



Bài giảng môn học:

**Thị giác máy tính (7080518)**

# CHƯƠNG 5: Theo vết, nhận dạng đối tượng

Đặng Văn Nam

[dangvannam@hmg.edu.vn](mailto:dangvannam@hmg.edu.vn)

# Nội dung chương 5



- 1. Mô tả bài toán theo vết đối tượng (Object Tracking)**
- 2. Một số phương pháp Object Tracking**
- 3. Ví dụ: Single Object Tracking với OpenCV và Python**
- 4. Bài toán phát hiện, nhận dạng đối tượng (Object Detection)**
- 5. Ví dụ: Nhận diện hoa dựa trên Học sâu (Deep learning)**

# 1. Mô tả bài toán theo vết đối tượng (Object tracking)

# Theo vết đối tượng trong Video

Trong thị giác máy tính có 2 bài toán phổ biến:

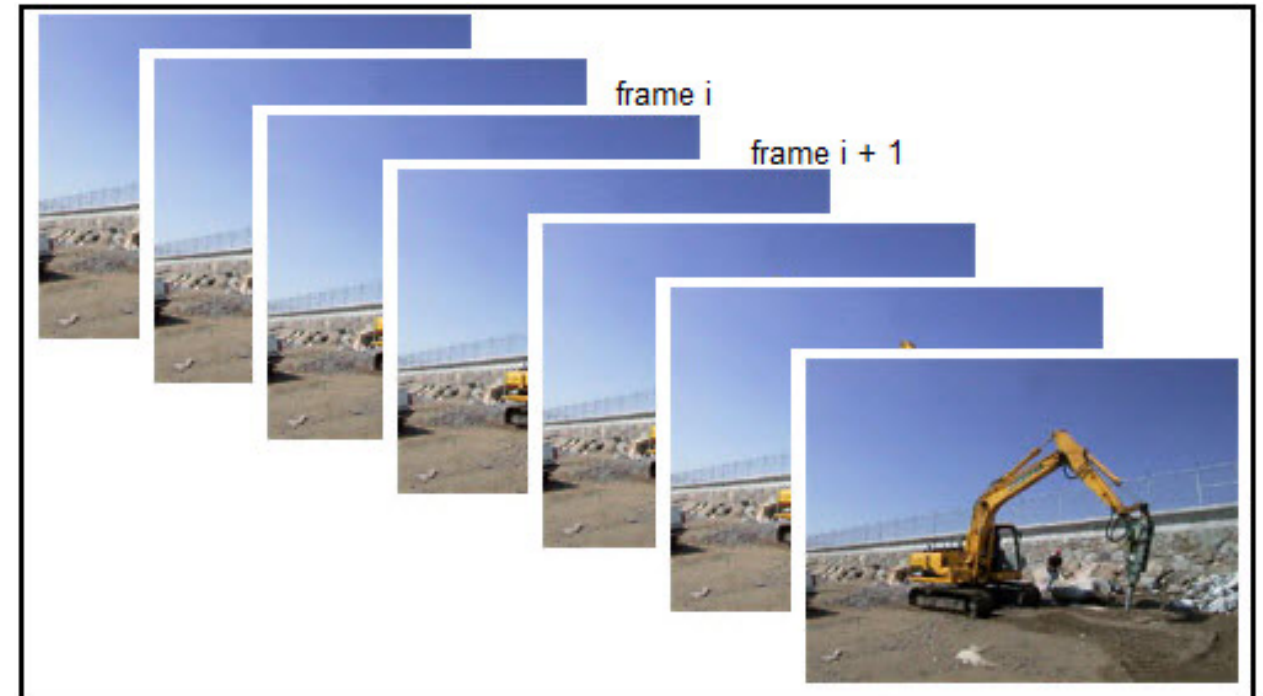
- **Phát hiện đối tượng - Object detection (OD)** – Làm việc trên một khung hình (frame)
- **Theo vết đối tượng - Object tracking (OT)** – Làm việc trên nhiều khung hình liên tiếp





# Theo vết đối tượng trong Video

- Video bản chất là một tập hợp các khung hình frame liên tiếp nhau. Các frame có mối liên hệ mật thiết với nhau theo thời gian.
- Mỗi một frame tại thời điểm nhất định sẽ có nhiều khả năng giống với các frame đứng ngay phía trước và ngay phía sau frame hiện tại.

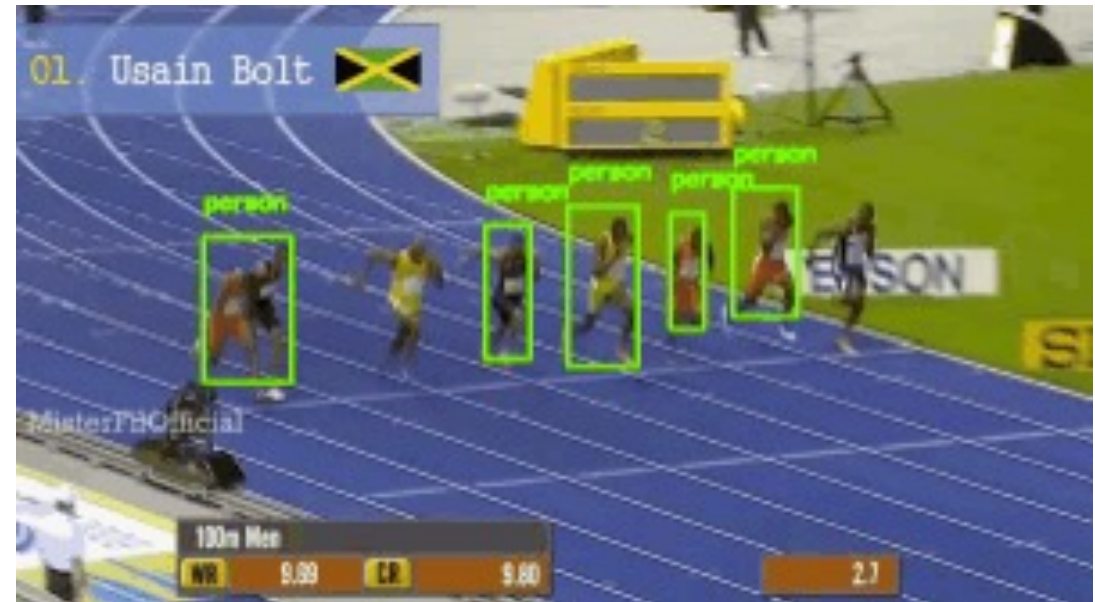


# Theo vết đối tượng trong Video (Video Object Tracking)



Mục tiêu của Video Object tracking là xác định vị trí của một hoặc nhiều đối tượng (target) trong các frame liên tiếp của Video

- **Single Object tracking (SOT)** – Chỉ có một object được theo dõi ngay cả khi môi trường có nhiều đối tượng.
- **Multiple Object tracking (OT)** – Nhiều đối tượng được theo dõi theo thời gian, thậm chí nó có thể theo dõi các đối tượng mới xuất hiện trong video.



- Trang bị kiến thức cơ bản về Object Tracking.
- Giới thiệu Các phương pháp Object tracking.
- Thực hành: Sử dụng OpenCV thực hiện theo vết đối tượng trong video (single object tracking) với một số thuật toán khác nhau

## **VIDEO OBJECT TRACKING** **- FIT.HUMG -**

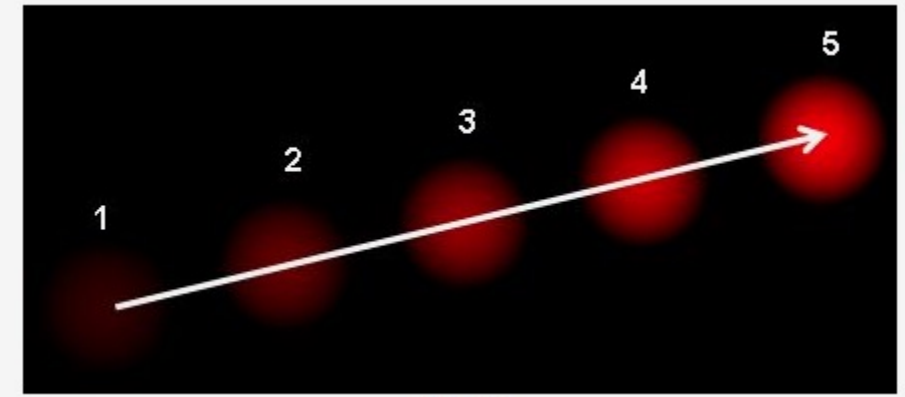
## 2. Một số phương pháp Object tracking



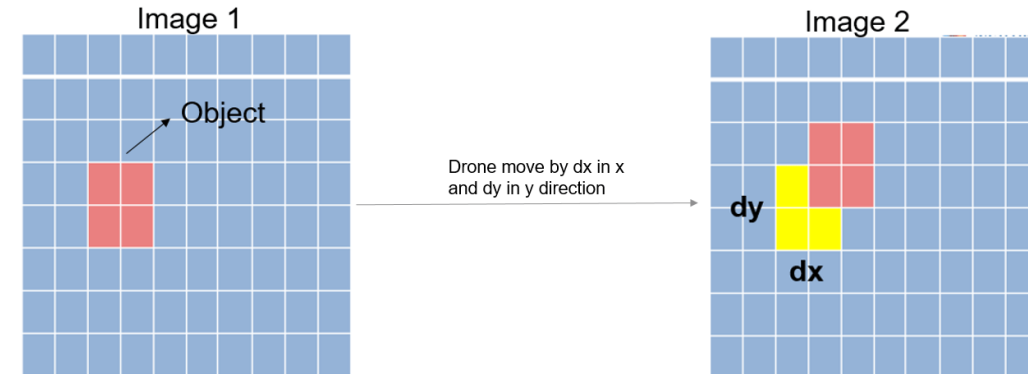
## 2.1 Optical Flow

# 1. Luồng quang học – Optical flow

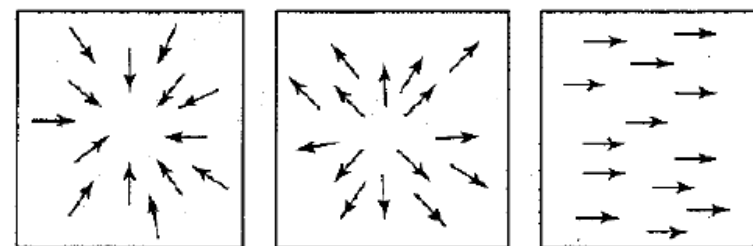
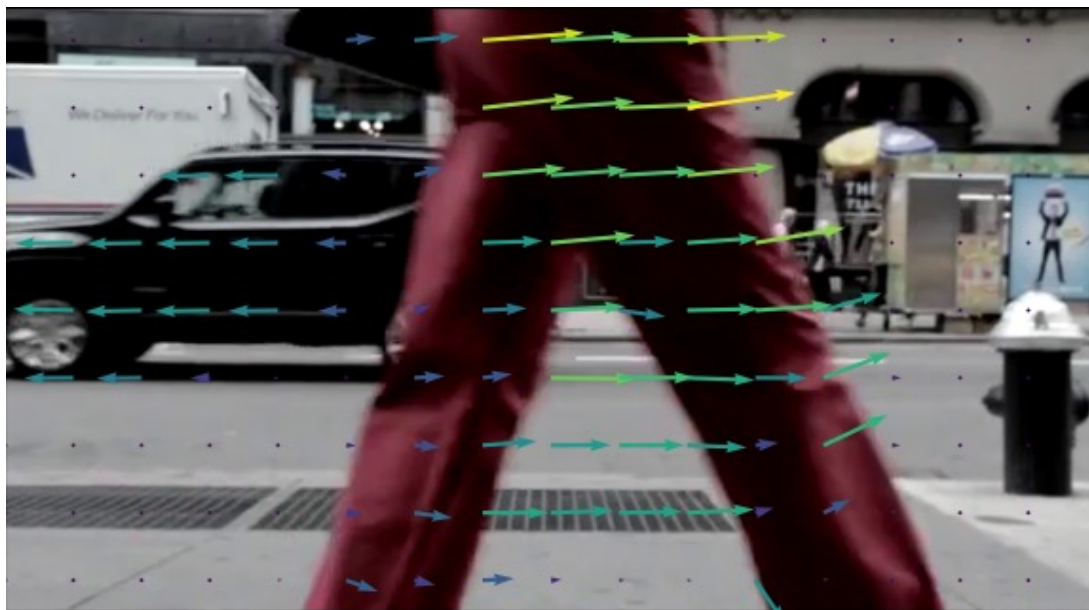
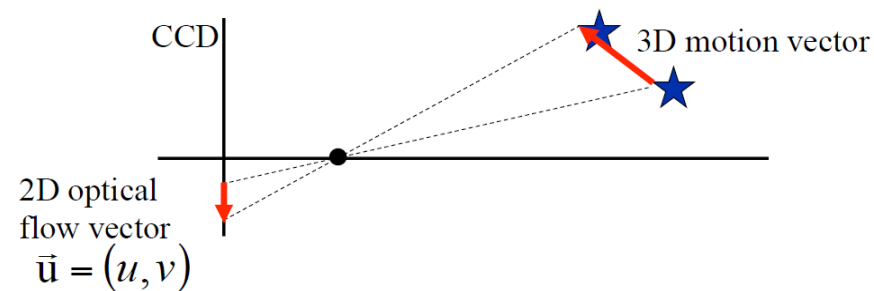
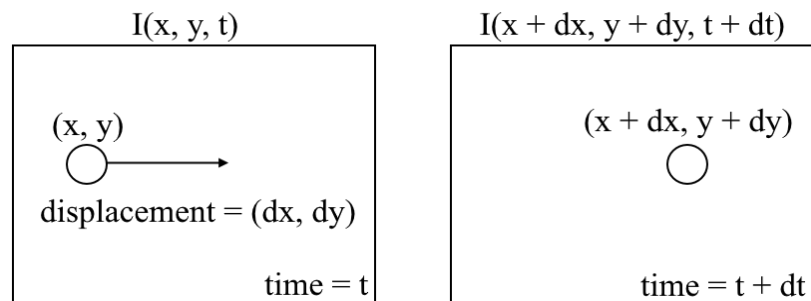
- Luồng quang học (Optical flow) là kỹ thuật ước lượng sự chuyển động ở mỗi điểm ảnh độc lập, có thể cho biết thông tin từng phần khung cảnh thay đổi ra sao theo không gian - thời gian dựa vào sự thay đổi độ sáng của các pixel điểm ảnh.
- Tập trung vào việc thu được vector dịch chuyển của đối tượng qua các frame.
- **Brightness consistency:** Độ sáng xung quanh một vùng nhỏ được cho là gần như không đổi, mặc dù vị trí của vùng có thể thay đổi.
- **Spatial coherence:** các điểm lân cận trong cùng một khung cảnh thường sẽ cùng thuộc một bề mặt do đó sẽ có những chuyển động tương tự.
- **Temporal persistence:** các điểm thường sẽ có sự chuyển động dần dần.



image



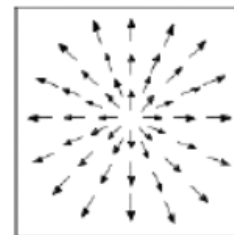
# 1. Luồng quang học – Optical flow



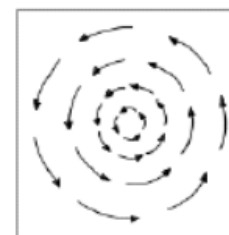
Zoom out

Zoom In

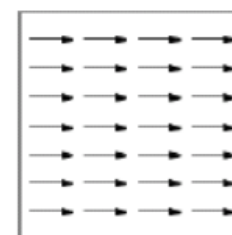
Pan right to left



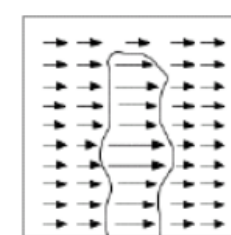
Forward motion



Rotation

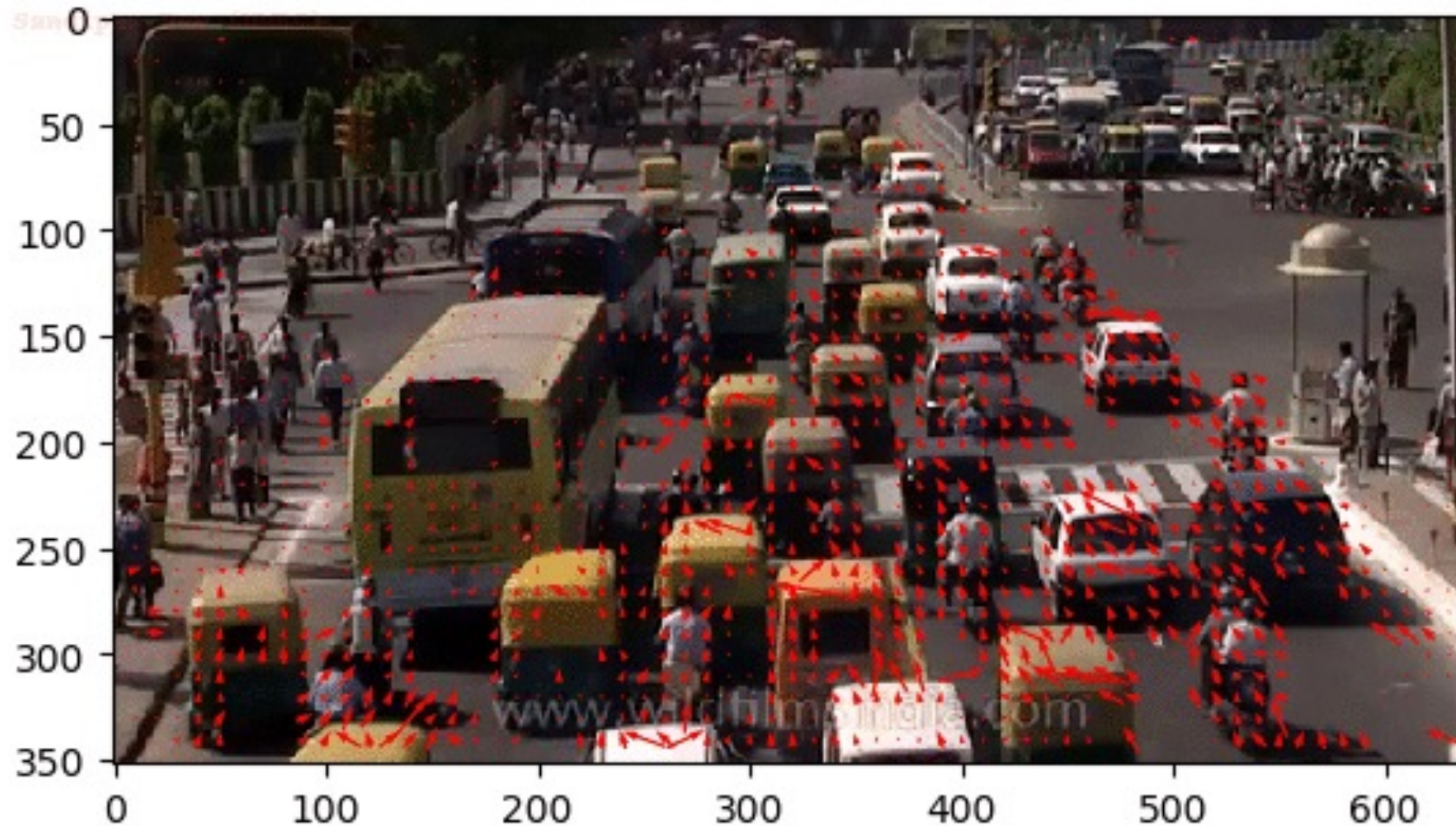


Horizontal translation



Closer objects appear to move faster!!

# 1. Luồng quang học – Optical flow



Chi tiết: <https://nanonets.com/blog/optical-flow/>



# 1. Luồng quang học – Optical flow



Left: Sparse Optical Flow - track a few "feature" pixels; Right: Dense Optical Flow - estimate the flow of all pixels in the image

Chi tiết: <https://nanonets.com/blog/optical-flow/>

➤ Example Optical flow: file **Optical\_Flow.ipynb**

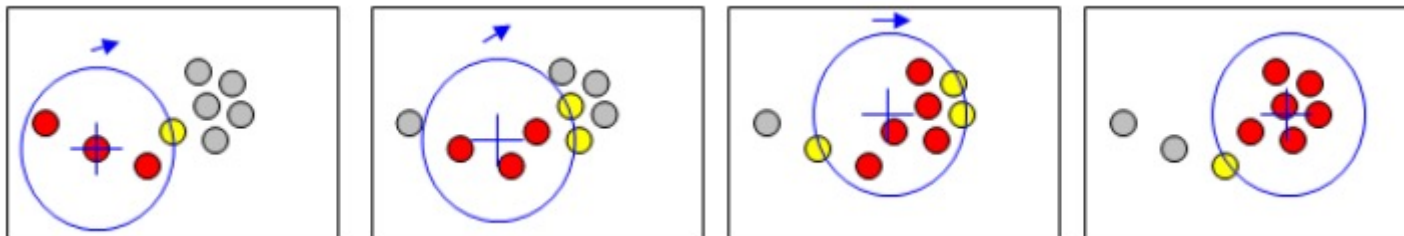
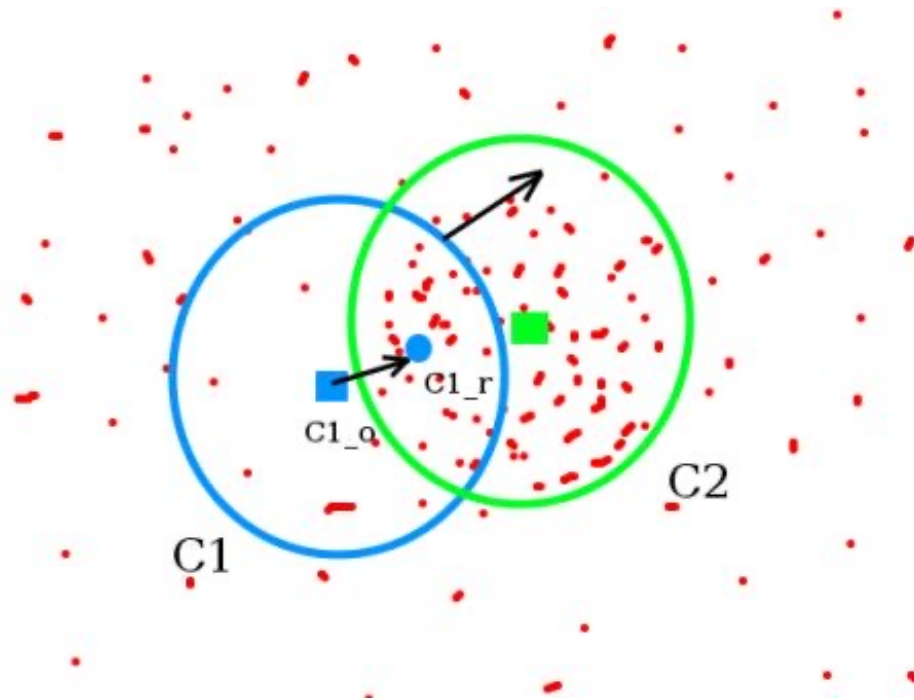


## 2.2 MeanShift - CamShift

# a. MeanShift

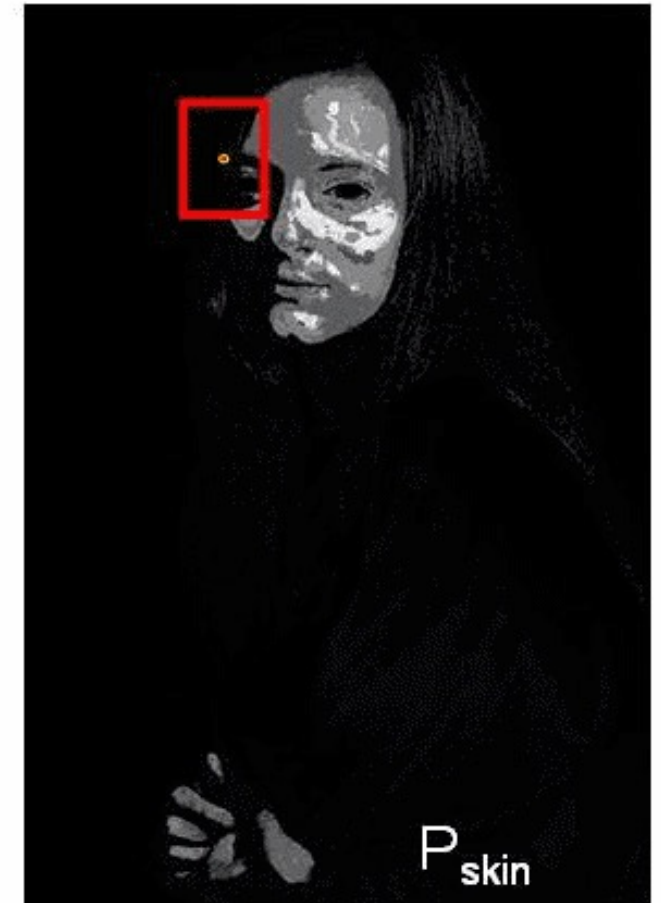
- MeanShift là một thuật toán dịch chuyển đệ quy một điểm dữ liệu đến trung bình của các điểm dữ liệu tại vùng lân cận của nó, tương tự như việc gom các điểm dữ liệu tạo thành một nhóm.
- Kích thước cửa sổ tìm kiếm là cố định qua các khung hình

➤ Example MeanShift:  
file **MeanShift\_CamShift.ipynb**



## b. CamShift

- Thuật toán CamShift (Continuously Adaptive MeanShift) được xuất phát và dựa trên nền tảng của thuật toán MeanShift.
- Ý tưởng chính của CamShift là tính lại trọng tâm của đối tượng dựa vào frame ảnh trước đó và kích thước của cửa sổ tìm kiếm được thay đổi động theo đối tượng



Mean shift window  
initialization

➤ Example MeanShift:  
file **MeanShift\_CamShift.ipynb**

### 3. Ví dụ

<https://www.youtube.com/watch?v=u8hCZUMDAAI&t=184s>

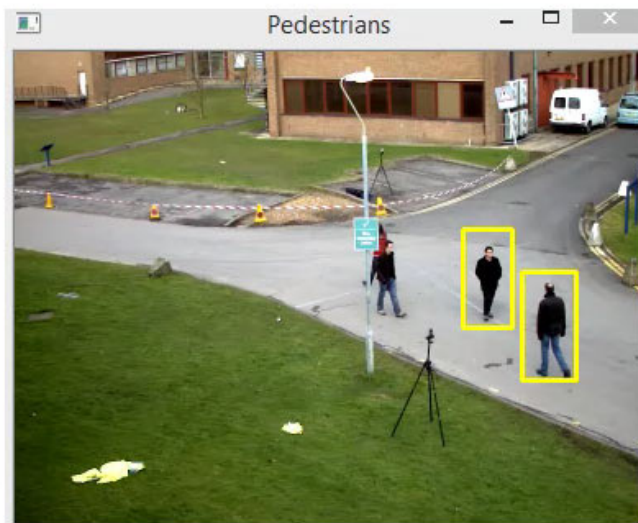
# Ví dụ







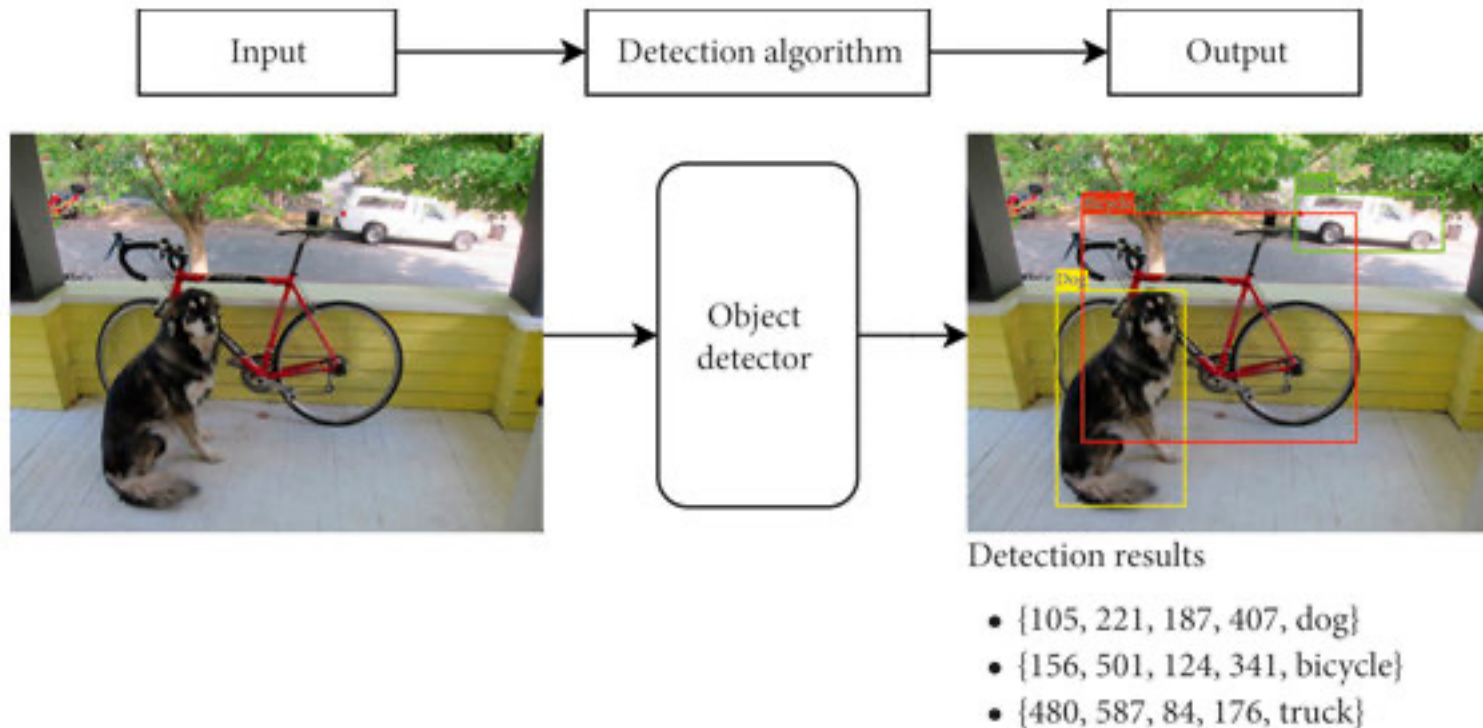
- BOOSTRING Tracker
- MIL Tracker
- KCF Tracker
- TLD Tracker
- MEDIANFLOW Tracker
- GOTURN Tracker
- MOSSER Tracker



## 4. Phát hiện, Nhận dạng đối tượng

# Giới thiệu bài toán nhận dạng đối tượng

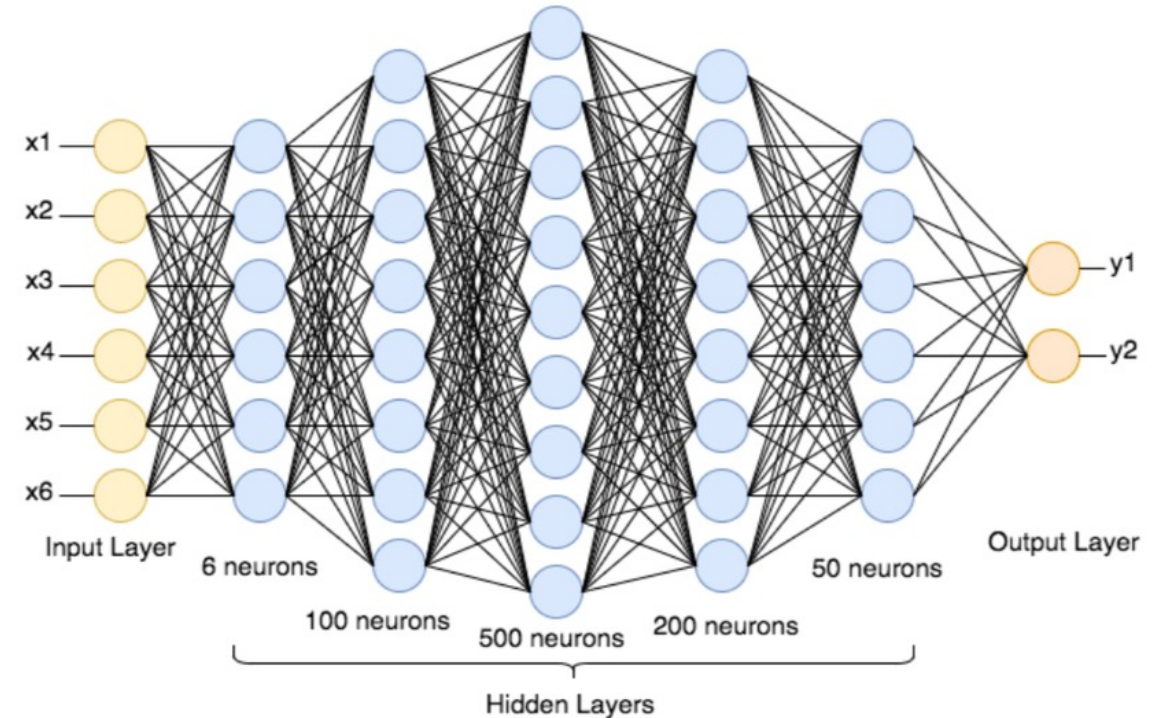
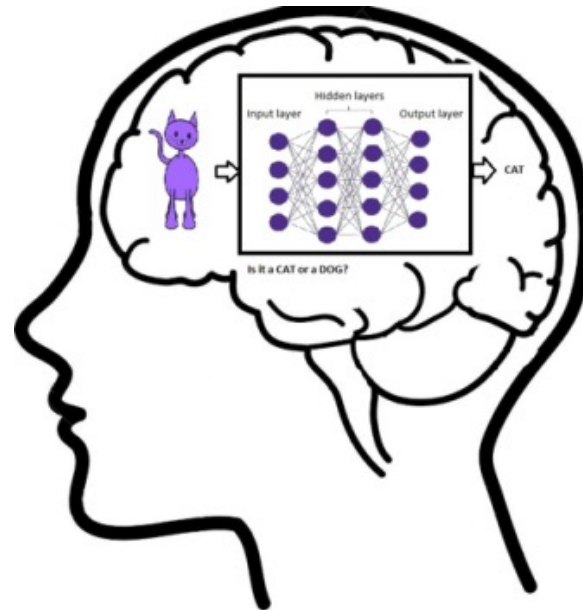
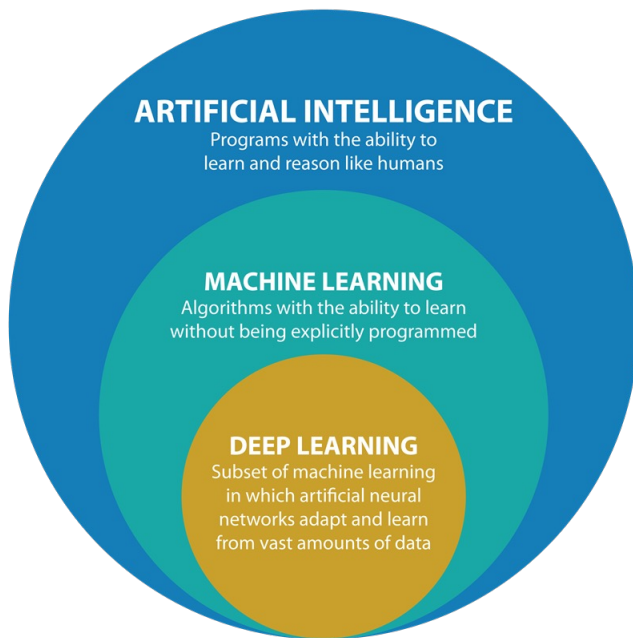
- Phát hiện đối tượng là một trong những bài toán quan trọng của thị giác máy tính với các ứng dụng trải rộng trong nhiều lĩnh vực khác nhau
- Object Detection đề cập đến khả năng của hệ thống máy tính và phần mềm để định vị các đối tượng trong một hình ảnh và xác định từng đối tượng.



Trong những năm gần đây, các phương pháp phát hiện đối tượng đã phát triển mạnh mẽ, đặc biệt đạt được bước cải tiến lớn về cả độ chính xác và tốc độ xử lý, điển hình như các phương pháp dựa trên mạng Học sâu (Deep learning)

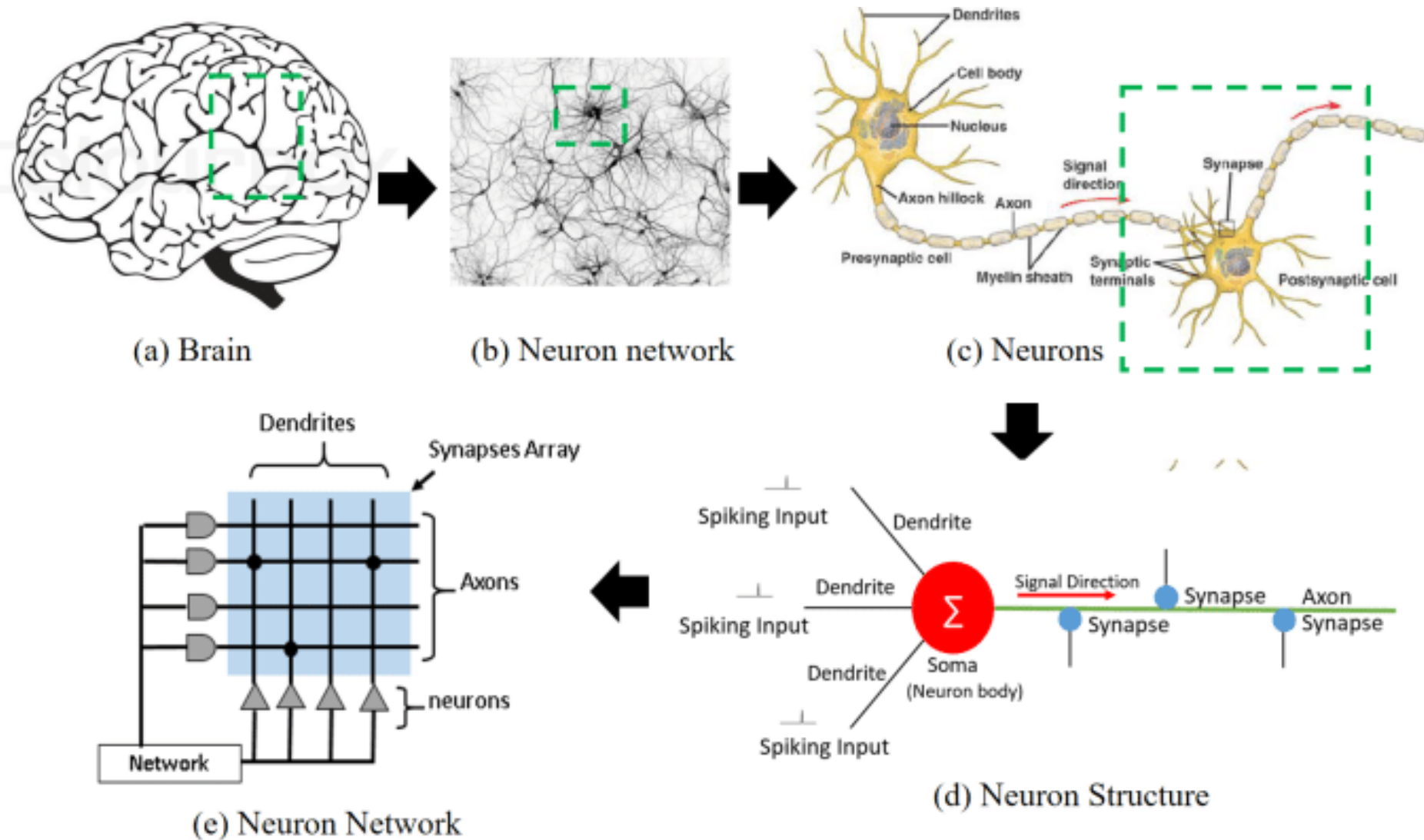
# Học sâu là gì?

- Học sâu là một tập con của Học máy (là một dạng đặc biệt của học máy); Học sâu lấy cảm hứng từ cách mà bộ não con người hoạt động, sử dụng mạng thần kinh nhân tạo nhiều lớp ẩn (Hidden layers) để tìm hiểu và khám phá thông tin từ dữ liệu.





# Học sâu là gì?





## 5. Ví dụ: Nhận diện hoa dựa trên học sâu

# Ví dụ: Nhận dạng Hoa dựa trên Học sâu (Deep learning)



Hoa cúc  
(daisy-650)



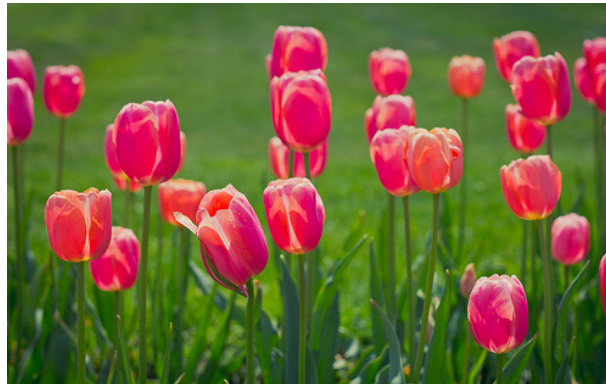
Hoa bồ công anh  
(dandelion-900)



Hoa hồng  
(roses-650)



Hoa hướng dương  
(sunflowers-700)

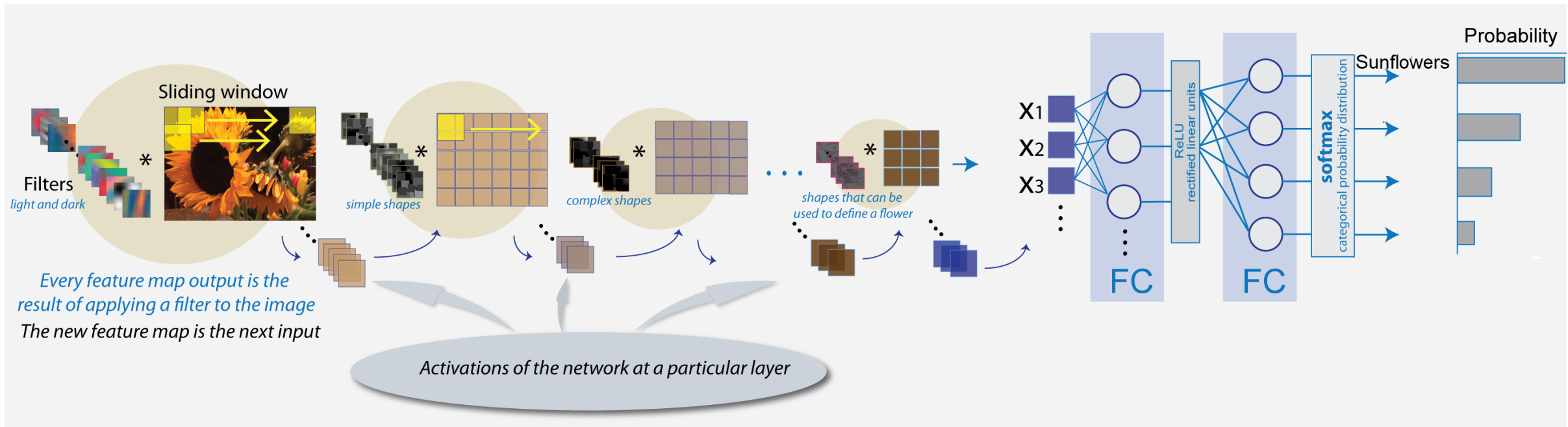
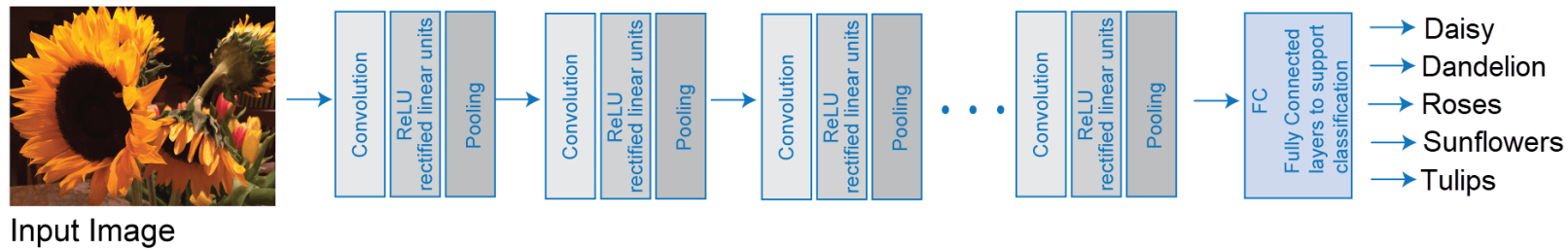


Hoa Tulip  
(tulips-800)

Tập dữ liệu bao gồm 3700 ảnh  
của 5 loại hoa:

# Ví dụ: Nhận dạng Hoa dựa trên Học sâu (Deep learning)

**Nhiệm vụ:** Xây dựng model học sâu sử dụng mạng CNN thực hiện phân lớp hoa



# Ví dụ: Nhận dạng Hoa dựa trên Học sâu (Deep learning)

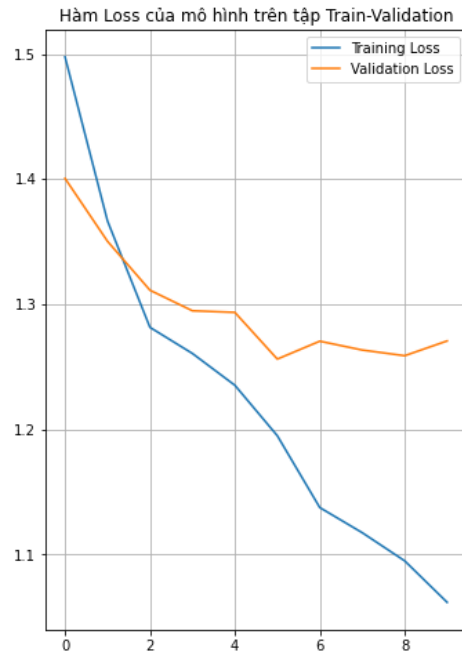
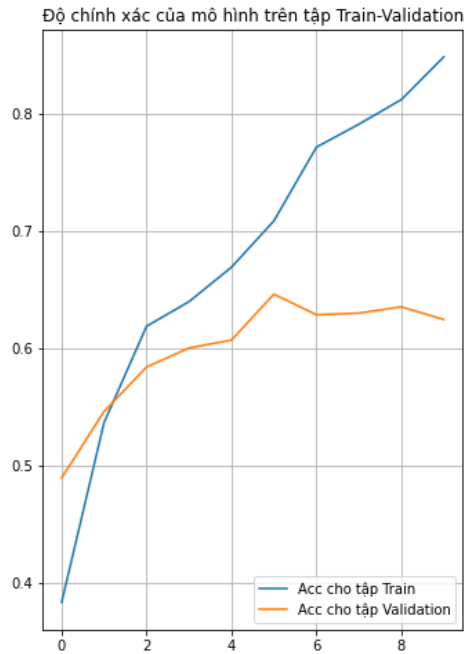
## Thiết kế mạng CNN

```
1 num_classes = 5
2
3 model = Sequential([
4     layers.experimental.preprocessing.Rescaling(1./255,
5                                                 input_shape=(img_height, img_width, 3)),
6     layers.Conv2D(16, 3, padding='same', activation='relu'),
7     layers.MaxPooling2D(),
8     layers.Conv2D(32, 3, padding='same', activation='relu'),
9     layers.MaxPooling2D(),
10    layers.Conv2D(64, 3, padding='same', activation='relu'),
11    layers.MaxPooling2D(),
12    layers.Flatten(),
13    layers.Dense(128, activation='relu'),
14    layers.Dense(num_classes)
15 ])
```



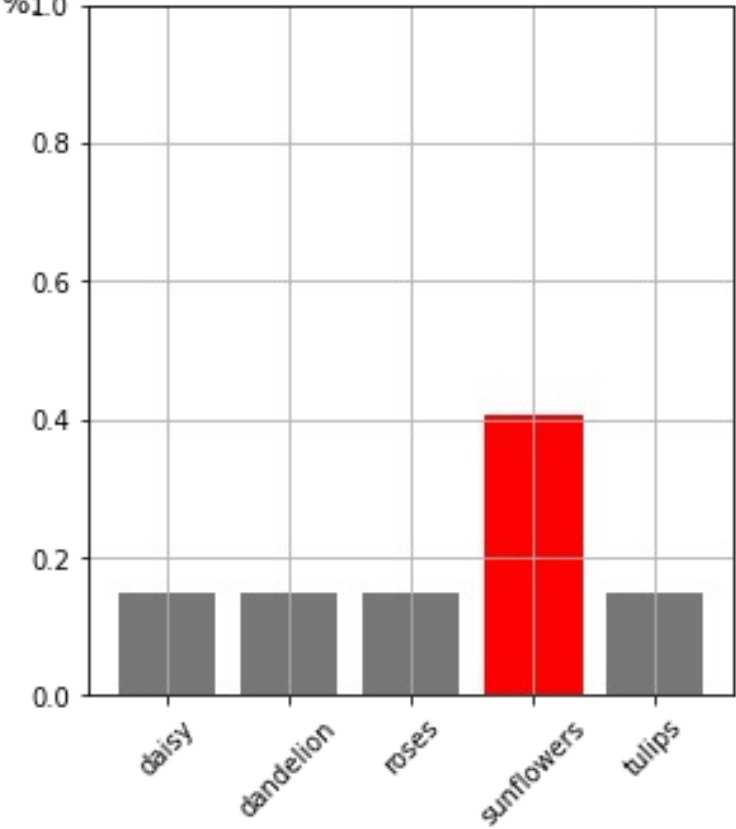
# Ví dụ: Nhận dạng Hoa dựa trên Học sâu (Deep learning)



## Acc và Loss của model:



## Sử dụng model phân lớp hoa

Đây là hoa SUNFLOWERS với độ tin cậy đạt 40.46 %



 +  TensorFlow +  Keras





QUESTIONS

**Q & A**

ANSWERS