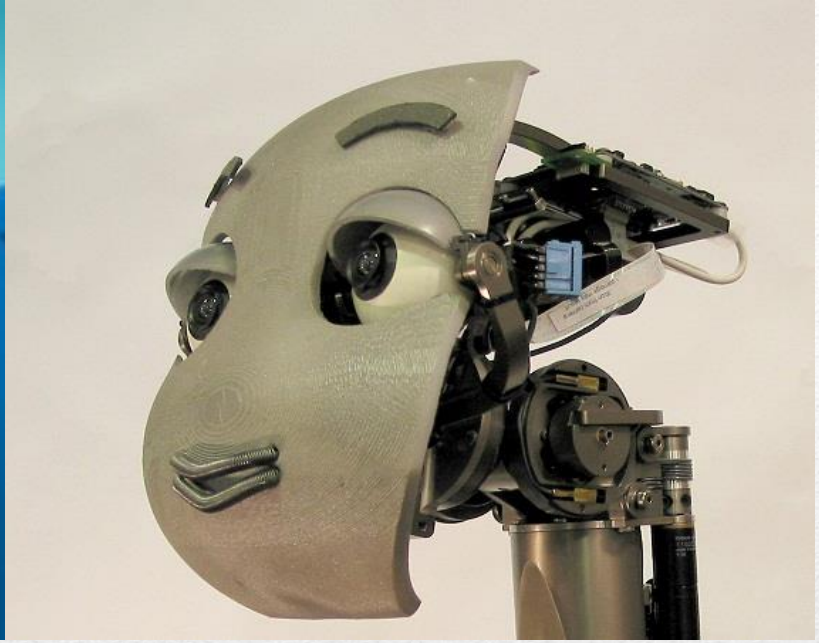


THỊ GIÁC MÁY TÍNH ...

COMPUTER VISION

Ms. HTHT

Phone Number: 0989 567 488

Chương 3. Một số phương pháp trích chọn đặc điểm ảnh và phân lớp

3.1. Phương pháp trích chọn đặc điểm SIFT (Scale-invariant feature extraction)

3.2. Phương pháp mẫu nhị phân cục bộ LBP (Local Binary Patterns)

3.3. Phương pháp biến đổi sóng nhỏ Gabor (Gabor wavelets)

3.4. Phương pháp phân lớp k láng giềng gần nhất (kNN)

3.5. Phân lớp với SVM (Support Vector Machine)

3.1. Phương pháp trích chọn đặc điểm SIFT (Scale-invariant Feature Extraction)

SIFT Background

David Lowe
Professor in UBC

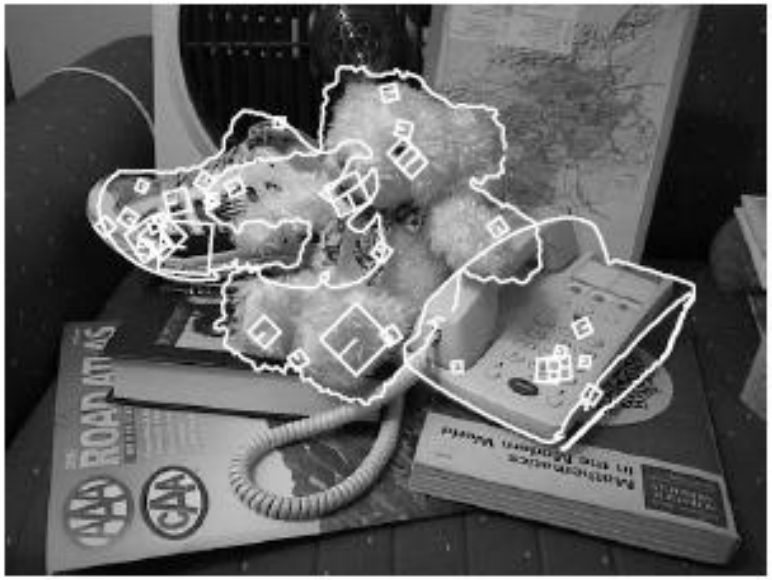
Scale-invariant feature transform

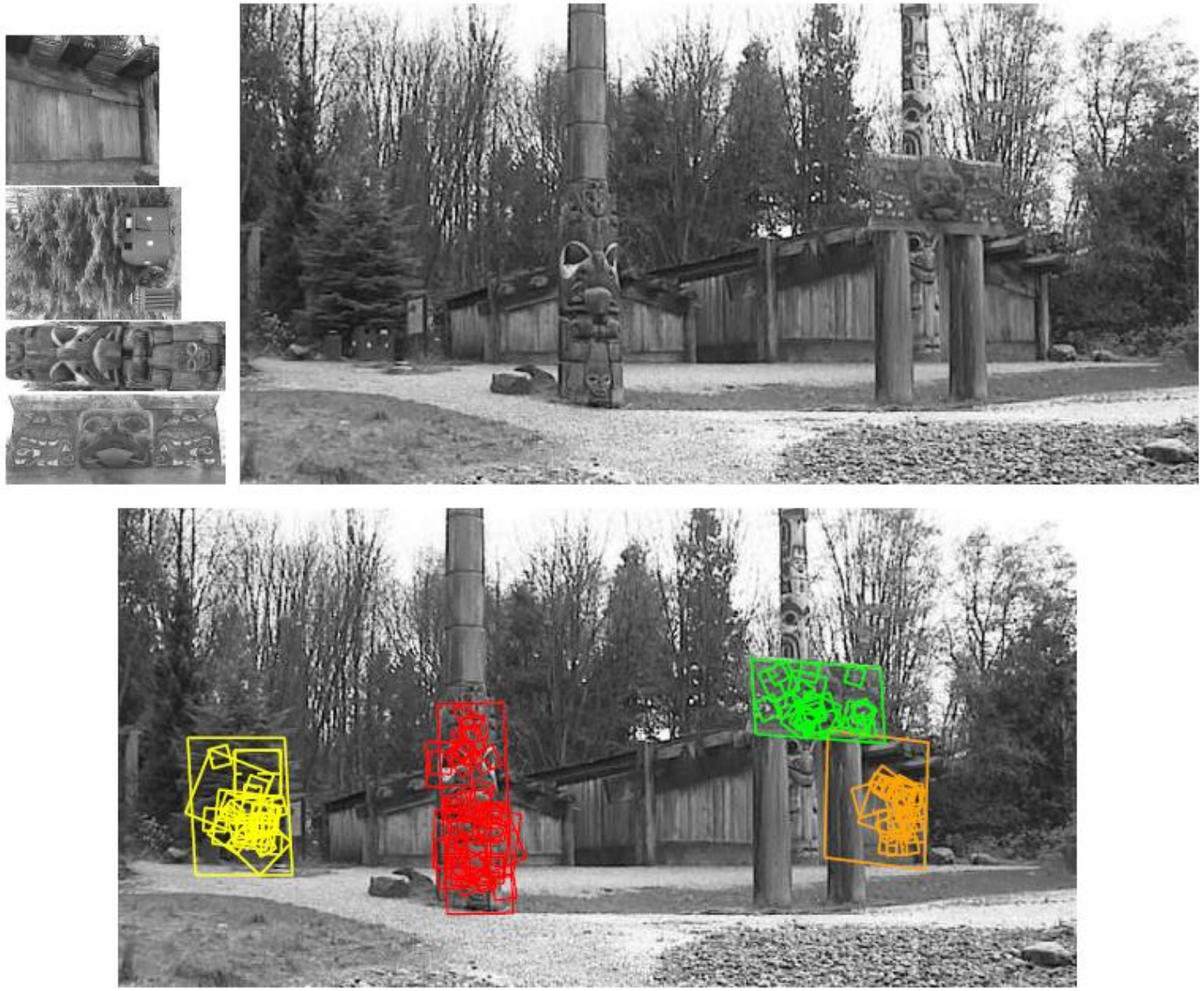
- **SIFT:** to detect and describe local features in an images.
- Proposed by *David Lowe* in ICCV1999.
- Refined in IJCV 2004.
- Cited more than 12000 times till now.
- Wildly used in image search, object recognition, video tracking, gesture recognition, *etc*.

Why SIFT is so popular?

An instance of object matching

object instance recognition (matching)


ROAD ATLAS



3.1. Đặc trưng cục bộ bất biến SIFT …

Định nghĩa – Tính chất

- Là các đặc trưng bất biến:
  - Thay đổi tỷ lệ ảnh
  - Quay ảnh
  - Thay đổi góc nhìn
  - Thêm nhiễu ảnh
  - Thay đổi cường độ chiếu sáng ảnh.
- Các đặc trưng cục bộ bất biến được trích rút từ các điểm hấp dẫn trên ảnh.

3.1. Đặc trưng cục bộ bất biến SIFT …

Phương pháp trích chọn

- Phát hiện các điểm cực trị
- Định vị các điểm hấp dẫn
- Xác định hướng cho các điểm hấp dẫn
- Mô tả các điểm hấp dẫn

Distinctive Image Features from Scale-Invariant Keypoints,
David G. Lowe, International Journal of Computer Vision, 2004

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 1: Phát hiện điểm cực trị [4],[6]

- Định vị các vùng ảnh: tìm những khu vực chứa những tính chất giống nhau khi nhìn ảnh dưới các góc nhìn khác nhau
  - Hàm không gian- tỷ lệ:
    - Tìm khu vực có khả năng bất biến với sự thay đổi tỷ lệ ảnh

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

$I(x, y)$ Ảnh truy vấn

$G(x, y, \sigma)$ Biến tỷ lệ Gaussian

$L(x, y, \sigma)$ Hàm không gian tỷ lệ của ảnh I

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 1: Phát hiện điểm cực trị (cont)

- Với $G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$

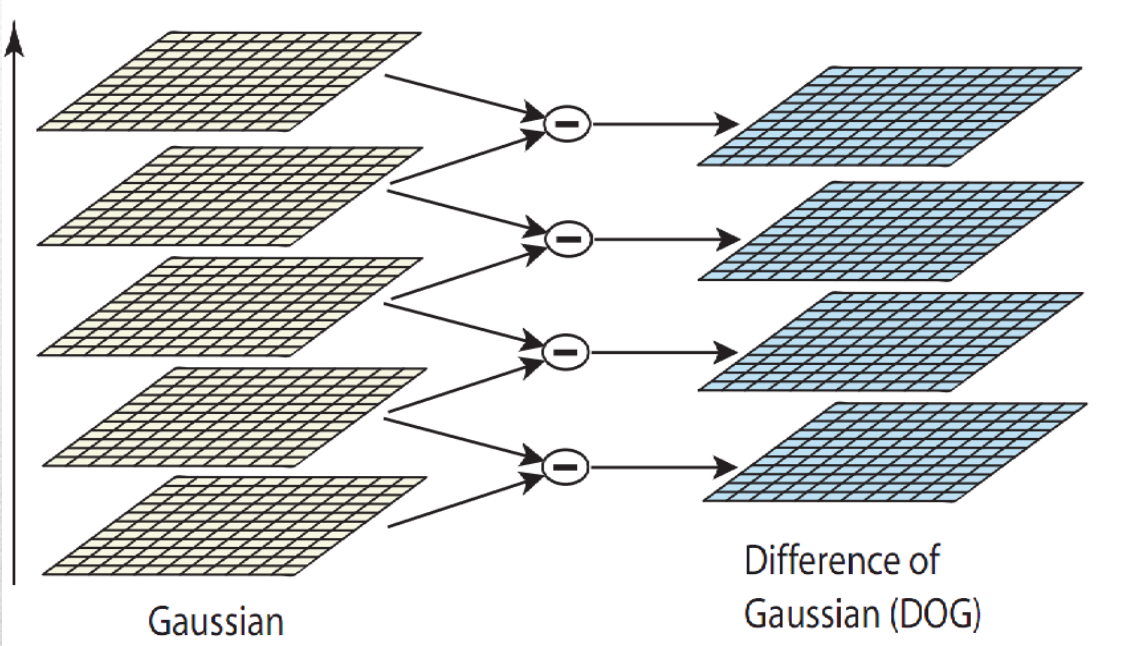
o Xem xét sự khác nhau của hàm không gian tỷ lệ giữa 2 tỷ lệ sai lệch nhau k lần

- Bộ lọc DoG:

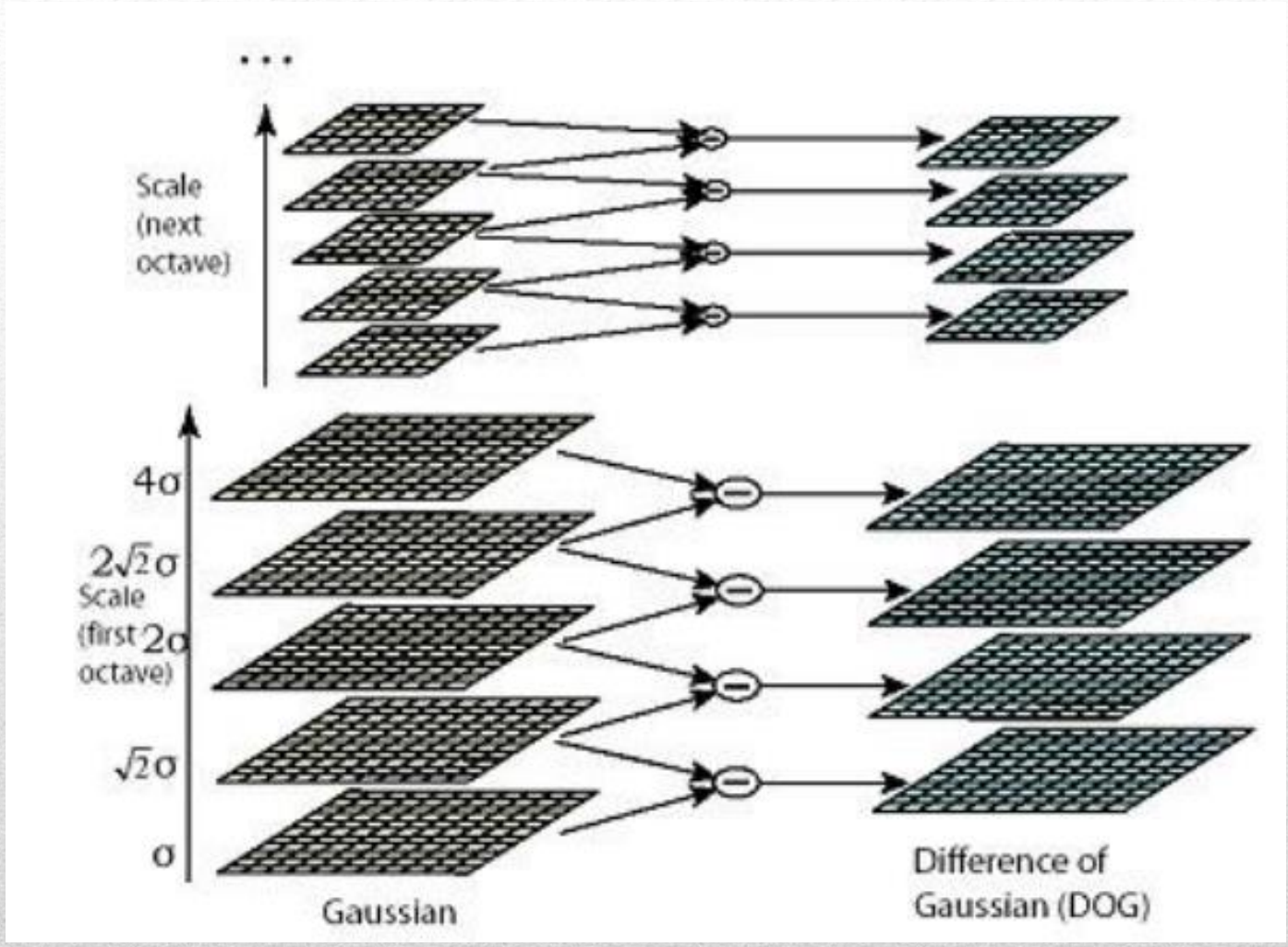
$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma). \end{aligned}$$

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 1: Phát hiện điểm cực trị (cont)



Định nghĩa bộ lọc DoG



Biểu đồ mô phỏng việc tính toán các DoG ảnh từ các ảnh kề mờ

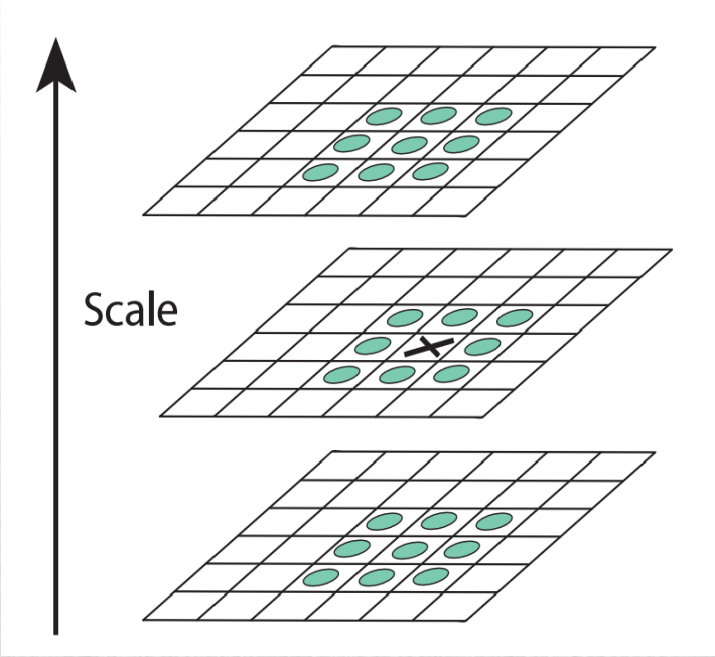
3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 1: Phát hiện điểm cực trị …

- **Xác định điểm hấp dẫn tiềm năng:**

  So sánh mỗi điểm ảnh được xác định bởi hàm DoG với:

  - *8 điểm ảnh xung quanh ở cùng tỷ lệ*
  - *9 điểm ảnh xung quanh ở các tỷ lệ liền trước*
  - *9 điểm ảnh xung quanh ở tỷ lệ liền sau.*



Hình Xác định điểm cực trị

Chọn là điểm hấp dẫn tiềm năng nếu nó đạt giá trị cực trị

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 2: Định vị điểm hấp dẫn

- Phân tích điểm hấp dẫn tiềm năng: lấy các thông tin về:
  - Vị trí ,
  - Tỷ lệ,
  - Tỷ lệ độ cong cơ sở (pricipal curvature).

- Loại bỏ điểm cực trị không phù hợp:
  - Điểm có độ tương phản thấp (không ổn định khi ảnh bị nhiễu),
  - Điểm ở những vị trí không thuận lợi dọc theo các cạnh.

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 2: Định vị điểm hấp dẫn …..

- Phương pháp:
  - Lấy mẫu: sử dụng các phương pháp lấy mẫu thích hợp để quyết định những điểm làm mẫu cho việc phân tích
  - Khai triển Taylor cho bộ lọc DoG tại các điểm mẫu:

$$D(\mathbf{x}) = D + \frac{\partial D}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}$$

$\mathbf{x} = (x, y, \sigma)^T$ Độ dịch so với các điểm lân cận của điểm lấy mẫu

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 2: Định vị điểm hấp dẫn ..

- Vùng chứa điểm hấp dẫn được xác định qua $\hat{\mathbf{x}}$

$$\hat{\mathbf{x}} = -\frac{\partial^2 D}{\partial \mathbf{x}^2}^{-1} \frac{\partial D}{\partial \mathbf{x}}$$

- Nếu $\hat{\mathbf{x}}$ > 0.5 : điểm hấp dẫn nằm ở gần điểm lấy mẫu khác.
- Thực hiện tiếp tục với các điểm lấy mẫu khác.
- Những điểm có $\hat{\mathbf{x}}$ thỏa mãn (< 0.5) được thêm vào tập hợp mẫu tốt nhất, tiếp tục phân tích tiếp.

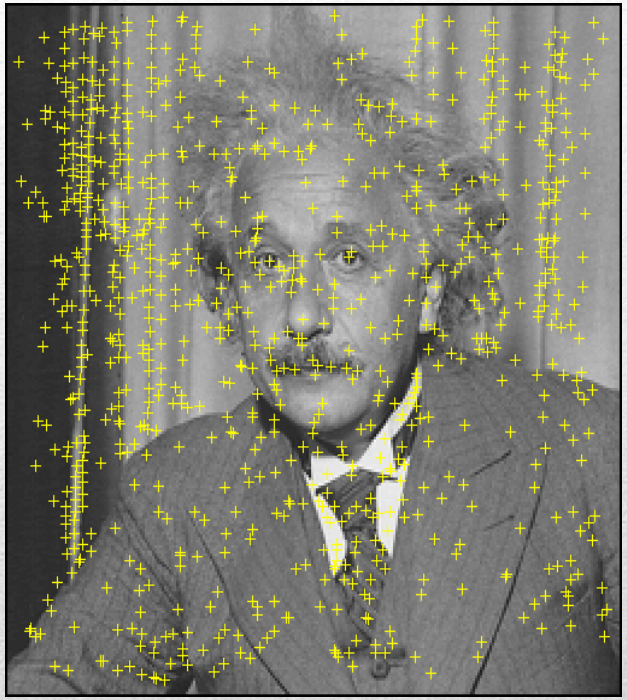
Bước 2: Định vị điểm hấp dẫn

- Dùng $D(\hat{\mathbf{x}})$ để loại những điểm cực trị không ổn định (độ tương phản thấp).
- Thay $\hat{\mathbf{x}}$ vào $D(\mathbf{x})$ ta được:

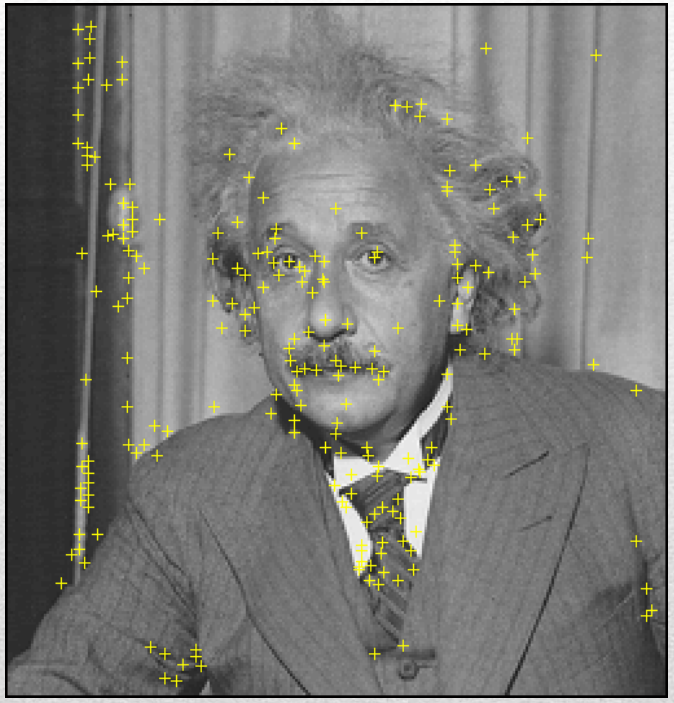
$$D(\hat{\mathbf{x}}) = D + \frac{1}{2}\frac{\partial D}{\partial \mathbf{x}}^T \hat{\mathbf{x}}$$

Nếu $D(\hat{\mathbf{x}})$ < 0.03 thì điểm lấy mẫu đó sẽ bị loại.

3.1. Đặc trưng cục bộ bất biến SIFT …



Trước



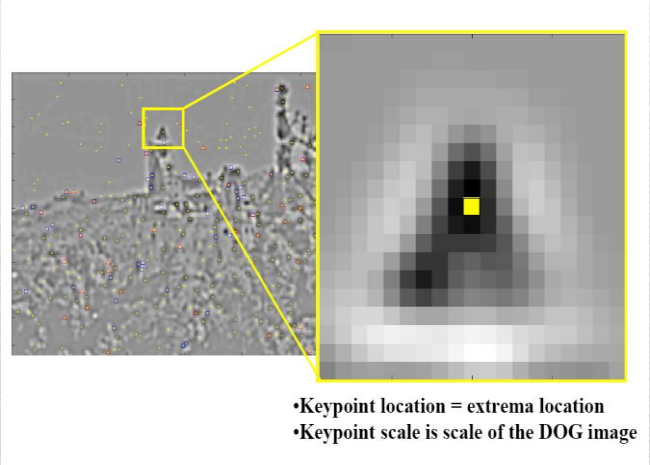
Sau

Định vị điểm hấp dẫn

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 3: Xác định hướng cho các điểm hấp dẫn

- Dựa vào hướng của điểm hấp dẫn, biết được điểm hấp dẫn bất biến với sự quay ảnh.
- Tại mỗi điểm hấp dẫn, trích xuất một ảnh Gaussian (khung Gaussian) chứa các điểm lân cận điểm hấp dẫn đó.



Hình - Trích ảnh Gaussian

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 3: Xác định hướng cho các điểm hấp dẫn

- Tính toán hướng và độ lớn cho điểm hấp dẫn:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

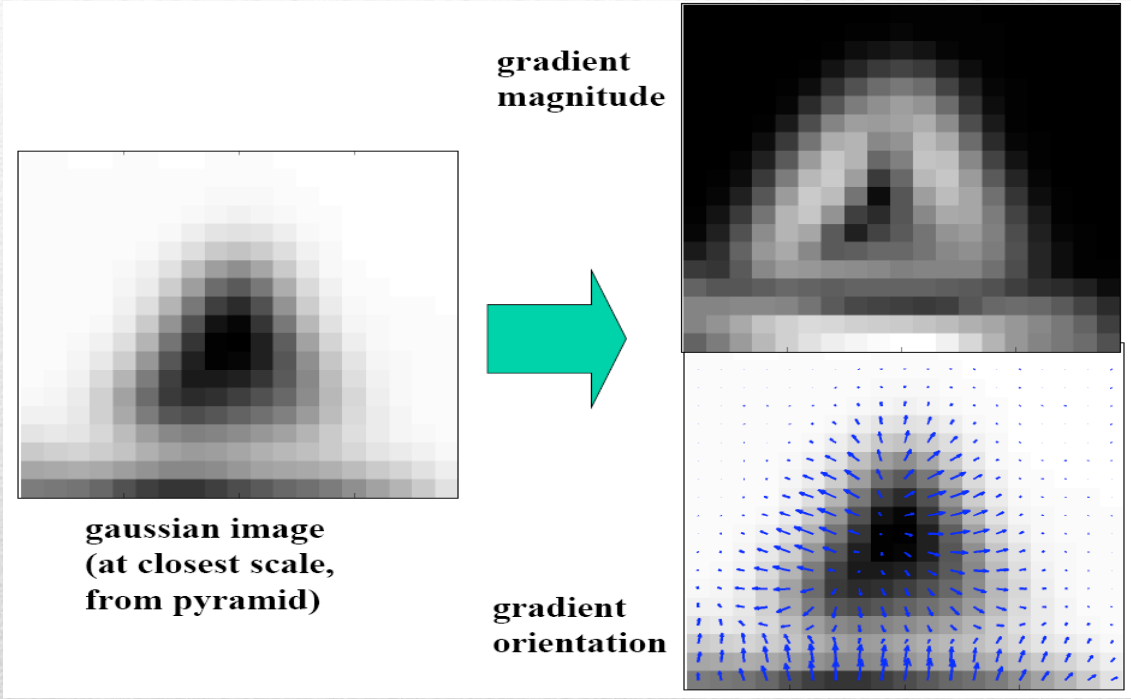
$m(x,y)$ Độ lớn của vector định hướng

$\theta(x,y)$ Hướng của vector định hướng (biểu diễn qua góc **Θ)**

$L(x,y)$ Ảnh Gaussian ở tỷ lệ nhỏ nhất

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 3: Xác định hướng cho các điểm hấp dẫn …

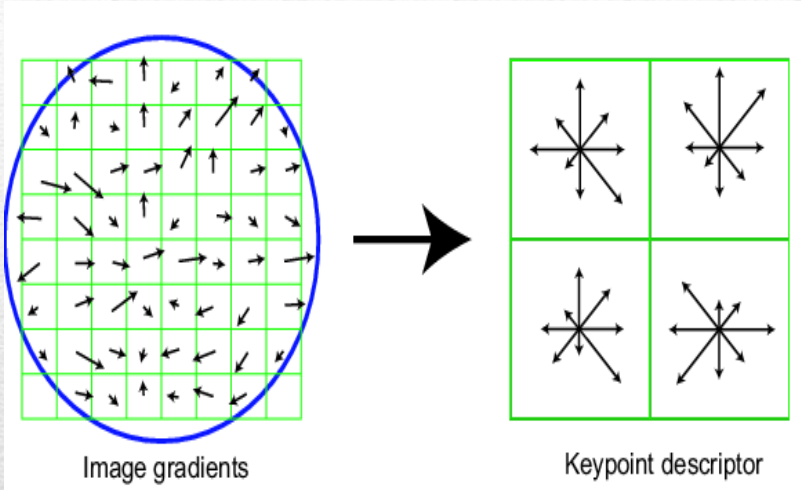


Hình - Tính độ lớn và hướng của gradient

3.1. Đặc trưng cục bộ bất biến SIFT …

Bước 4 : Mô tả điểm hấp dẫn

- Trích các ảnh Gaussian chứa 8×8 điểm hấp dẫn.
- Xét trên mỗi khu vực nhỏ 4×4 điểm hấp dẫn:
  - Tổng hợp các vector định hướng của các điểm trong khu vực.
    - Chung gốc.
    - Độ dài mỗi vector tương ứng độ lớn gradient m của nó.



Hình - Tạo mảng lưu lược đồ định hướng

- Tạo ra mảng chứa các lược đồ định hướng tổng hợp đó
  Số chiều = 8 hướng × (4×4) điểm hấp dẫn = 128 chiều

3.1. Đặc trưng cục bộ bất biến SIFT …

Độ đo tương đồng:

- Độ đo Cosin

$$d(x,y) = \frac{x \cdot y}{||x||_2 \, ||y||_2}$$

- Khoảng cách góc

$$d(x,y) = \cos^{-1}(x \cdot y)$$

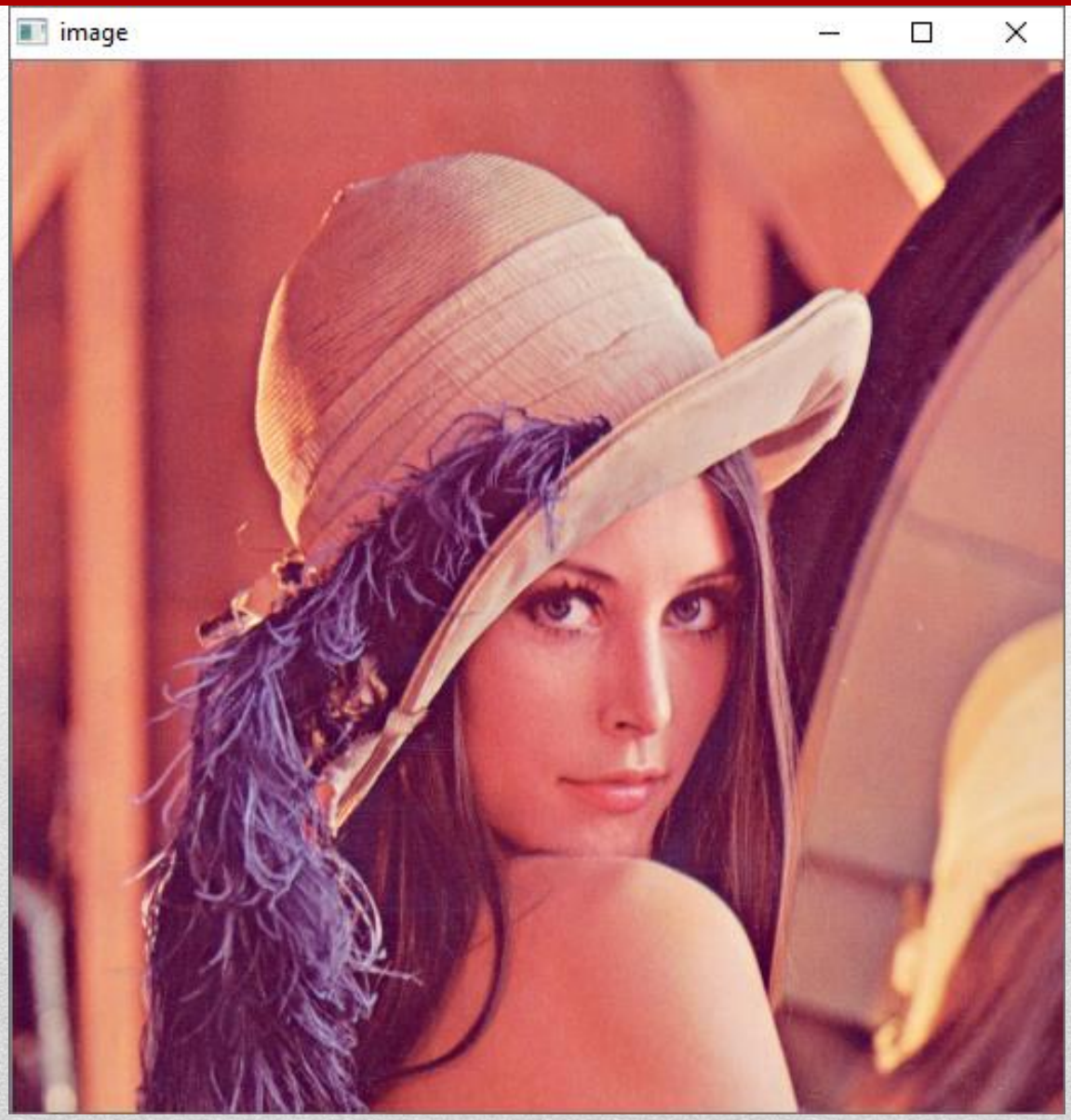
- Khoảng cách Euclid :

$$d(x,y) = \sqrt{\sum_i (x_i - y_i)^2}$$

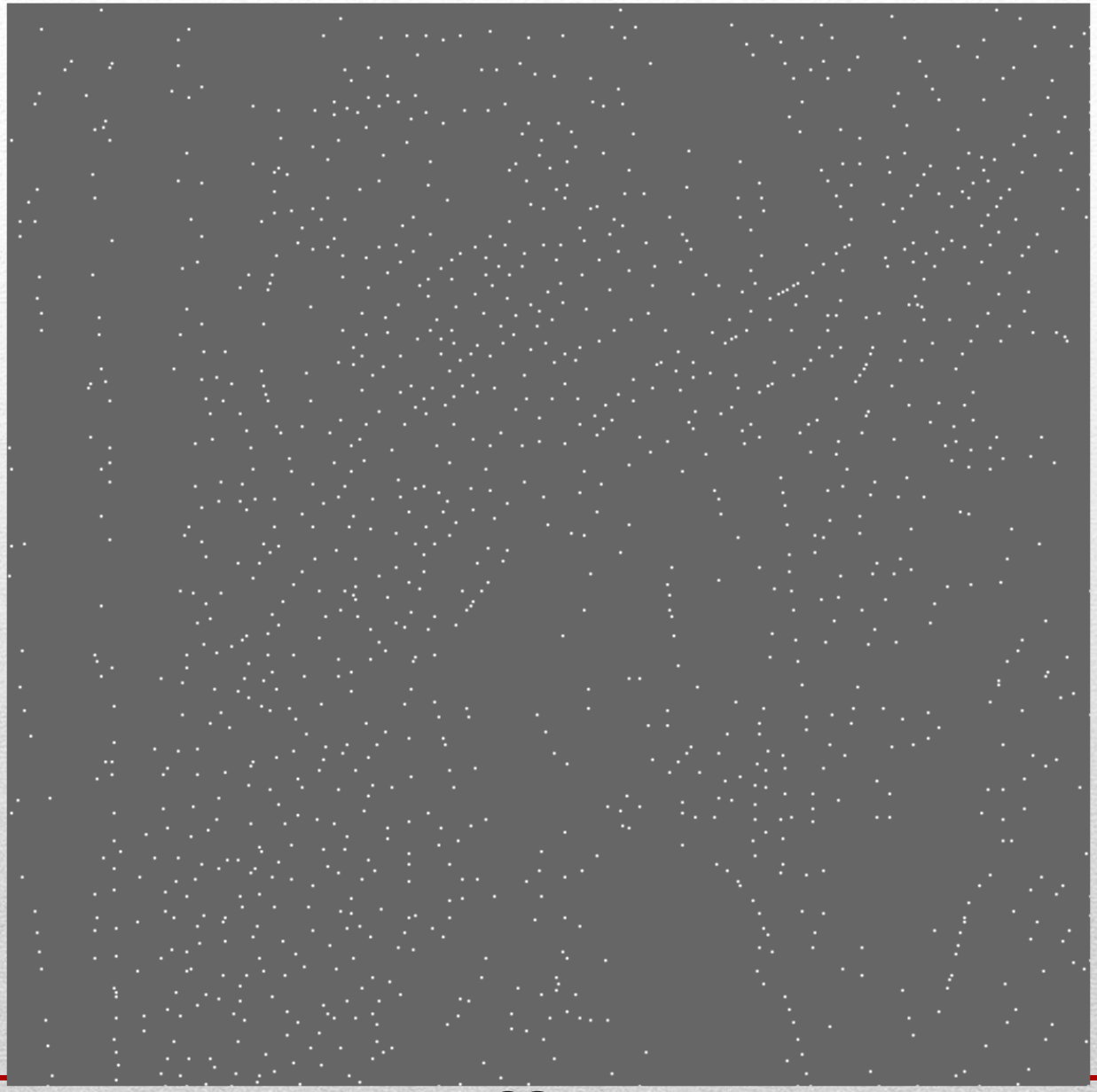
- Khoảng cách city-block:

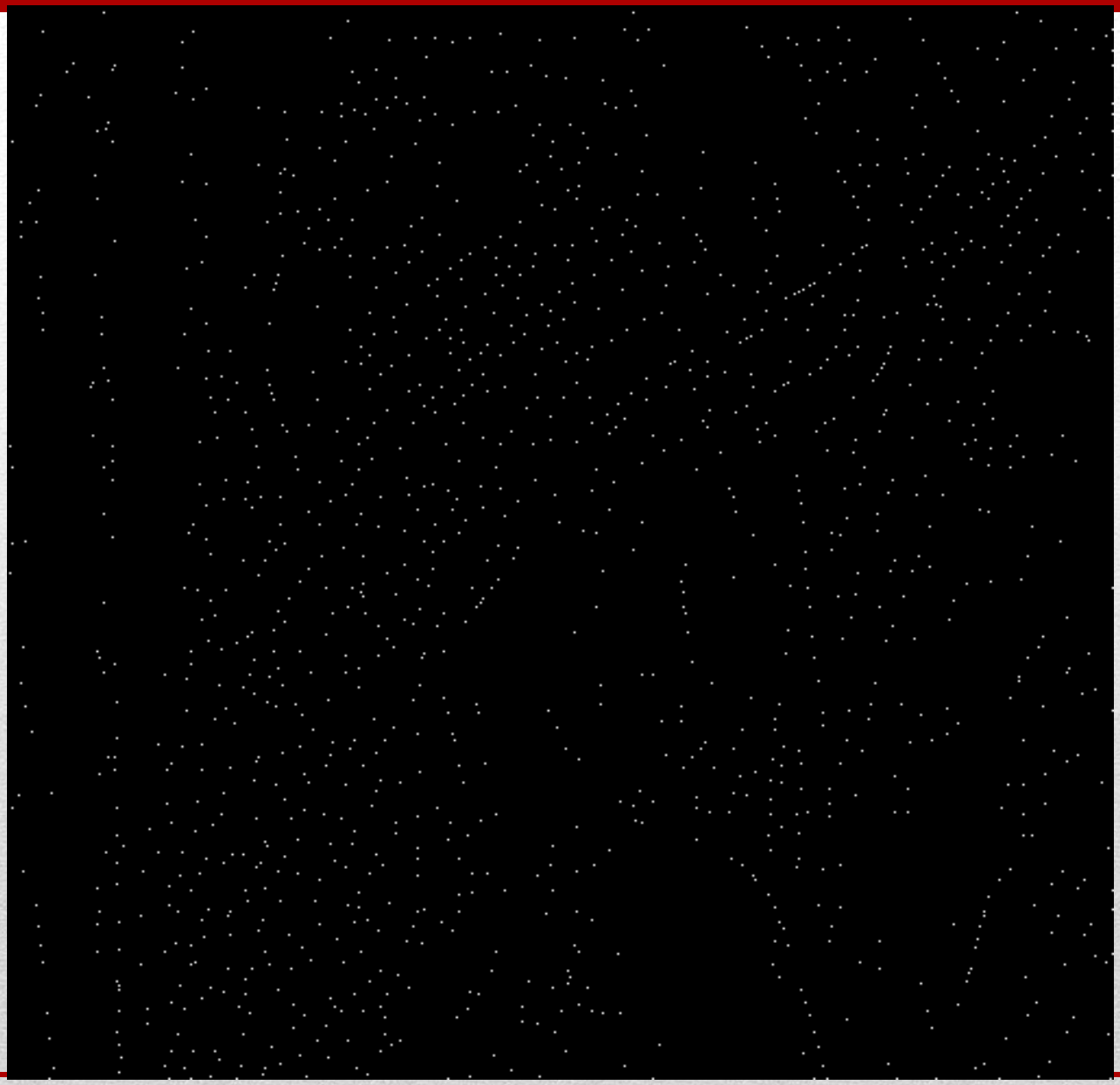
$$d(x,y) = \sum_i |x_i - y_i|$$

x, y là 2 vector đặc trưng



image





Code Matlab

Chạy Demo trong thư mục : vijay_ti_sift

Code Python

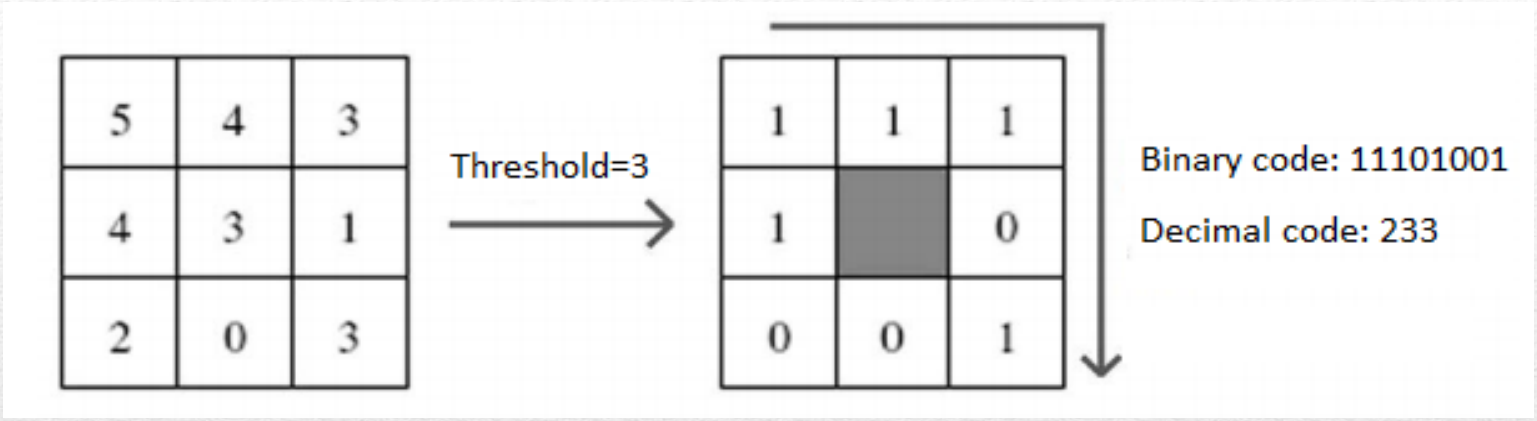
...\opencv\sources\samples\python\asift.py

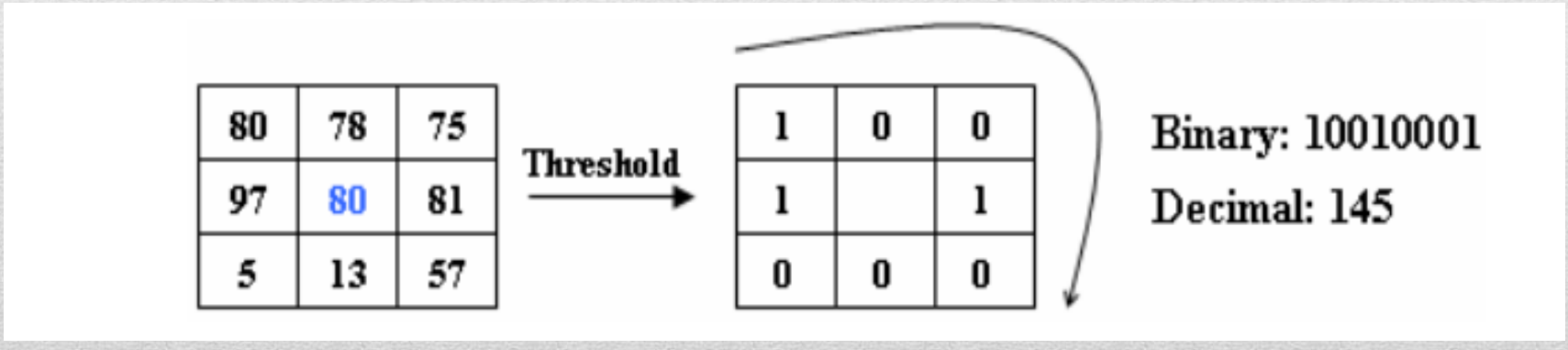
3.2. Phương pháp mẫu nhị phân cục bộ LBP (Local Binary Patterns)

Phương pháp mẫu nhị phân cục bộ LBP

- Ojala và các đồng nghiệp đã giới thiệu phương pháp LBP và chỉ ra khả năng phân tích cao của chúng cho sự phân lớp vân.
- LBP là một toán tử 3×3, nó tổng quát hóa cấu trúc không gian cục bộ của một ảnh.
  - Tại một vị trí pixel (x_c, y_c) cho trước, LBP được định nghĩa như một chuỗi nhị phân có trật tự dựa trên sự so sánh giá trị độ xám của pixel trung tâm (x_c, y_c) và 8 pixel lân cận của nó.
  - Như vậy mỗi pixel sẽ được biểu diễn bởi một chuỗi nhị phân, đây là giá trị của pixel trung tâm trong **biểu diễn bằng toán tử LBP.**

Ví dụ tính toán LBP





Giá trị thập phân của của chuỗi LBP có thể được biểu diễn như sau:

$$LBP(x_c, y_c) = \sum_{n=0}^{7} s(g_n - g_c)2^n$$

- g_c là giá trị độ xám của pixel trung tâm (x_c, y_c),
- g_n là giá trị độ xám của pixel thứ n trong lân cận 8 của pixel trung tâm.
- Hàm s_n được định nghĩa như sau:

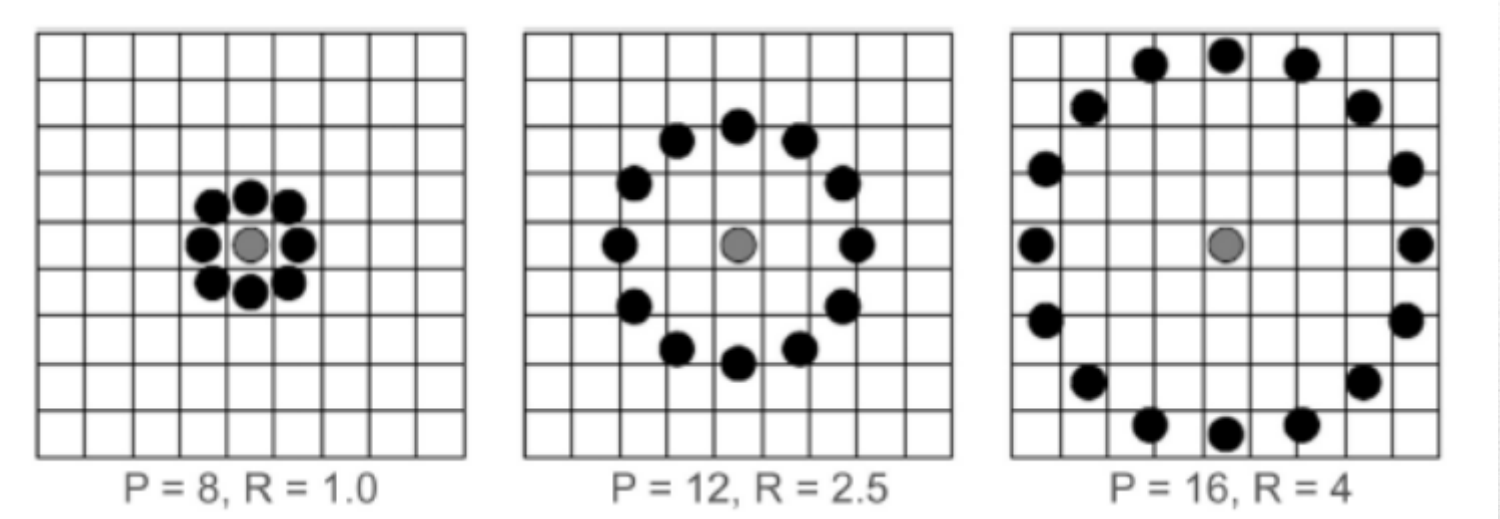
$$s(x) = \begin{cases} 1 & if \quad x \geq 0 \\ 0 & if \quad x < 0 \end{cases}$$

Bằng cách định nghĩa này, toán tử LBP bất biến

- Đối với sự biến đổi độ sáng đều
- Bảo toàn trật tự mật độ các pixel trong một lân cận cục bộ.

Sau đó, Ojala và các đồng nghiệp đã mở rộng toán tử LBP đến một lân cận tròn với các bán kính khác nhau.

- $LBP_{P, R}$ kí hiệu xem xét P pixels lân cận trên một vòng tròn có bán kính R.



Minh họa toán tử LBP mở rộng với các giá trị P và R khác nhau Giá trị các pixel được nội suy cho các điểm không nằm trong tâm của một pixel

Nếu tọa độ của pixel tâm là (x_c, y_c) thì tọa độ của P pixel lân cận trên đường tròn tâm (x_c, y_c) bán kính R (tính theo đường tròn lượng giác) là:

$$x_p = x_c + R\cos(2\Pi p / P)$$
$$y_p = y_c + R\sin(2\Pi p / P), \quad p = \{0, 1, \ldots, P-1\}$$

- Hầu hết các thông tin vân (texture) trong ảnh được chứa trong một tập con của các mẫu LBP.
- Các mẫu này được gọi là các mẫu đồng nhất (uniform patterns).
  - Các mẫu đồng nhất chứa tối đa hai sự chuyển đổi bit từ 0 đến 1 hoặc từ 1 đến 0.

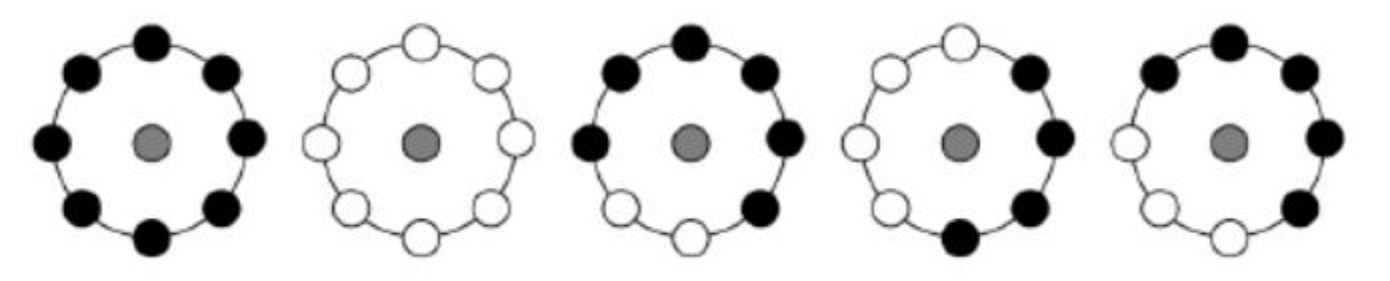
    11111111, 00000110 hoặc 10000111 là ví dụ cho các mẫu đồng nhất.

  - Toán tử $LBP_{P,R}$ đồng nhất được kí hiệu là $LBP_{P,R}^{u2}$

- Với chuỗi LBP có chiều dài P thì số mẫu có tối đa hai sự chuyển đổi (mẫu đồng nhất là) P(P-1).
- Có hai mẫu không có sự chuyển đổi nào là mẫu toàn 0 hoặc 1.

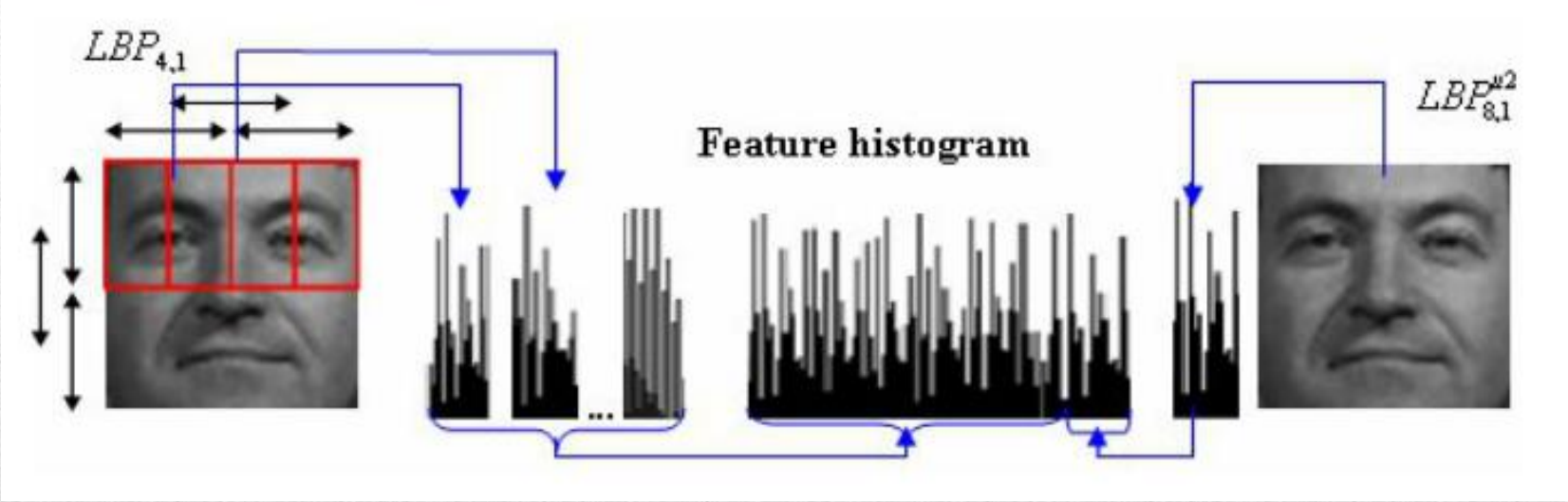
- Việc sử dụng mẫu LBP đồng nhất có hai lợi điểm quan trọng.
  - Thứ nhất là tiết kiệm bộ nhớ, vì trường hợp LBP tổng quát chúng ta có 2^P mẫu có thể, nhưng nếu chỉ xét mẫu đồng nhất thì chúng ta có tối đa là P(P+1) + 2 nếu chúng ta sử dụng toán tử $LBP_{P,R}^{u2}$.
  - Lợi điểm thứ hai là LBP^{u2} chỉ phát hiện những mẫu vân cục bộ quan trọng như các điểm, các điểm cuối đường thẳng, biên cạnh và các góc..

Hình minh họa một số mẫu vân quan trọng có thể được phát hiện bởi LBP^{u2}

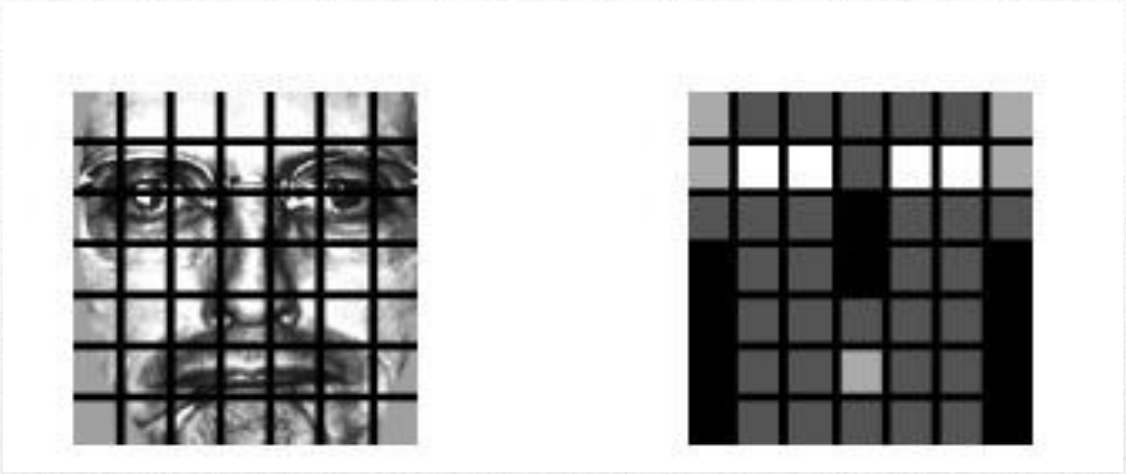


Từ trái sang phải, các mẫu vân cơ bản: *điểm chấm, điểm chấm nhạt, điểm cuối đường thẳng, biên cạnh, góc* được phát hiện bởi LBP^{u2}

- Bởi vì khả năng phân tách và chi phí tính toán thấp, LBP trở nên rất phổ biến.
- Trong nhận dạng mẫu LBP đã được áp dụng cho:
  - phát hiện khuôn mặt,
  - xác thực khuôn mặt,
  - truy vấn ảnh.



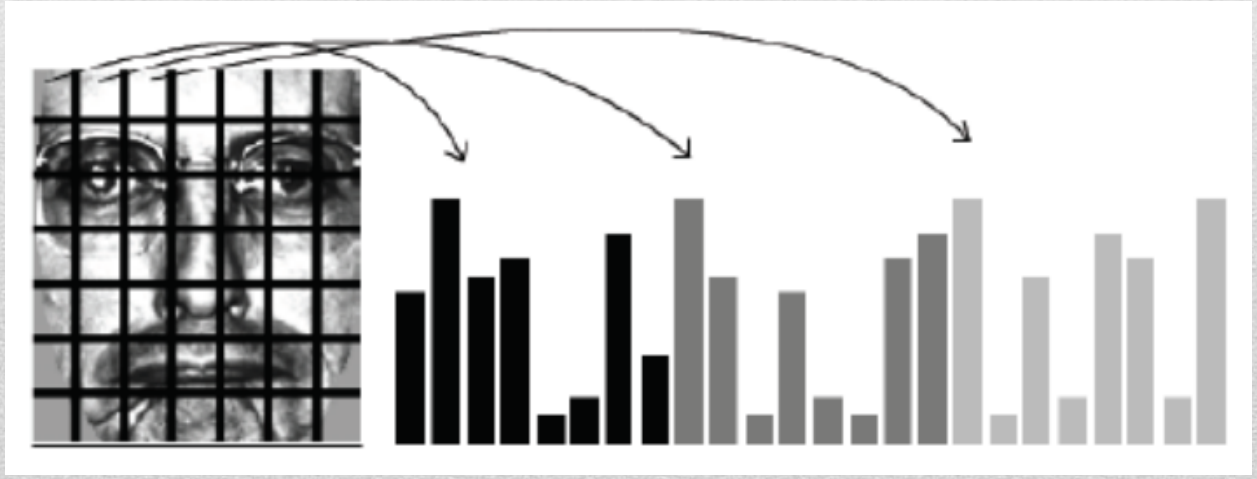
Biểu diễn khuôn mặt: histogram đặc trưng được tạo bằng cách kết hợp histogram LBP cục bộ và histogram LBP toàn cục.



Weight matrix

2	1	1	1	1	1	2
2	4	4	3	4	4	2
1	1	1	2	1	1	1
0	1	1	2	1	1	0
0	1	1	1	1	1	0
0	1	1	2	1	1	0
0	1	1	1	1	1	0

Computing of face object histogram



Code Matlab

Chạy tệp facedetect.m trong thư mục LBP

3.3. Phương pháp biến đổi sóng nhỏ Gabor (Gabor wavelets)

Biến đổi sóng nhỏ Gabor

Hàm Gabor Wavelet được đề xuất bởi Dennis Gabor như là 1 công cụ dùng để phát hiện tín hiệu từ tạp nhiễu.

$$\Psi(t,t_0,\omega) = e^{-\sigma(t-t_0)^2}.e^{-i\omega(t-t_0)}$$

Sau đó, hàm Gabor Wavelet được Daugman tổng quát hóa thành dạng 2 chiều. Bộ lọc Gabor 2 chiều hiện nay được sử dụng rộng rãi trong nhiều ứng dụng xử lý ảnh trên máy tính.

$$\Psi_i(\vec{x}) = \frac{\left\|\vec{k_i}\right\|^2}{\sigma^2} e^{-\frac{\left\|\vec{k_i}\right\|^2\left\|\vec{x}\right\|^2}{2\sigma^2}} \left[e^{j\vec{k_i}\vec{x}} - e^{-\frac{\sigma^2}{2}} \right]$$

Mỗi Ψ_i là một sóng phẳng đặc trưng bởi vector hàm sóng Gauss $\overrightarrow{k_i}$, với σ là độ lệch tiêu chuẩn của hàm sóng Gauss.

Tần số tiêu chuẩn của bộ lọc Gabor thứ i được cho bởi hàm sóng đặc trưng:

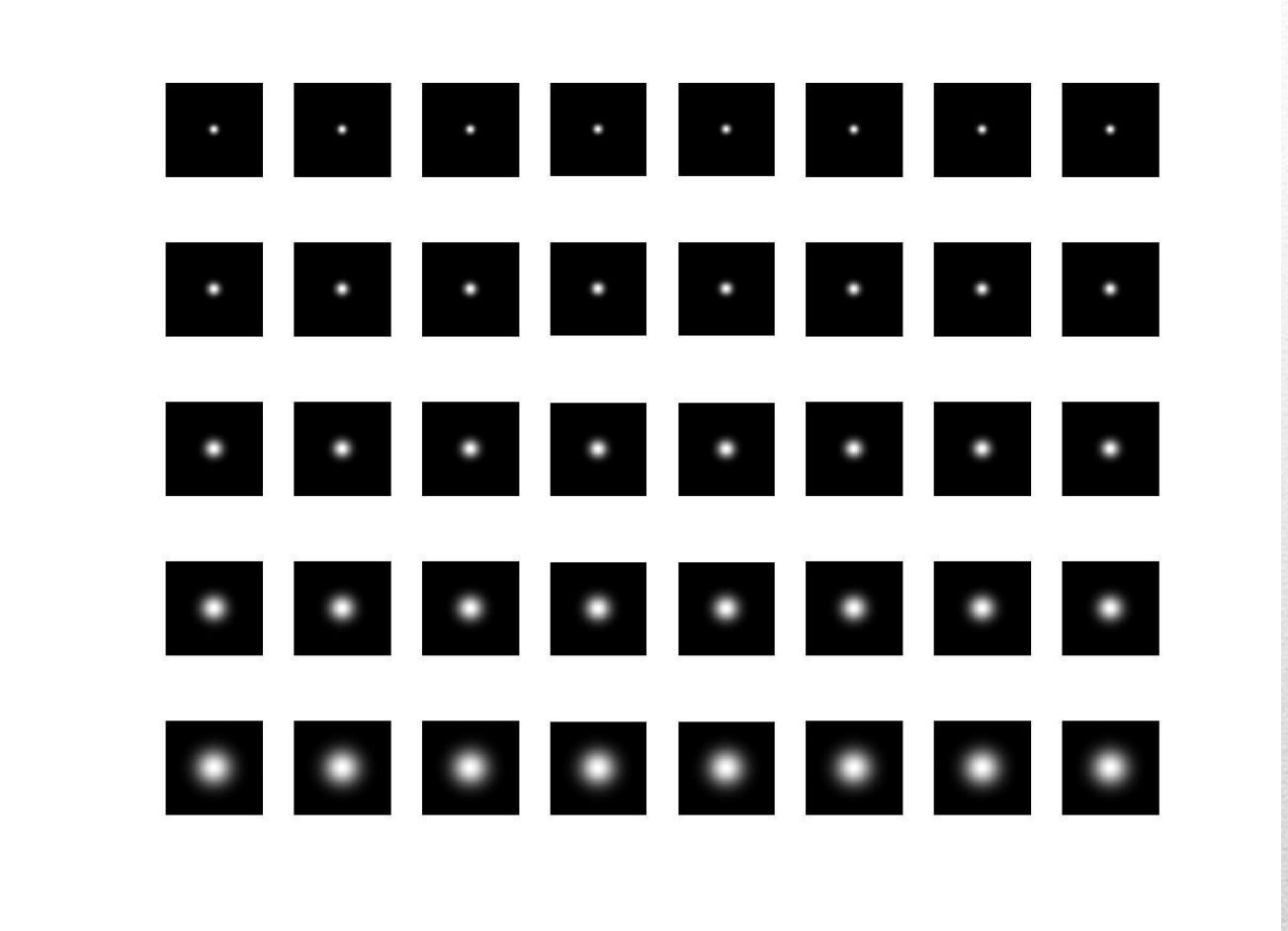
$$k_i = \frac{K_{\max}}{f^v} e^{iu\pi/8}$$

Một số thông số quan trọng đối với biến đổi Gabor Wavelet:

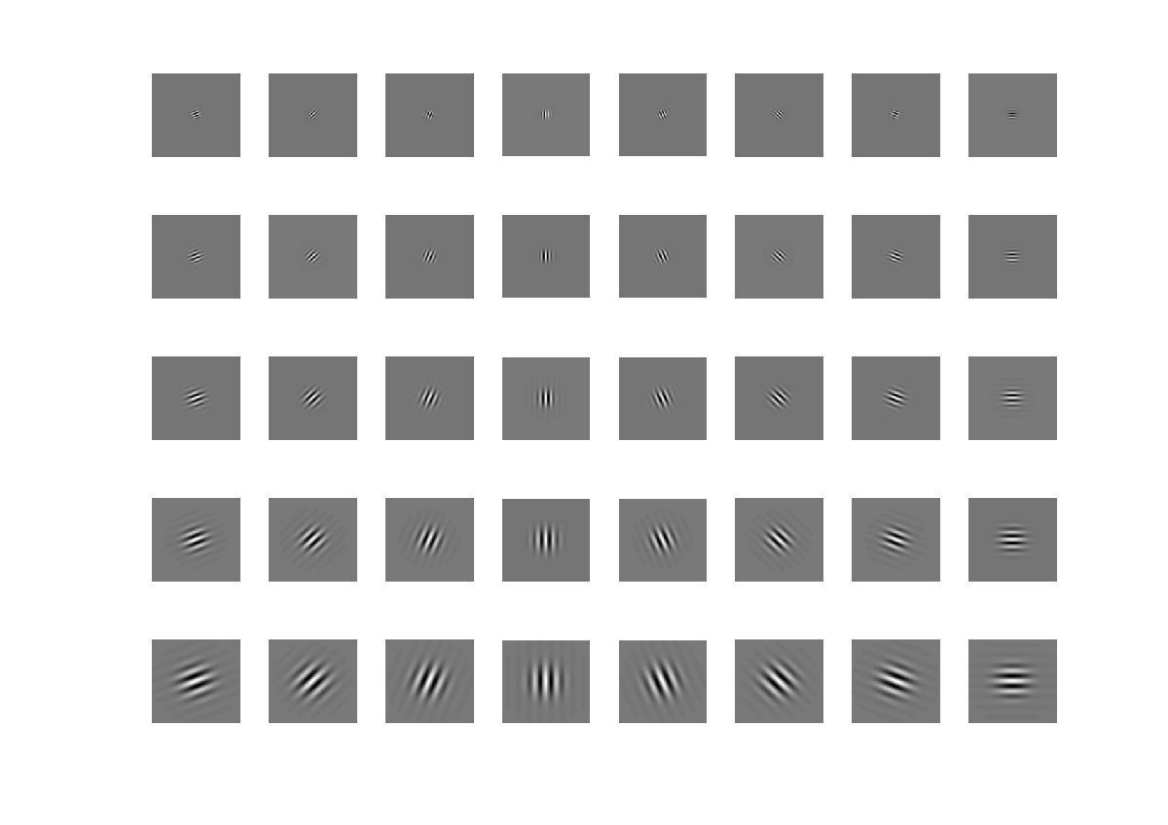
- θ mô tả hướng của Wavelet. Thông số này khiến cho Wavelet quay quanh tâm của nó. Trong đa số các trường hợp θ có giá trị trong khoảng 0 đến π . Ở đây ta sẽ chọn θ từ π với bước nhảy π/8 : θ = uπ/8 với u=1,....,8

- λ mô tả bước sóng của sóng sin hoặc tần số trung tâm của Wavelet. Wavelet có bước sóng lớn sẽ đáp ứng với những thay đổi cường độ của ảnh. Wavelet có bước sóng nhỏ sẽ đáp ứng với những cạnh.

Biểu diễn dạng ảnh của bộ lọc Gabor Wavelet với 5 tỉ lệ và 8 hướng:



Hình Biểu diễn **biên độ** của bộ lọc Gabor Wavelet



Phần thực của bộ lọc Gabor Wavelet

Biểu diễn hình ảnh bằng Gabor Wavelet

Biến đổi Gabor Wavelet của một ảnh I cho trước được cho bởi công thức sau:

$$R_i(\vec{x}) = \int I(\vec{x}')\Psi_i(\vec{x} - \vec{x}')d\vec{x}'$$

Với $I(\vec{x})$ là cường độ ảnh tại vị trí $\vec{x}$

$$\Psi_i(\vec{x}) = \frac{\|\vec{k_i}\|^2}{\sigma^2} e^{-\frac{\|\vec{k_i}\|^2\|\vec{x}\|^2}{2\sigma^2}} \left[e^{j\vec{k_i}\vec{x}} - e^{-\frac{\sigma^2}{2}} \right]$$

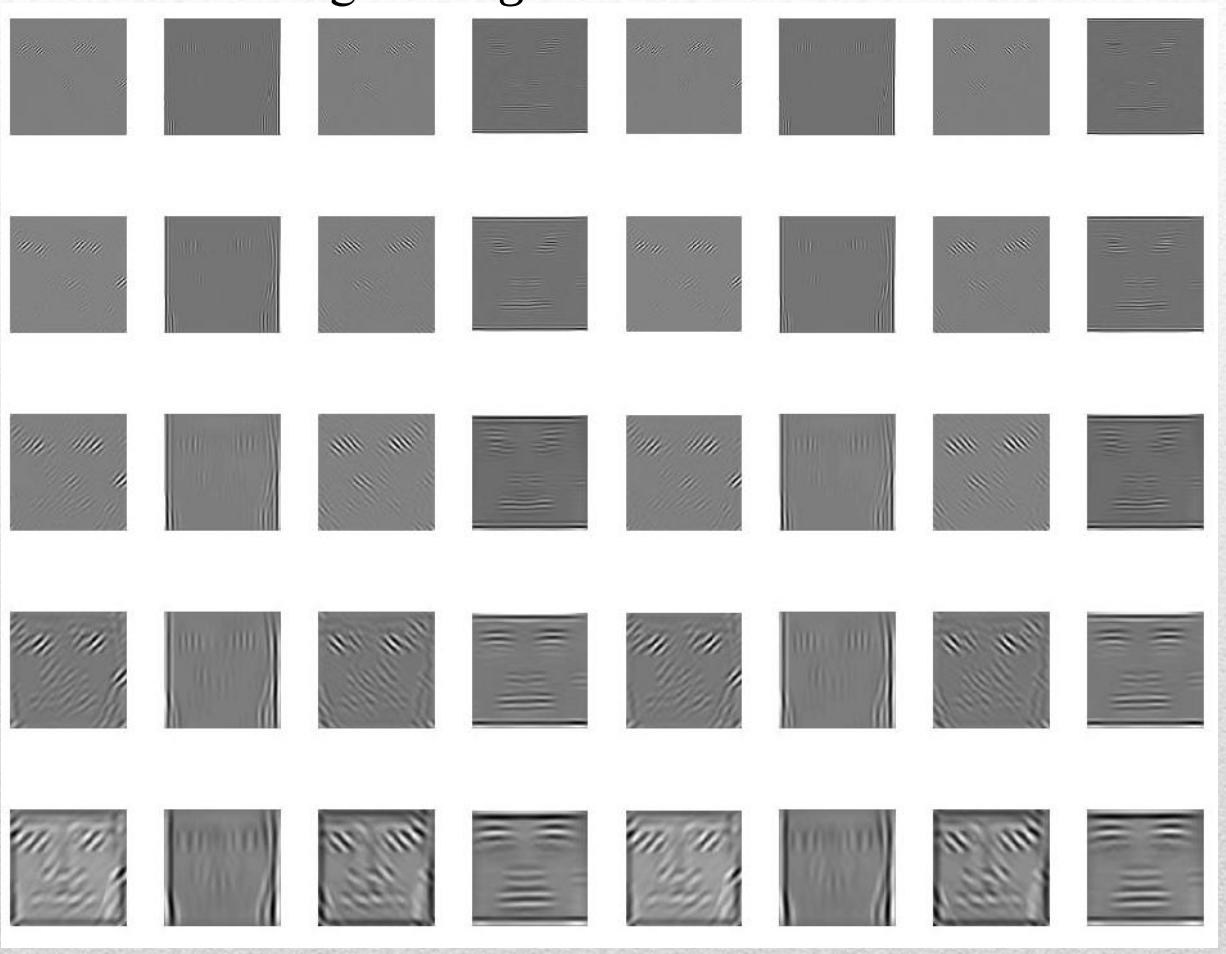
Một hình ảnh biểu diễn bởi biến đổi Gabor Wavelet cho phép mô tả **cấu trúc của không gian tần số** và **mối liên hệ không gian**.

Chập hình ảnh với tổ hợp bộ lọc Gabor:

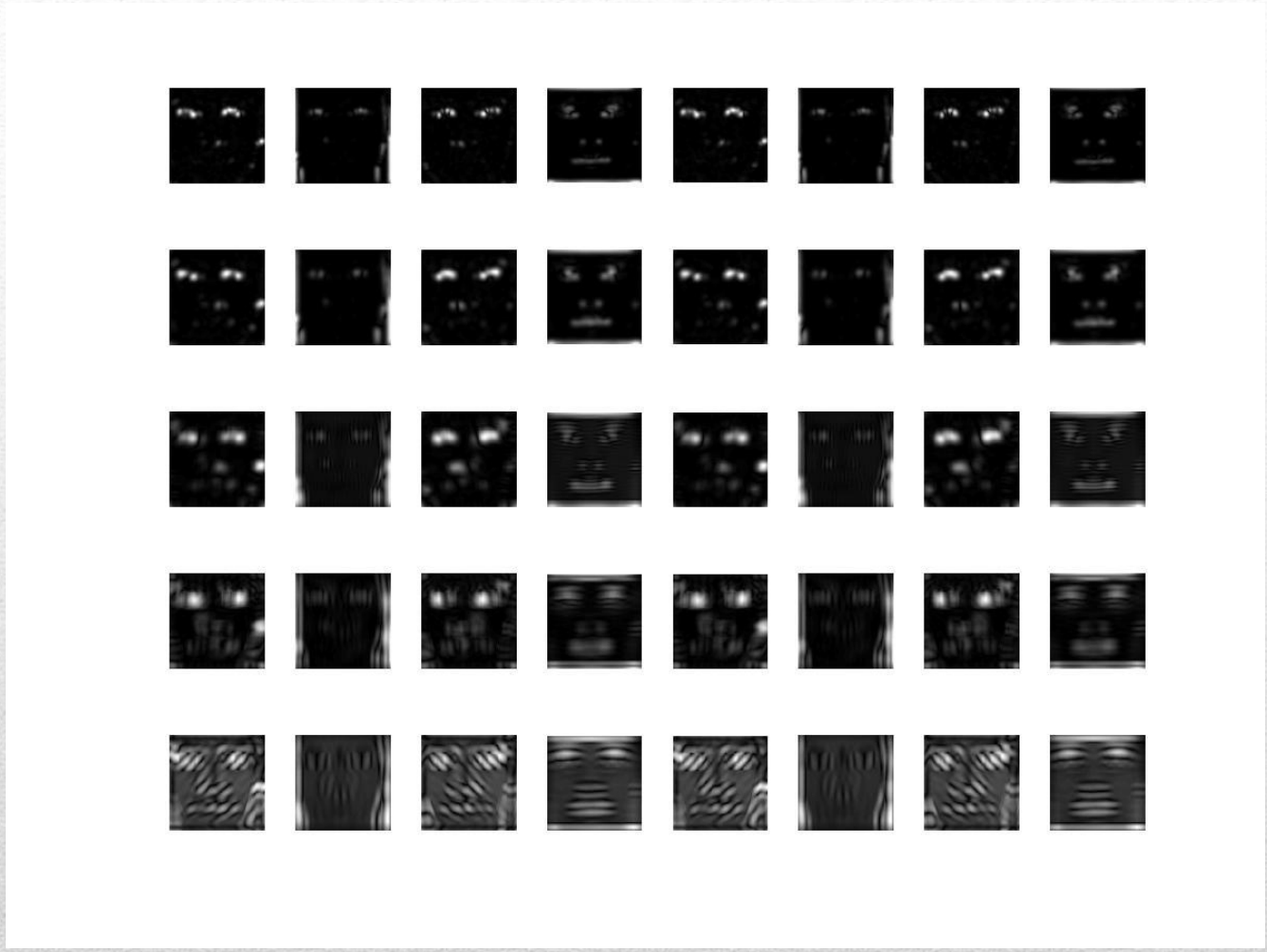
- cho 5 không gian tần số (v = 0,...,4) và
- 8 hướng (u = 1,...,8) thu được toàn bộ mật độ phổ tần số, cả biên độ và pha.

Cho ảnh gốc:

Biểu diễn ảnh gốc bằng Gabor Wavelet:



Phần thực



Phần biên độ

ỨNG DỤNG GABOR WAVELETS TRONG THUẬT TOÁN NHẬN DẠNG KHUÔN MẶT

- Nhận dạng khuôn mặt dùng biến đổi Gabor Wavelet là việc sử dụng các vector đặc trưng để mã hóa khuôn mặt.
- Thay vì dùng biểu đồ các điểm đặc trưng trên khuôn mặt, các điểm có năng lượng cao sẽ được sử dụng để so sánh các khuôn mặt.
- Điều này không chỉ giúp giảm khối lượng tính toán mà còn tăng độ chính xác của thuật toán do không cần phải xác định các điểm đặc trưng trên khuôn mặt bằng tay

Matlab Code

- Chạy tệp GaborExample.m trong thư mục Gabor

  (thử nghiệm biến đổi Gabor)

- Chạy tệp main.m trong thư mục FDV54_gabor

  (ứng dụng phát hiện khuôn mặt)

PHÂN LỚP DỮ LIỆU

Tóm tắt về phân lớp dữ liệu

- **Phân lớp dữ liệu** là một kỹ thuật trong khai phá dữ liệu.

- **Mục đích**: Để dự đoán những nhãn phân lớp cho các bộ dữ liệu hoặc mẫu mới.

- ***Đầu vào***: Một tập các mẫu dữ liệu huấn luyện, với một nhãn phân lớp cho mỗi mẫu dữ liệu

- ***Đầu ra***:Bộ phân lớp dựa trên tập huấn luyện, hoặc những nhãn phân lớp

- Kỹ thuật phân lớp dữ liệu được tiến hành bao gồm 2 bước:

- **Bước 1:** Xây dựng mô hình từ tập huấn luyện
- **Bước 2:** Sử dụng mô hình – kiểm tra tính đúng đắn của mô hình và dùng nó để phân lớp dữ liệu mới.

Bước 1. Xây dựng mô hình

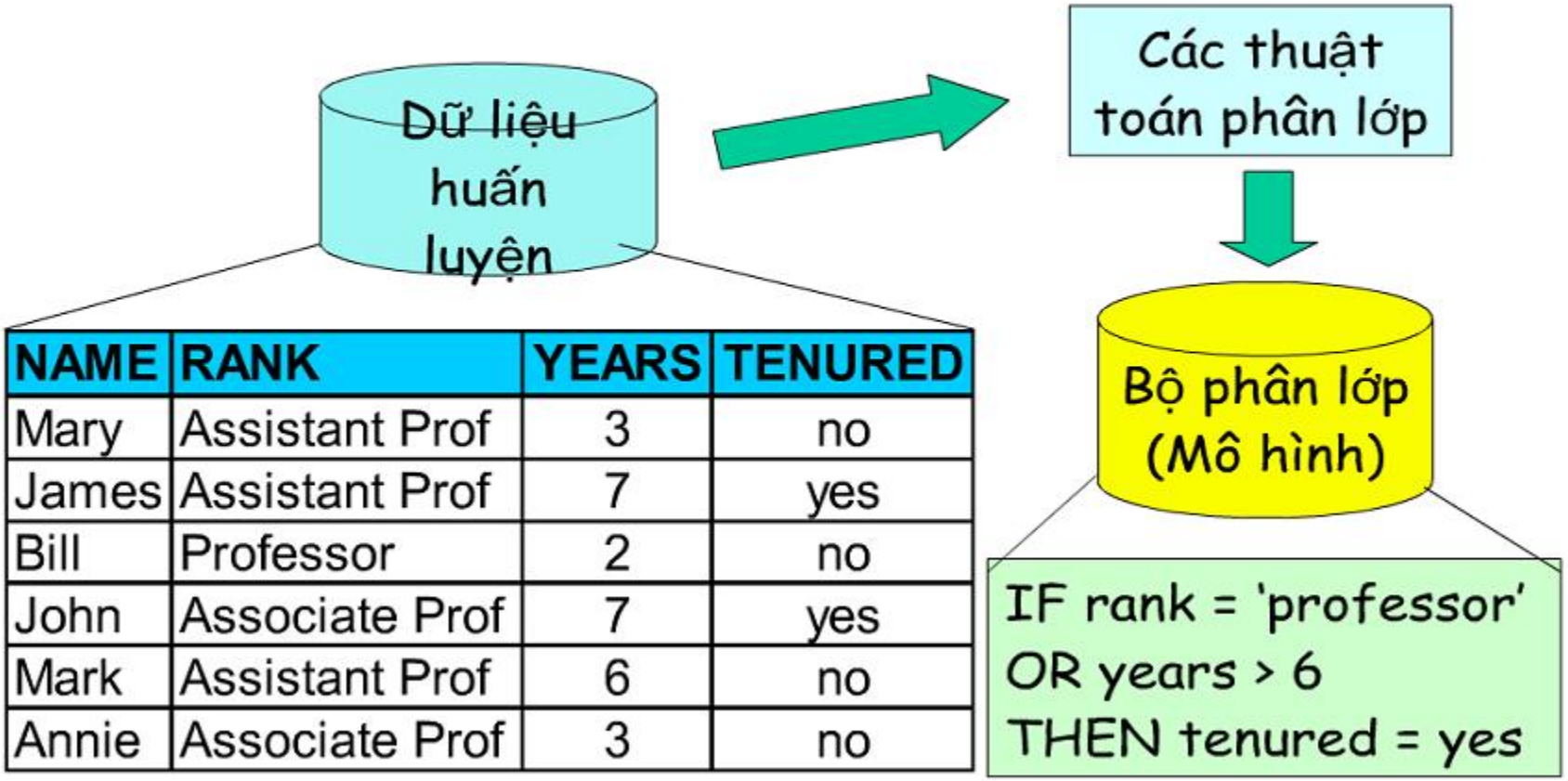
- Mỗi bộ/mẫu dữ liệu được phân vào một lớp được xác định trước.

- Lớp của một bộ/mẫu dữ liệu được xác định bởi thuộc tính gán nhãn lớp

- Tập các bộ/mẫu dữ liệu huấn luyện - tập huấn luyện – được dùng để xây dựng mô hình.

- Mô hình được biểu diễn bởi các luật phân lớp, các cây quyết định hoặc các công thức toán học.

Ví dụ: xây dựng mô hình



NAME	RANK	YEARS	TENURED
Mary	Assistant Prof	3	no
James	Assistant Prof	7	yes
Bill	Professor	2	no
John	Associate Prof	7	yes
Mark	Assistant Prof	6	no
Annie	Associate Prof	3	no

Bước 2: Sử dụng mô hình

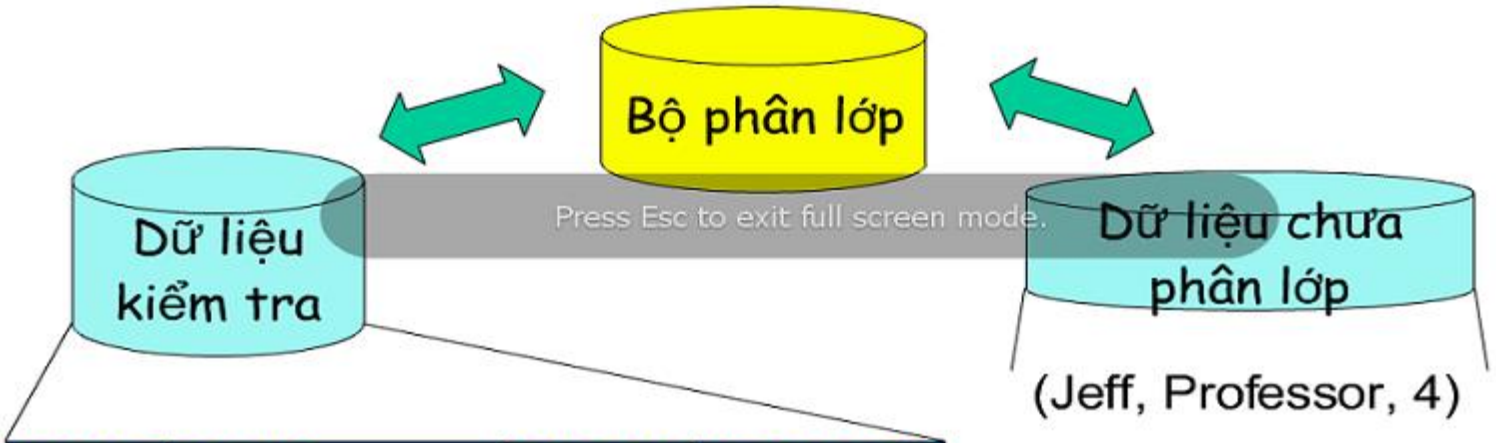
- Phân lớp cho những đối tượng mới hoặc chưa được phân lớp

- Đánh giá độ chính xác của mô hình

+ Lớp biết trước của một mẫu/bộ dữ liệu đem kiểm tra được so sánh với kết quả thu được từ mô hình.

+ Tỉ lệ chính xác bằng phần trăm các mẫu/bộ dữ liệu được phân lớp đúng bởi mô hình trong số các lần kiểm tra.

Ví dụ: sử dụng mô hình



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Lisa	Associate Prof	7	no
Jack	Professor	5	yes
Ann	Assistant Prof	7	yes

- Các thuật toán phân lớp dữ liệu phổ biến:

+ Thuật toán SVM
+ Thuật toán phân lớp K láng giềng gần nhất.

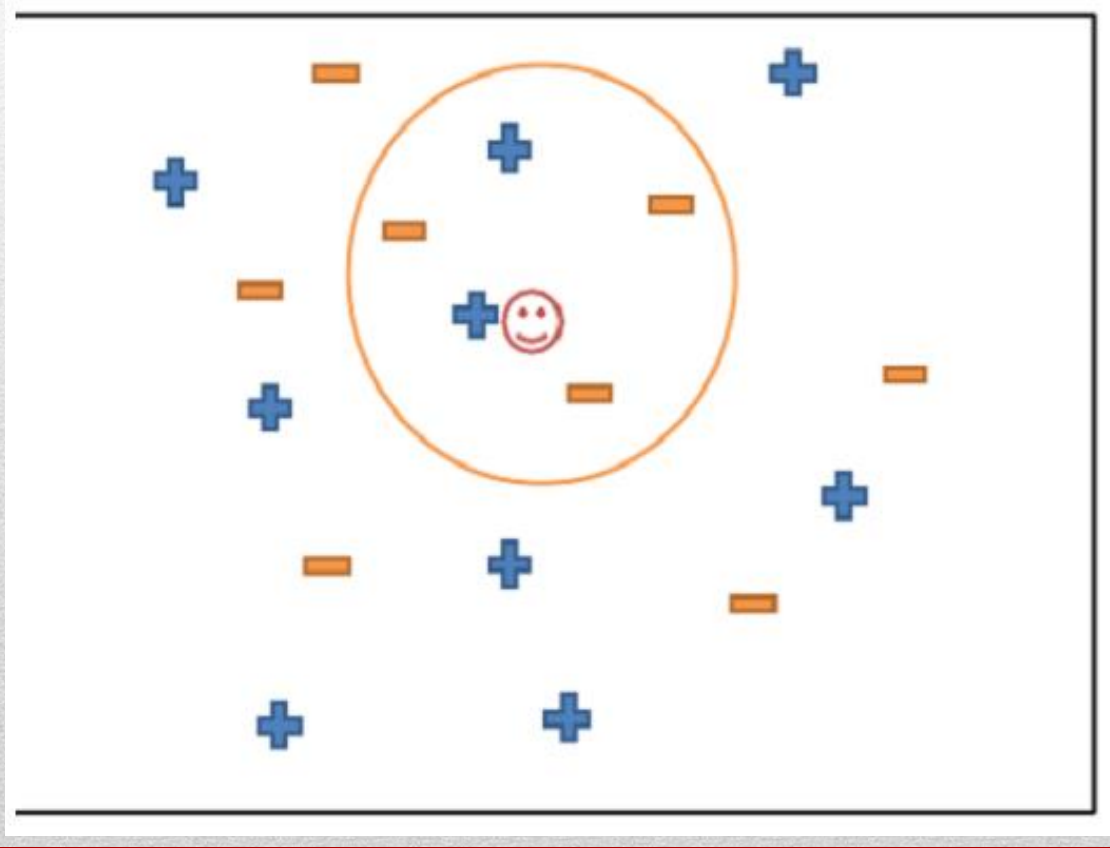
3.4. Phương pháp phân lớp k - láng giềng gần nhất (kNN - K-Nearest Neighbors)

- Được Fix & Hodges đưa ra vào năm 1951
- K-Nearest Neighbors algorithm (kNN) được sử dụng rất phổ biến trong lĩnh vực Data Mining.
- kNN là phương pháp để phân lớp các đối tượng dựa vào khoảng cách gần nhất giữa đối tượng cần xếp lớp (Query point) với tất cả các đối tượng trong Training Data.

Một đối tượng được phân lớp dựa vào k láng giềng của nó (K là số nguyên dương được xác định trước).

Người ta thường dùng khoảng cách Euclidean để tính khoảng cách giữa các đối tượng.

Ví dụ: Trong hình dưới đây, training Data được mô tả bởi dấu (+) và dấu (-), đối tượng cần được xác định lớp cho nó (Query point) là hình mặt cười đỏ. Nhiệm vụ của chúng ta là ước lượng (hay dự đoán) lớp của Query point dựa vào việc lựa chọn số láng giềng gần nhất với nó. Nói cách khác chúng ta muốn biết liệu Query Point sẽ được phân vào lớp (+) hay lớp(-).



Nhận xét:

- **1-Nearest neighbor**: Kết quả là + (Query Point được xếp vào lớp dấu +).
- **2-Nearest neighbor**: Không xác định được lớp cho Query Point. Bởi vì số láng giềng gần nhất là 2. Trong đó có 1 lớp + và 1 lớp -. Vấn đề này sẻ được nói cụ thể ở phần sau.
- **5-Nearest neighbor**: Kết quả là - (Query Point được xếp vào lớp dấu – vì 5 láng giềng gần nhất có 3 đối tượng thuộc lớp -, nhiều hơn lớp + chỉ có 2 đối tượng).

Thuật toán kNN dùng trong phân lớp được mô tả như sau:

1. Xác định giá trị tham số K (số láng giềng gần nhất)
2. Tính khoảng cách giữa đối tượng cần phân lớp (Query Point) với tất cả các đối tượng trong training data (thường sử dụng khoảng cách Euclidean, Cosine…)
3. Sắp xếp khoảng cách theo thứ tự tăng dần và xác định k láng giềng gần nhất với Query Point
4. Lấy tất cả các lớp của k láng giềng gần nhất đã xác định
5. Dựa vào phần lớn lớp của láng giềng gần nhất để xác định lớp cho Query Point.

Ví dụ phân lớp hoa Iris

What's Iris?

Versicolor

Virginica

Setosa

Ví dụ phân lớp hoa Iris …

Chiều dài đài hoa	Chiều rộng đài hoa	Chiều dài cánh hoa	Chiều rộng cánh hoa	Tên loài hoa
5.1	3.5	1.4	0.2	Setosa
4.7	3.2	1.3	0.2	Setosa
7.0	3.2	4.7	1.4	Versicolor
6.3	3.3	6.0	2.5	Virnigica
5.9	3.0	4.2	1.5	Versicolor

Ví dụ phân lớp hoa Iris …

Chiều dài đài hoa	Chiều rộng đài hoa	Chiều dài cánh hoa	Chiều rộng cánh hoa	Tên loài hoa
5.1	3.5	1.4	0.2	Setosa
4.7	3.2	1.3	0.2	Setosa
7.0	3.2	4.7	1.4	Versicolor
6.3	3.3	6.0	2.5	Virnigica
5.9	3.0	4.2	1.5	Versicolor
5.1	**3.8**	**1.6**	**0.2**	**X**

Chúng ta cần xác định X thuộc loài hoa nào?

Ví dụ phân lớp hoa Iris …

Bởi vì có sự chênh lệch lớn của những giá trị thuộc tính, nên tránh sự phụ thuộc cục bộ.
Trước hết chúng ta cần chuẩn hóa dữ liệu trước khi tính toán.
Công thức chuẩn hóa cột v như sau:

$$a_i = \frac{v_i}{\max(v)}$$

Trong đó:
- v_i là giá trị thực tế của thuộc tính thứ i.
- a_i là giá trị sau khi chuẩn hóa của thuộc tính thứ i.
- max(v) là giá trị lớn nhất của cột v

Ví dụ phân lớp hoa Iris …

Bảng dữ liệu sau khi chuẩn hóa như sau

Chiều dài đài hoa	Chiều rộng đài hoa	Chiều dài cánh hoa	Chiều rộng cánh hoa	Tên loài hoa
0.73	0.92	0.23	0.08	Setosa
0.67	0.84	0.22	0.08	Setosa
1	0.84	0.78	0.56	Versicolor
0.9	0.87	1	1	Virnigica
0.84	0.79	0.7	0.6	Versicolor
0.73	**1**	**0.27**	**0.08**	**X**

Ví dụ phân lớp hoa Iris …

Tính khoảng cách Euclide:

X = (x1, x2,…, xn) và Y = (y1, y2,…, yn).

$$d(X, Y) = \sqrt{\sum_{i=1}^{n} (X_i - Y_i)^2}$$

Mục đích cuối của ta là so sánh những khoảng cách này, do đó trong tính toán ta bỏ qua quá trình lấy căn bậc 2.

Ví dụ:

X = (0.73, 1, 0.27, 0.08).

Y = (0.73, 0.92, 0.23, 0.08).

$D(X,Y)= (0.73 - 0.73)^2 + (1 - 0.92)^2 + (0.27 - 0.23)^2 + (0.08 - 0.08)^2 = 0.008$

Ví dụ phân lớp hoa Iris …

Khoảng cách của X so với các item khác như sau:

Tên loài hoa	Khoảng cách
Setosa	0.008
Setosa	0.178
Versicolor	0.767
Virginica	1.194
Versicolor	0.715

Ví dụ phân lớp hoa Iris …

Sắp xếp khoảng cách theo thứ tự tăng dần:

Tên loài hoa	Khoảng cách
Setosa	0.008
Setosa	0.178
Versicolor	0.715
Versicolor	0.767
Virginica	1.194

Ví dụ phân lớp hoa Iris …

Phân lớp hoa iris với k = 3:

Tên loài hoa	Khoảng cách
Setosa	0.008
Setosa	0.178
Versicolor	0.715

Trong 3 hàng xóm thì ta thấy có 2 lớp Setosa, 1 lớp Versicolor. Nên ta kết luận loài hoa cần phân lớp thuộc vào lớp Setosa.

Ví dụ phân lớp hoa Iris …

Phân lớp hoa iris với k = 4:

Tên loài hoa	Khoảng cách
Setosa	0.008
Setosa	0.178
Versicolor	0.715
Versicolor	0.767

Trong 4 hàng xóm thì ta thấy có 2 hàng xóm là Setosa, 2 hàng xóm Versicolor. Ta thấy rằng số hàng xóm bằng nhau. Ta tiến hành tính trung bình khoảng cách của 2 lớp này:

Ví dụ phân lớp hoa Iris …

Trung bình khoảng cách của 2 lớp:

Tên loài hoa	Khoảng cách	Trung bình khoảng cách
Setosa	0.008	(0.008 + 0.178)/2 = 0.0973
Setosa	0.178	
Versicolor	0.715	(0.715 + 0.767)/2 = 0.741
Versicolor	0.767	

Ta chọn khoảng cách nhỏ nhất là 0.0973, tương ứng với loài hoa Setosa. Nên kết luận loài hoa cần phân lớp thuộc hoa Setosa.

Những vấn đề của thuật toán

1. Chọn giá trị k phù hợp?

2. Làm thế nào ít lưu trữ dữ liệu nhưng vẫn đảm bảo tính chính xác? Được S. Cost và S. Salzberg đưa ra giải pháp là đặt trọng số cho những tính chất đặc trưng vào năm 1993.

3. Ứng dụng trong việc sắp xếp đồ thị của những công việc gần nhau. Được G. T. Toussaint đưa ra vào năm 2002.

4. Ứng dụng logic mờ cho kNN để lấy những yêu cầu chăm sóc sức khỏe của bệnh nhân. Được J. C. Bezdek, S. K. Chuah, and D. Leep đưa ra vào năm 1986.

5. Ứng dụng trong nhận dạng biển số xe (nhận dạng ký tự của biển số xe)

3.5. Phân lớp với SVM (Support Vector Machine)

SVM

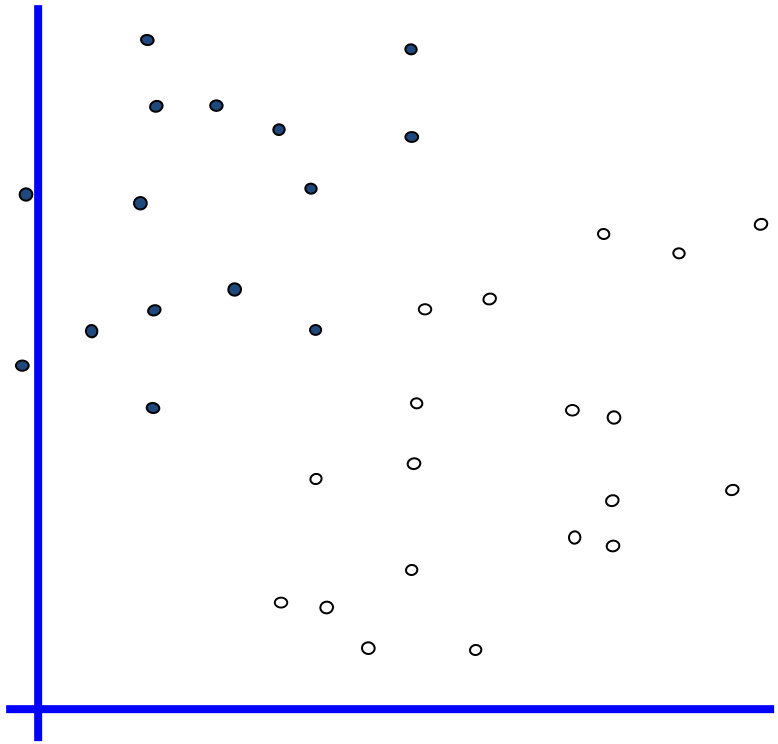
- **SVM** là một phương pháp máy học có giám sát giải quyết được các bài toán phân lớp. Do Vladimir N. vapnik đề xuất năm 1995

- **Ý tưởng** : Theo toán học, ánh xạ một vector mới vào *không gian các vector đặc trưng* (space of feature vectors) mà ở đó một *siêu phẳng tối ưu* được tìm ra để tách dữ liệu thuộc hai lớp khác nhau.

SVM tìm *siêu phẳng tối ưu*.

denotes +1

denotes -1



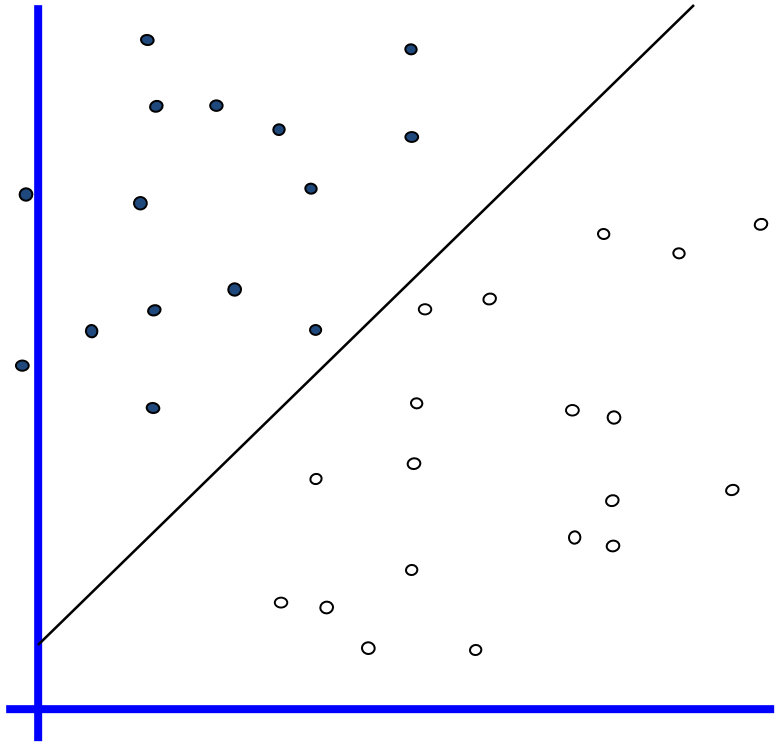
How would you classify this data?

SVM tìm *siêu phẳng tối ưu.*

$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

denotes +1

denotes -1



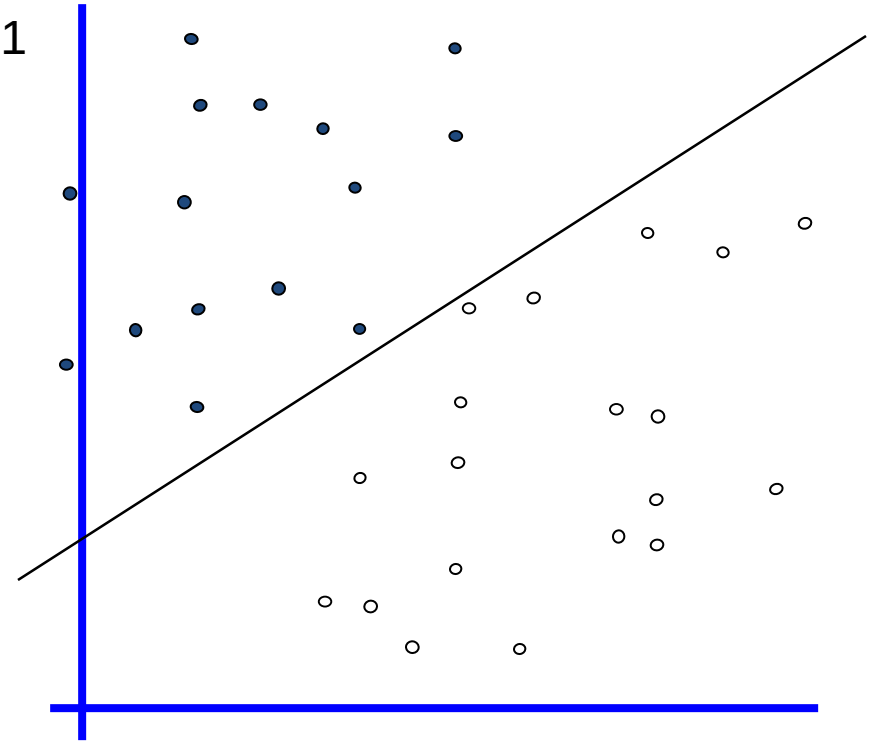
How would you classify this data?

SVM tìm *siêu phẳng tối ưu.*

$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

denotes +1

denotes -1

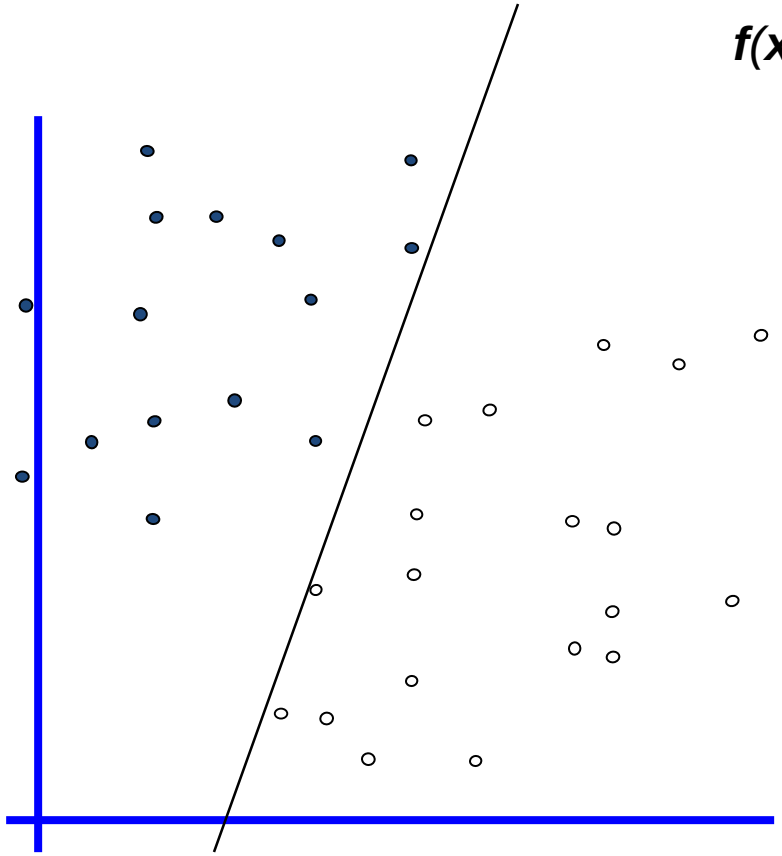


How would you classify this data?

SVM tìm *siêu phẳng tối ưu.*

denotes +1

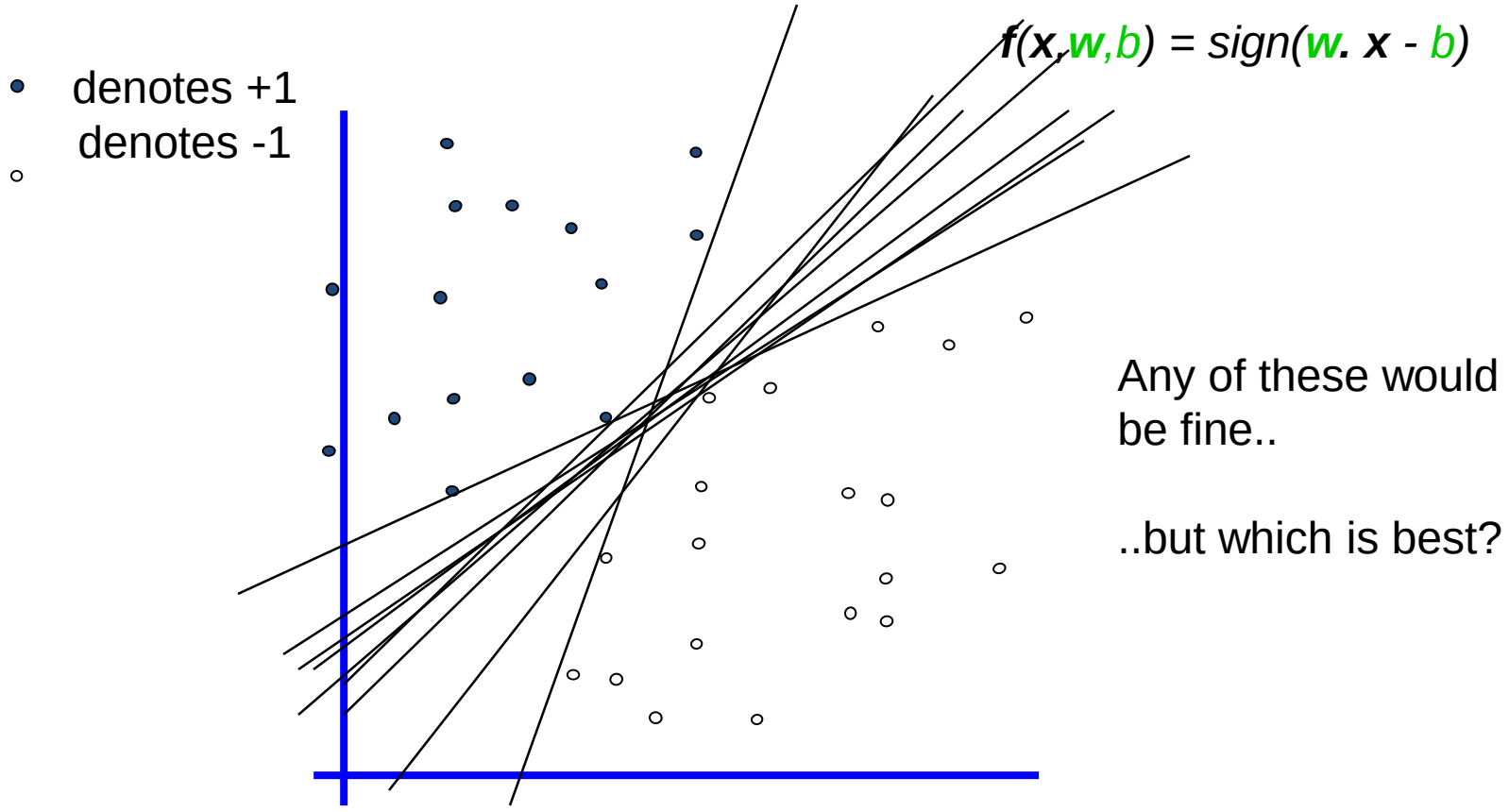
denotes -1



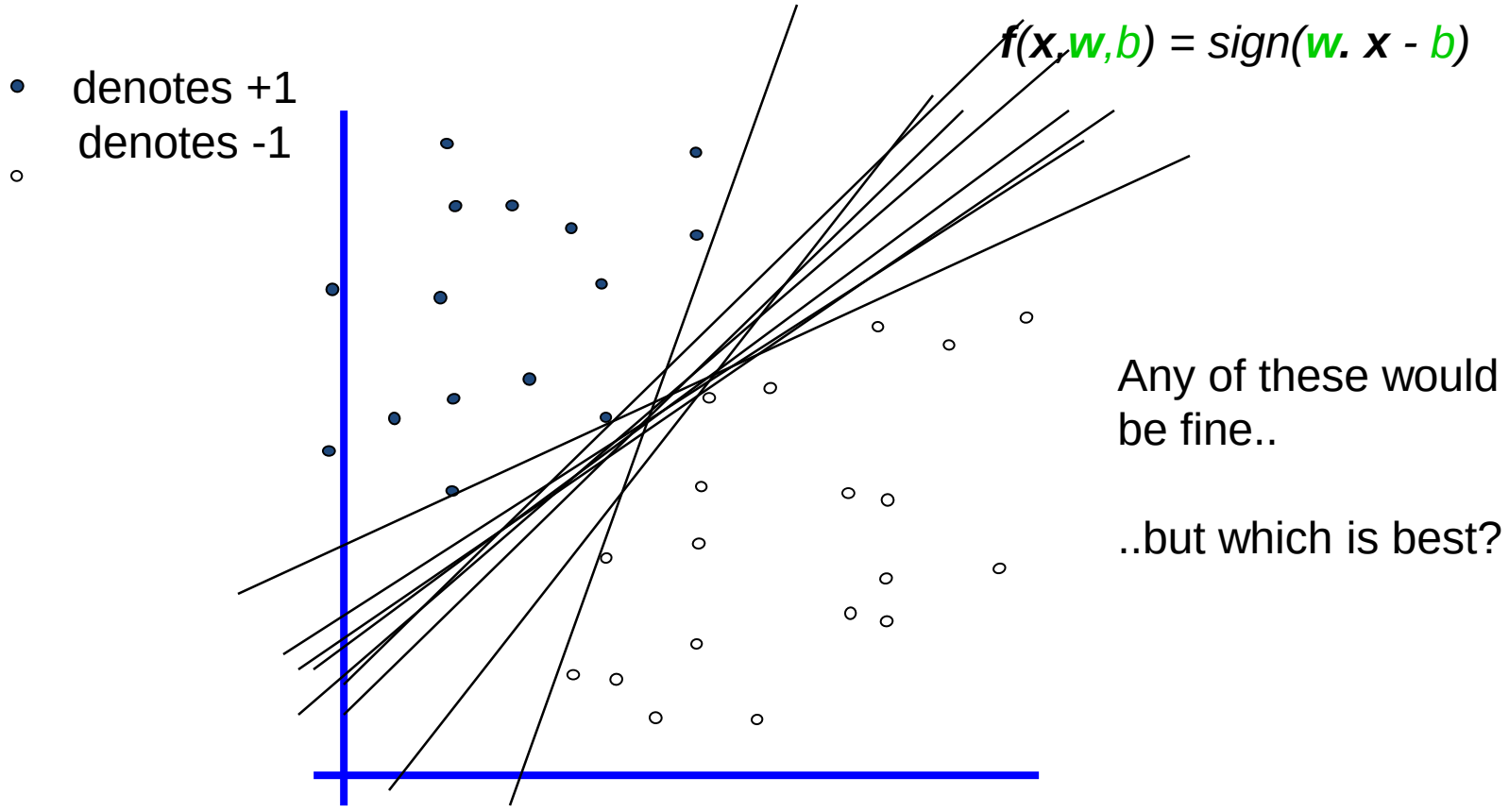
$f(\mathbf{x},\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

How would you classify this data?

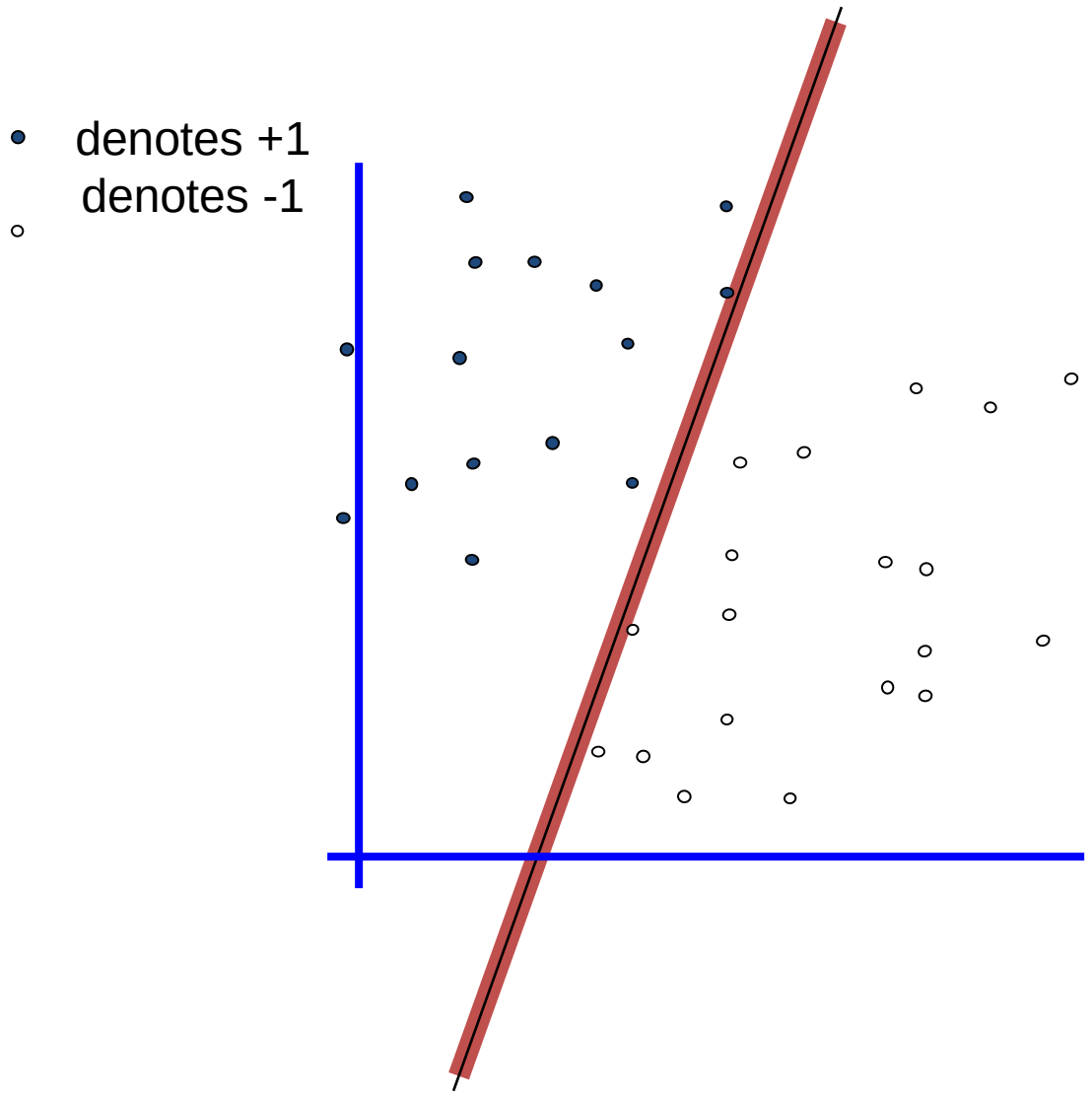
SVM tìm *siêu phẳng tối ưu.*



SVM tìm *siêu phẳng tối ưu.*

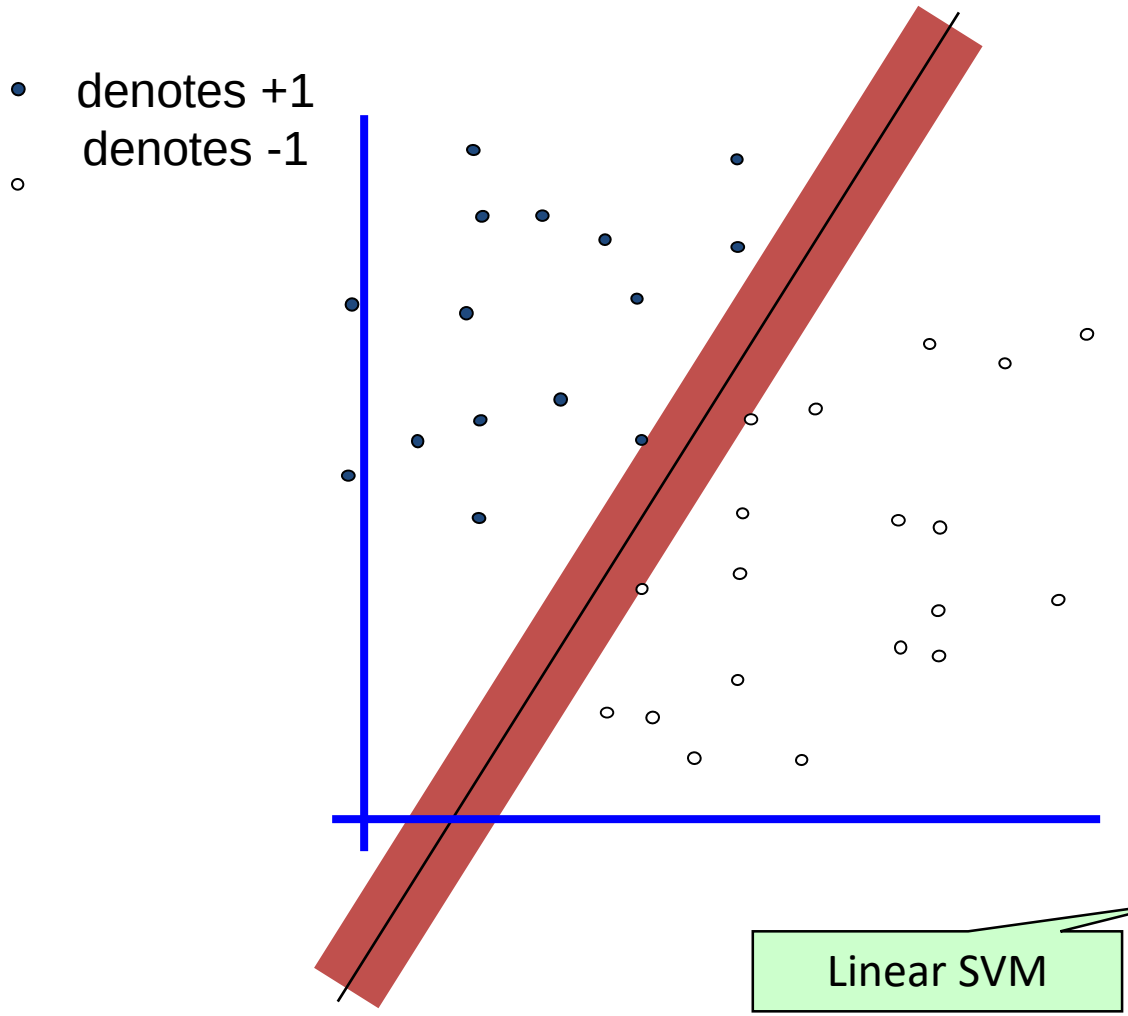


SVM tìm *siêu phẳng tối ưu.*



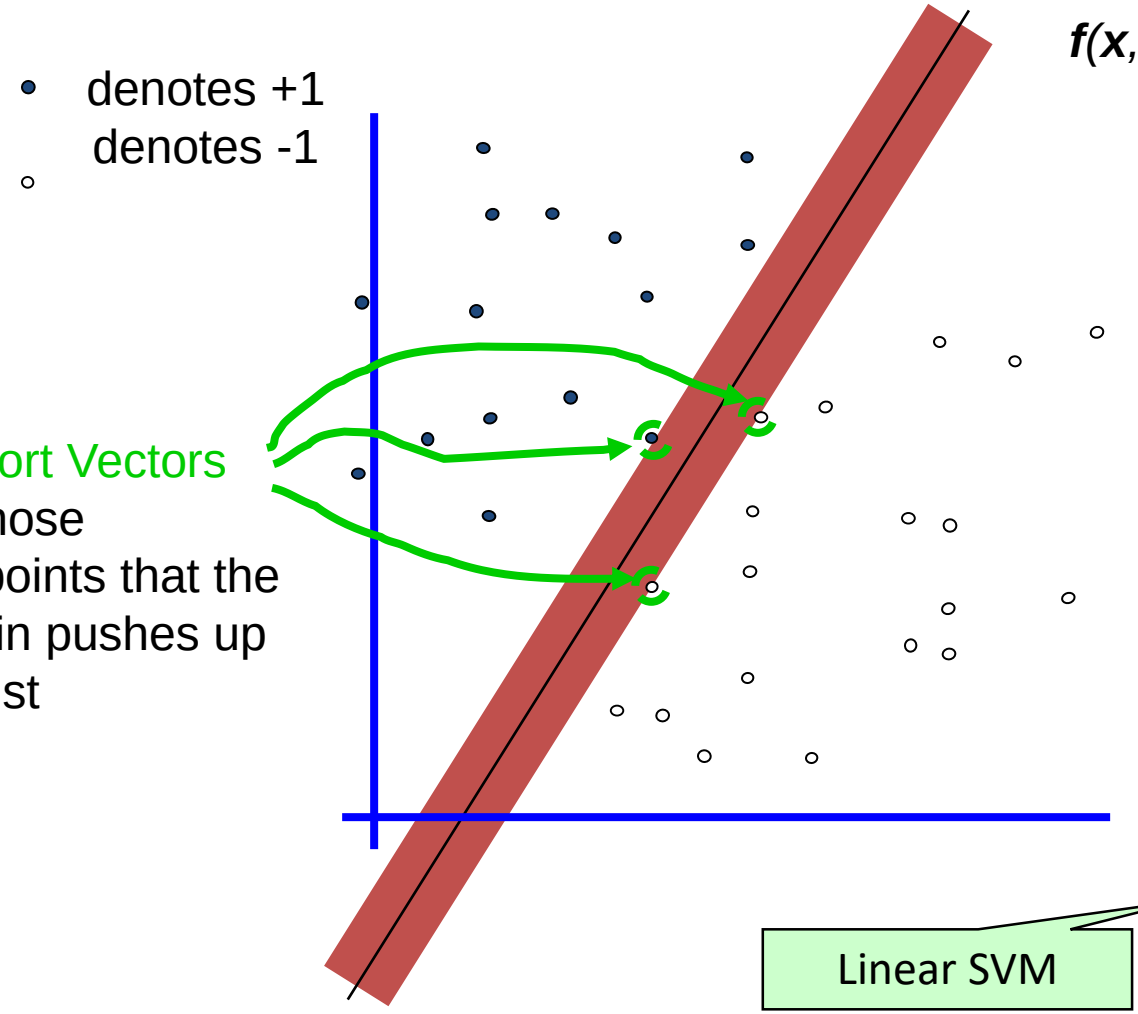
Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

SVM tìm *siêu phẳng tối ưu.*



The maximum margin linear classifier is the linear classifier with the, um, maximum margin. This is the simplest kind of SVM (Called an LSVM)

SVM tìm *siêu phẳng tối ưu.*

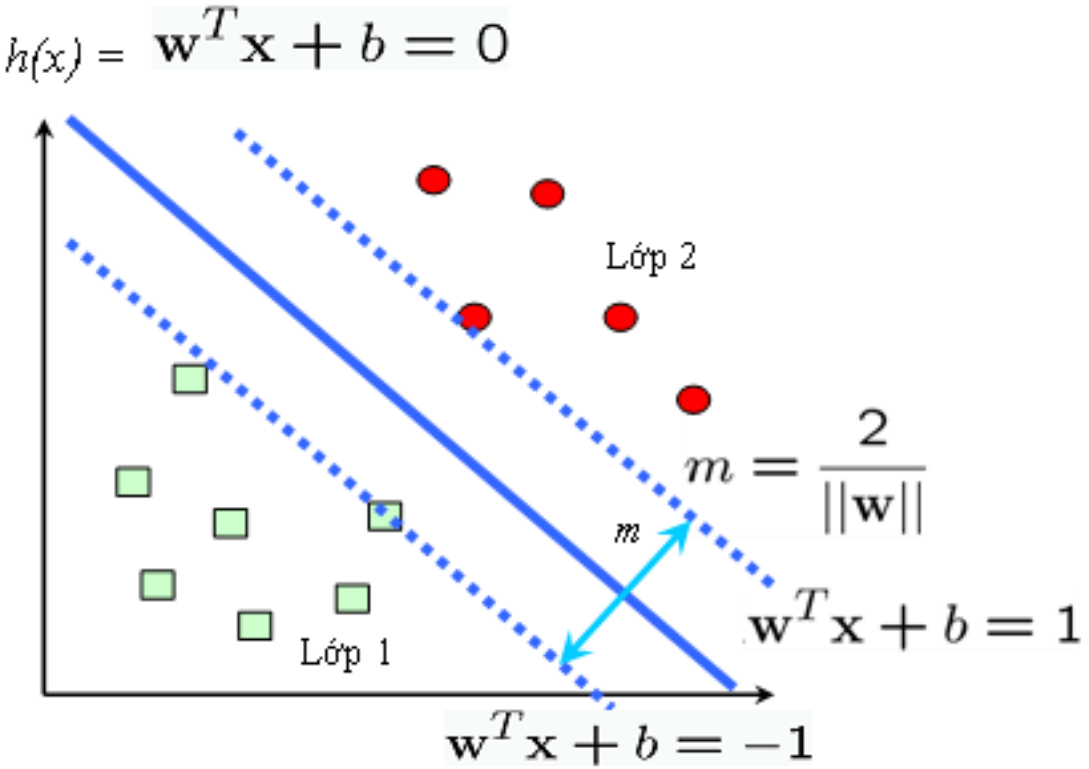


$f(\mathbf{x}, \mathbf{w}, b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

The maximum margin linear classifier is the linear classifier with the, um, maximum margin. This is the simplest kind of SVM (Called an LSVM)

SVM tìm *siêu phẳng tối ưu.*

- Từ đó bài toán đặt ra là tìm siêu phẳng tách . Đây cũng là bài toán chính của SVM.
- *Cho tập dữ liệu*



- Từ đó bài toán đặt ra là tìm siêu phẳng tách $w \bullet x + b = 0$. Đây cũng là bài toán chính của SVM.
- *Cho tập dữ liệu :*

$$Tr = \{(x_1, y_1), ..., (x_l, y_l)\}, \qquad x_i \in R^n, y_i \in \{-1, 1\}$$

- Tập dữ liệu *Tr* có thể phân chia tuyến tính được mà không có nhiễu. Việc của SVM là làm sao chúng ta chọn được các tham số *w* và *b* sao cho dữ liệu huấn luyện có thể diễn giãi được như sau :
  - $x_i \cdot w + b \geq +1$ *với* $y_i = +1$
  - $x_i \cdot w + b \leq -1$ *với* $y_i = -1$.
- Kết hợp hai biểu thức trên :
  - $y_i(x_i \cdot w + b) - 1 \geq 0$, *với mọi i.*
- $Min|\, y_i(x_i \cdot w + b)| = 1$.

- Vấn đề đặt ra bây giờ là xác định các hệ số *w* và *b* như thế nào để siêu phẳng tìm được là tốt nhất?
- Siêu phẳng tốt nhất là siêu phẳng mà có khoảng cách từ điểm dữ liệu huấn luyện gần nhất đến siêu phẳng là xa nhất. Mà khoảng cách từ một điểm dữ liệu x_i đến siêu phẳng là; (en.wikipedia.com)

$$d(w,b;x_i) = \frac{\left|w^T.x_i + b\right|}{\|w\|}$$

h(w,b) là tổng của khoảng cách từ điểm dữ liệu gần nhất của lớp 1 đến siêu phẳng và khoảng cách từ điểm dữ liệu gần nhất của lớp –1 đến siêu phẳng. Ta có:

$$
\begin{aligned}
h(w,b) &= \min_{x_i, y_i=-1} d(w,b;x_i) + \min_{x_i, y_i=1} d(w,b;x_i) \\
&= \min_{x_i, y_i=-1} \frac{\left|w^T.x_i + b\right|}{\|w\|} + \min_{x_i, y_i=1} \frac{\left|w^T.x_i + b\right|}{\|w\|} \\
&= \frac{1}{\|w\|}\left(\min_{x_i, y_i=-1} \left|w^T.x_i + b\right| + \min_{x_i, y_i=1} \left|w^T.x_i + b\right| \right) \\
&= \frac{2}{\|w\|}
\end{aligned}
$$

- Như vậy, siêu phẳng tối ưu là siêu phẳng có

$$h(w,b) = 2/\|w\|$$

lớn nhất, tương đương với $\|w\|$ là nhỏ nhất.

SVM tìm *siêu phẳng tối ưu.*

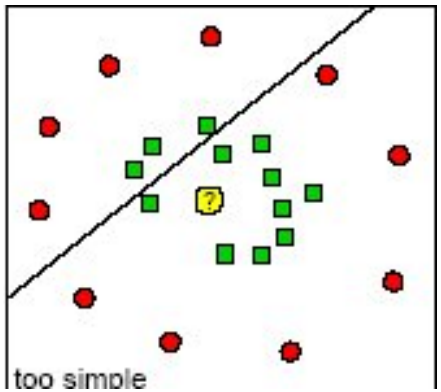
- Tóm lại, việc tìm siêu phẳng tốt nhất tương đương với việc giải bài toán tối ưu sau:

$$\begin{cases} \underset{w}{Min}\ \Phi(w) = \frac{1}{2}\|w\|^2 \\ y_i(w^T . x_i + b) \geq 1, \qquad i = 1, \ldots, l \end{cases}$$

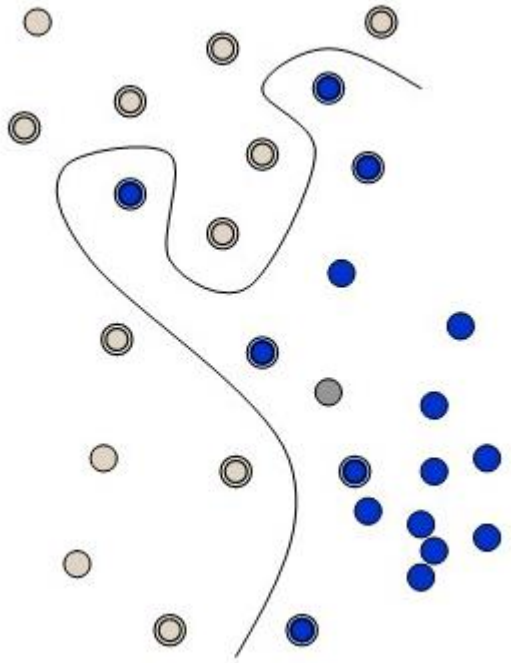
Lagrange multipliers : ***w, b ?***

- Lagrange?.
- Còn rất nhiều trường hợp khác?.

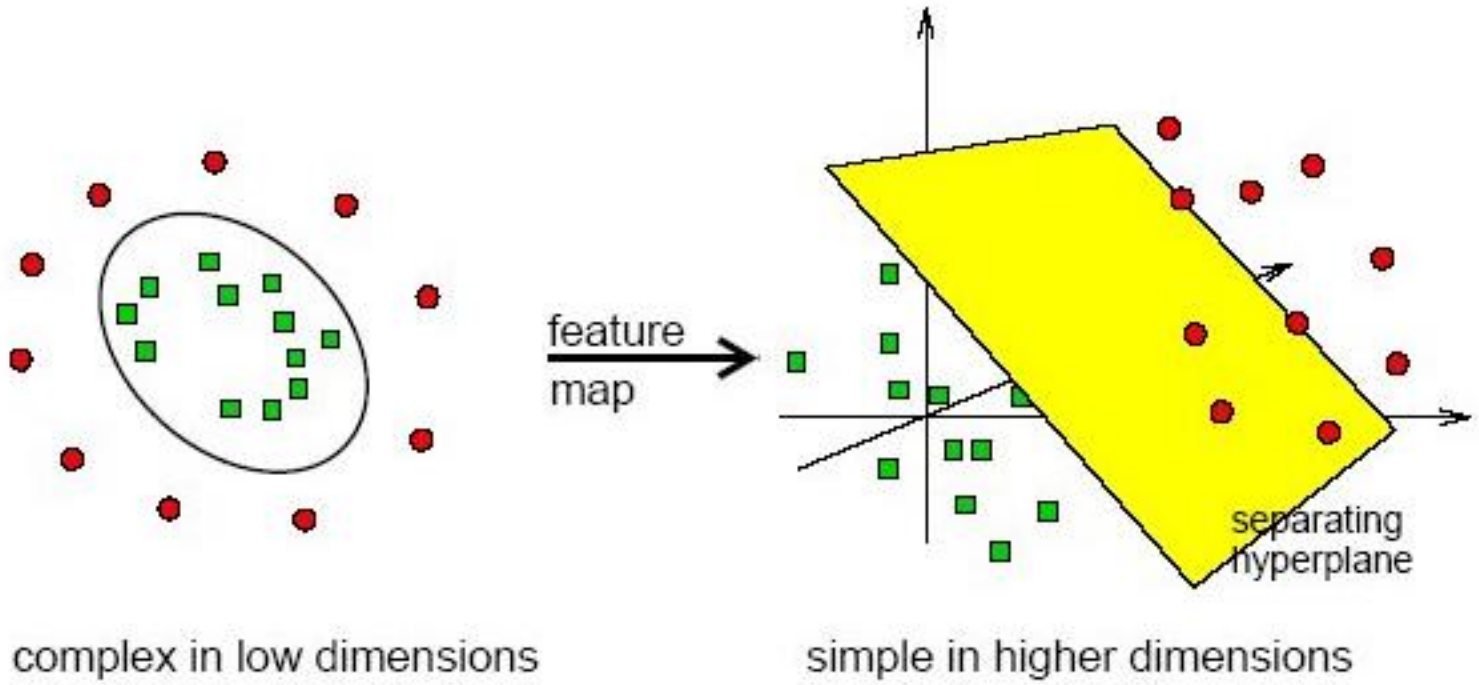
SVM tìm *siêu phẳng tối ưu*.



?



Kernel


feature
map
separating
hyperplane
complex in low dimensions
simple in higher dimensions

Ứng dụng của SVM: Sử dụng trong phân loại và nhận dạng

- Trong việc nhận dạng chữ viết tay tiếng Việt

+Dựa trên cơ sở các thành phần liên thông của ảnh, phân tập ký tự tiếng Việt thành 3 nhóm (box, dạng chữ rời, dạng chữ liên tục) và tách các ký tự có dấu thành các phần rời nhau.

+Sau đó xây dựng máy phân lớp SVM để nhận dạng cho từng phần chữ và phần dấu.

+ Kết quả thực nghiệm cho thấy mô hình nhận dạng có độ chính xác tương đối cao.

- **Trong nhận dạng mặt người :**

+Giai đoạn huấn luyện: các ảnh mẫu được vector hóa x= { $X_1,\ldots,X_{900}$ } rồi dùng phương pháp PCA để rút trích đặc trưng thành vector y ={ $Y_1,\ldots,Y_{100}$ } rồi đưa vào bộ huấn luyện SVM

+Giai đoạn nhận dạng: Mẫu cần nhận dạng được vector hóa và rút trích như trên sau đó đưa vào bộ nhận dạng SVM để xác định lớp cho mẫu.

Finish of chapter 3