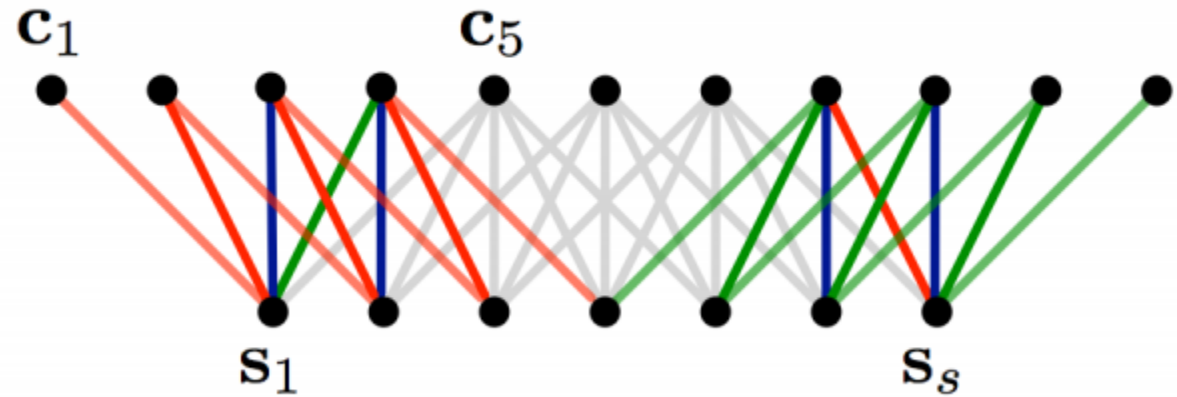
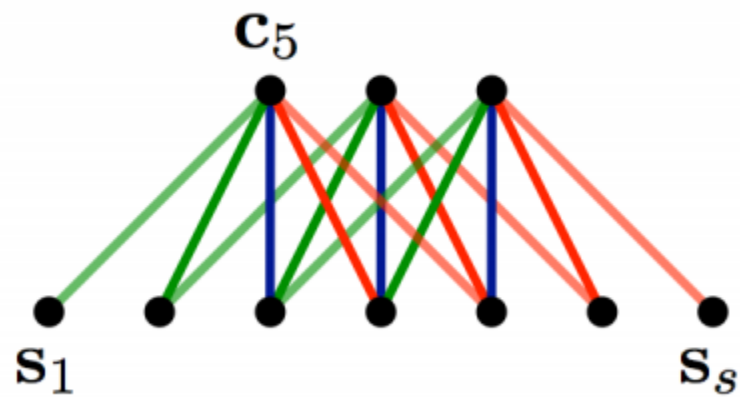


without strides



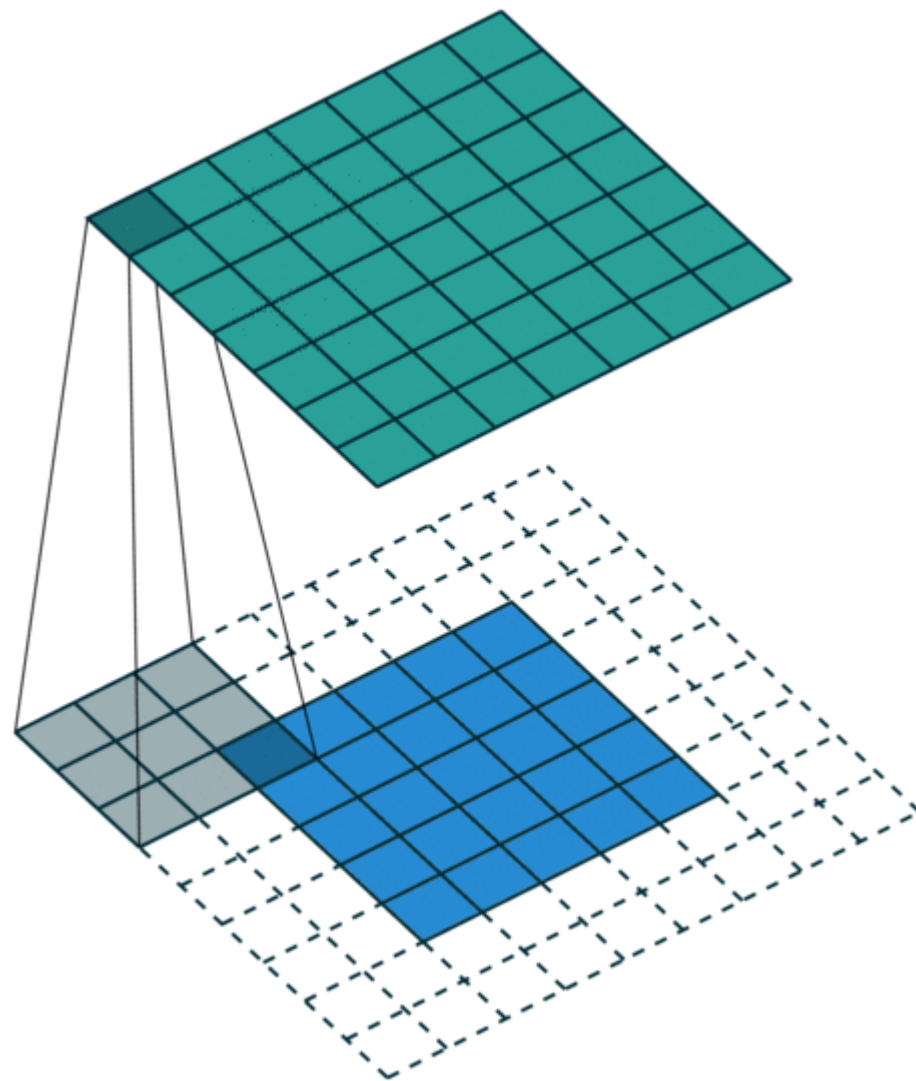
with strides

TERM: PADDING



A Convolutional Neural Network for Modelling Sentences (2014)

TERM: PADDING



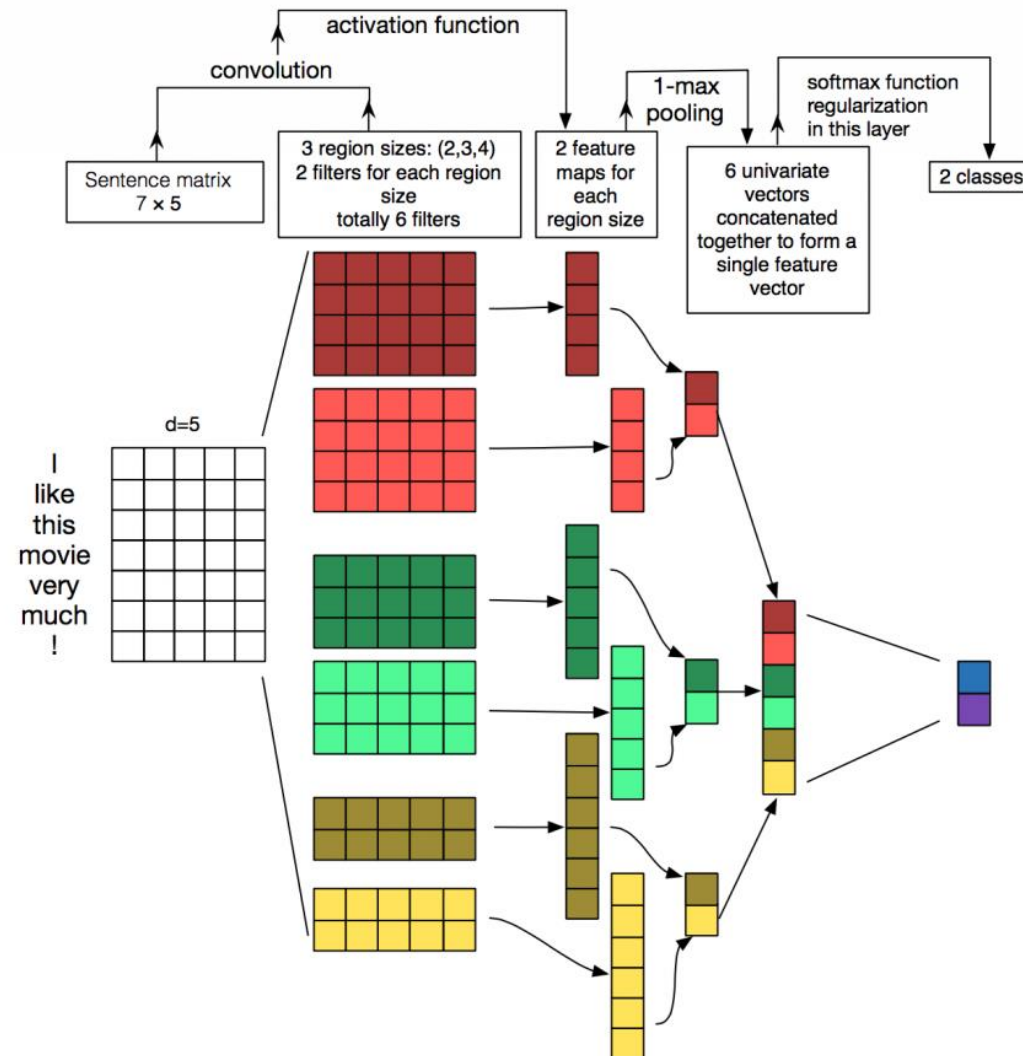


NLP (NEURO LINGUISTIC PROGRAMMING): PHÙ HỢP CHO NHỮNG LĨNH VỰC NÀO ?

Phân loại, Ví dụ:

- Phân tích cảm xúc
- Phát hiện các thư rác
- Phân loại các chủ đề

NLP



THANKS FOR LISTENING

Thành viên nhóm:

1. Nguyễn Vũ Dương - 20520465
2. Bùi Hữu Đức – 20520449
3. Phạm Phước An – 20520375
4. Hoàng Công Danh - 20520431