Psychoinformatics & Neuroinformatics

Week 13

Image Processing & Computer Vision

by Tsung-Ren (Tren) Huang 黃從仁

Applications of Computer Vision

Automated inference on criminality using face images:

(a) Three samples in criminal ID photo set S_c.

(b) Three samples in non-criminal ID photo set S_n

Figure 1. Sample ID photos in our data set.

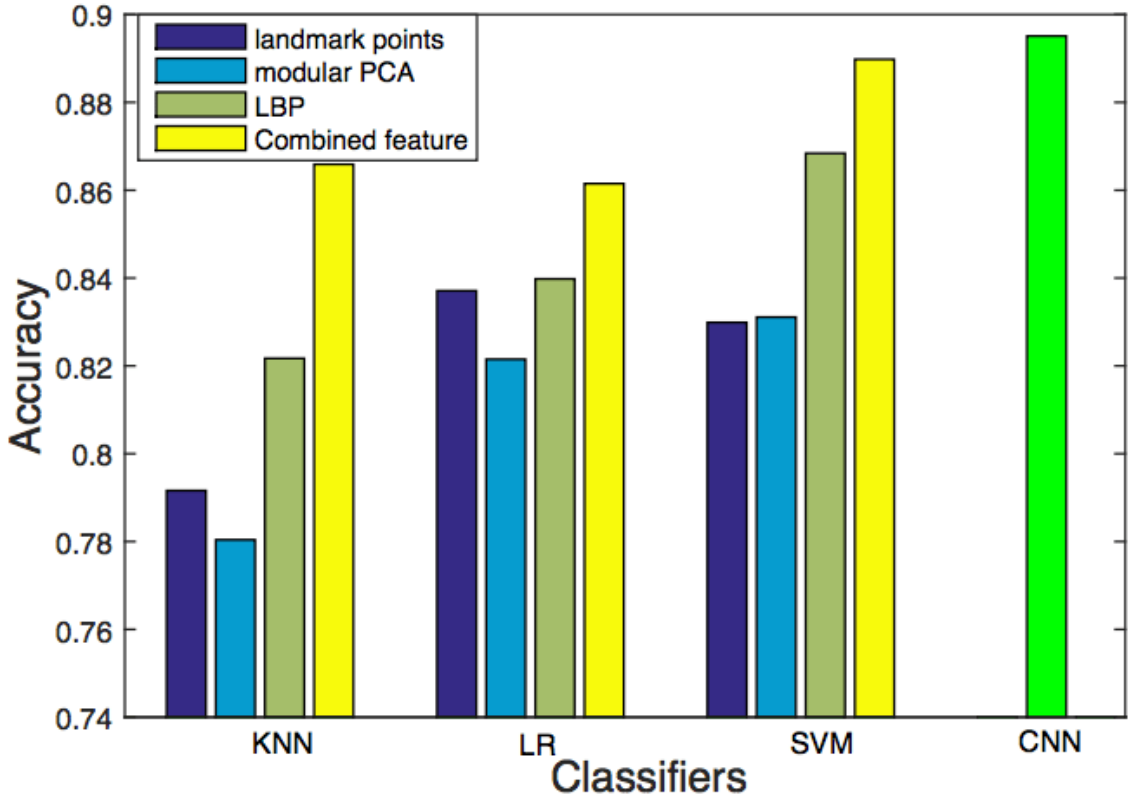


Figure 2. Accuracy of all four classifiers in all thirteen cases.

Topics for today

Image Processing

The processing unit is a pixel

Traditional Computer Vision

The processing unit is an object

DL-based Computer Vision

The foundation is the convolutional NN

Topics for today

Image Processing

The processing unit is a pixel

Traditional Computer Vision

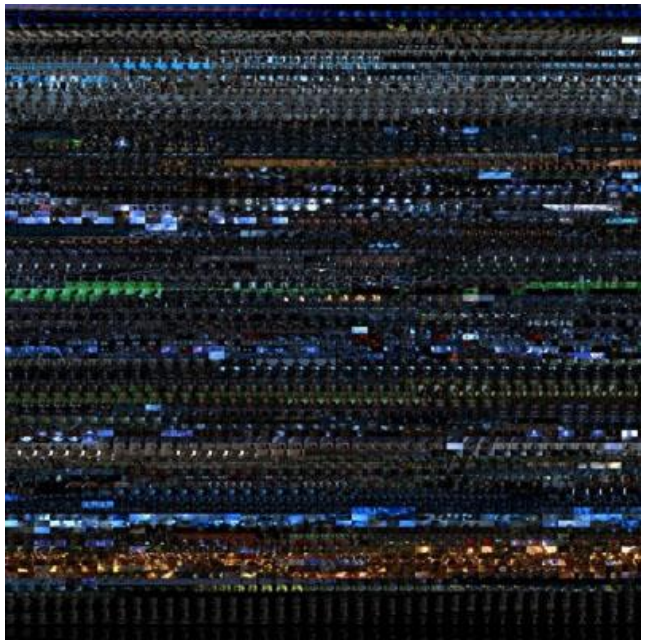
The processing unit is an object

DL-based Computer Vision

The foundation is the convolutional NN

Examples of Data/Image Visualization

The frames of two movies:



Graduation photos
of different generations:

The Class of 1988 (left panel)

The Class of 1988 (right panel)

The Class of 1967 (left panel)

The Class of 1967 (right panel)

Two Python Packages

Matplotlib for visualization & PIL/Pillow for processing

Matplotlib: Visualization with Python

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

- Create publication quality plots.
- Make interactive figures that can zoom, pan, update.
- Customize visual style and layout.
- Export to many file formats.
- Embed in JupyterLab and Graphical User Interfaces.
- Use a rich array of third-party packages built on Matplotlib.

The Python Imaging Library adds image processing capabilities to your Python interpreter.

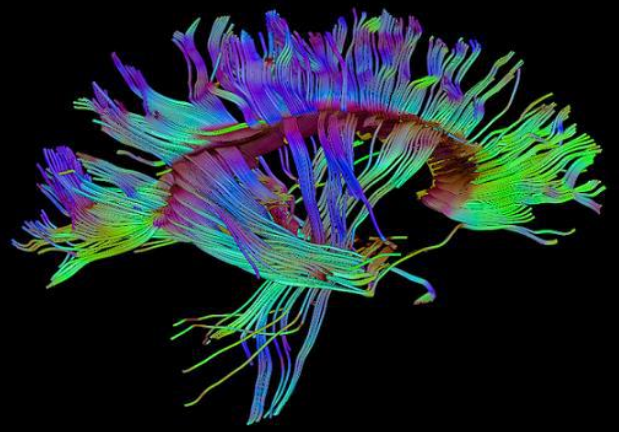
This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities.

The core image library is designed for fast access to data stored in a few basic pixel formats. It should provide a solid foundation for a general image processing tool.

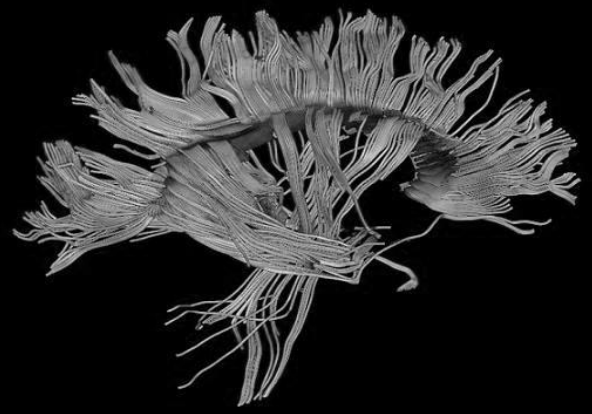
Example 1: Color to gray

You can do it by yourself or by pillow:

```
import matplotlib.pyplot as plt, numpy as np
img1=plt.imread('DTI.jpg')
plt.imshow(img1); plt.figure()
img2=np.mean(img,2)
plt.imshow(img2,cmap=plt.cm.gray)
```



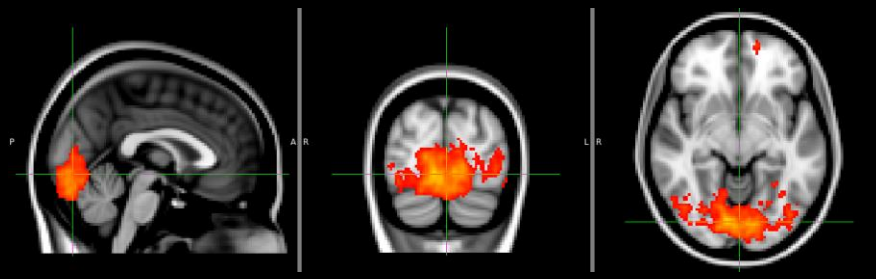
```
from PIL import Image
img1=Image.open('DTI.jpg')
img2=img.convert('L')
img2.show(); img2.save('DTI_gray.jpg')
```



Example 2: Image Contrast

You can run +-*/ on images:

```
import numpy as np,
import matplotlib.pyplot as plt
img=[]
img.append(np.float64(plt.imread('MRI1.jpg')))
img.append(np.float64(plt.imread('MRI2.jpg')))
img.append(img[1]-img[0]) #contrast
for i in range(3):
 plt.subplot(1,3,i+1); plt.axis('off')
 plt.imshow(img[i],cmap=plt.cm.gray)
```



Example 3: Image Blending & Filtering

You can run +-*/ on images:

```
from PIL import Image
from PIL.ImageFilter import CONTOUR
img1=Image.open('house.jpg')
img2=Image.open('face.jpg')
k=np.arange(1,10,2)/10.0
for i in range(5):
    hybrid=Image.blend(img1,img2,k[i])
    hybrid=hybrid.filter(CONTOUR)
    hybrid.show()
```

Topics for today

Image Processing

The processing unit is a pixel

Traditional Computer Vision

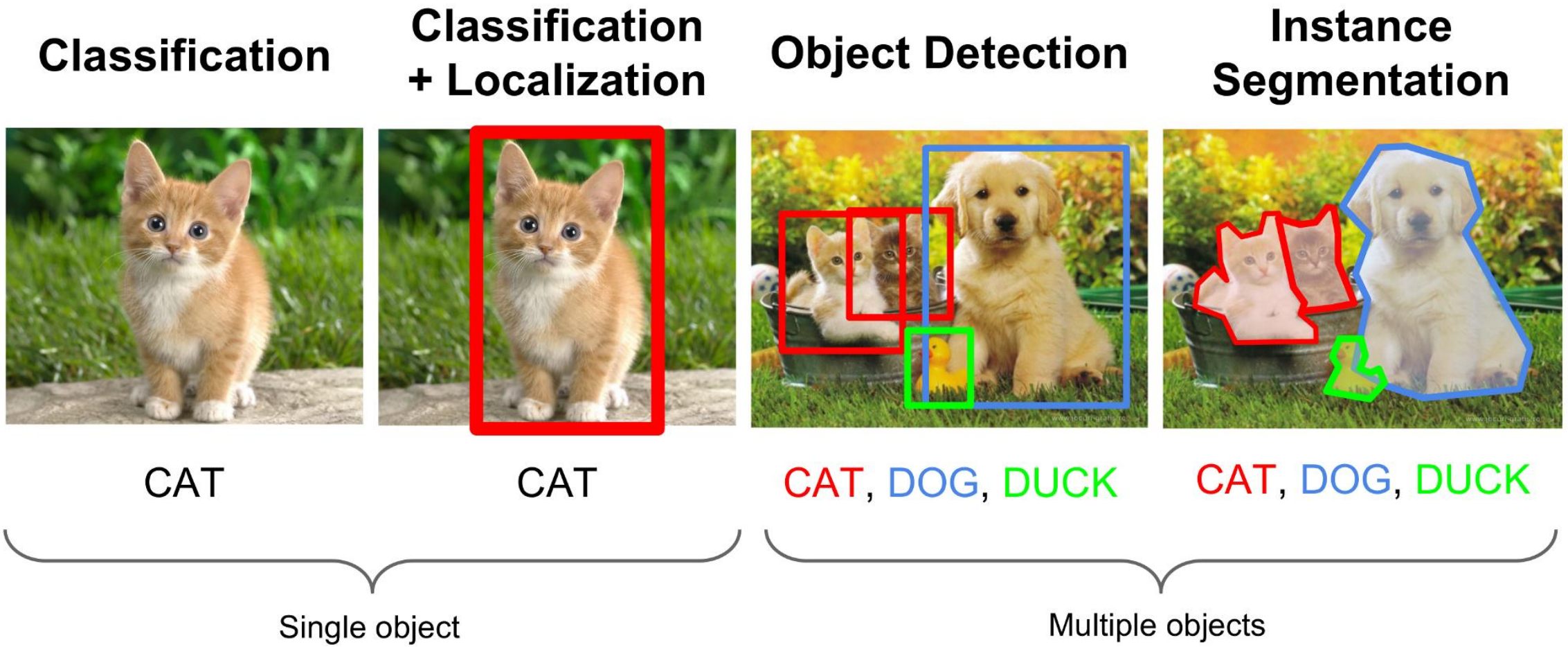
The processing unit is an object

DL-based Computer Vision

The foundation is the convolutional NN

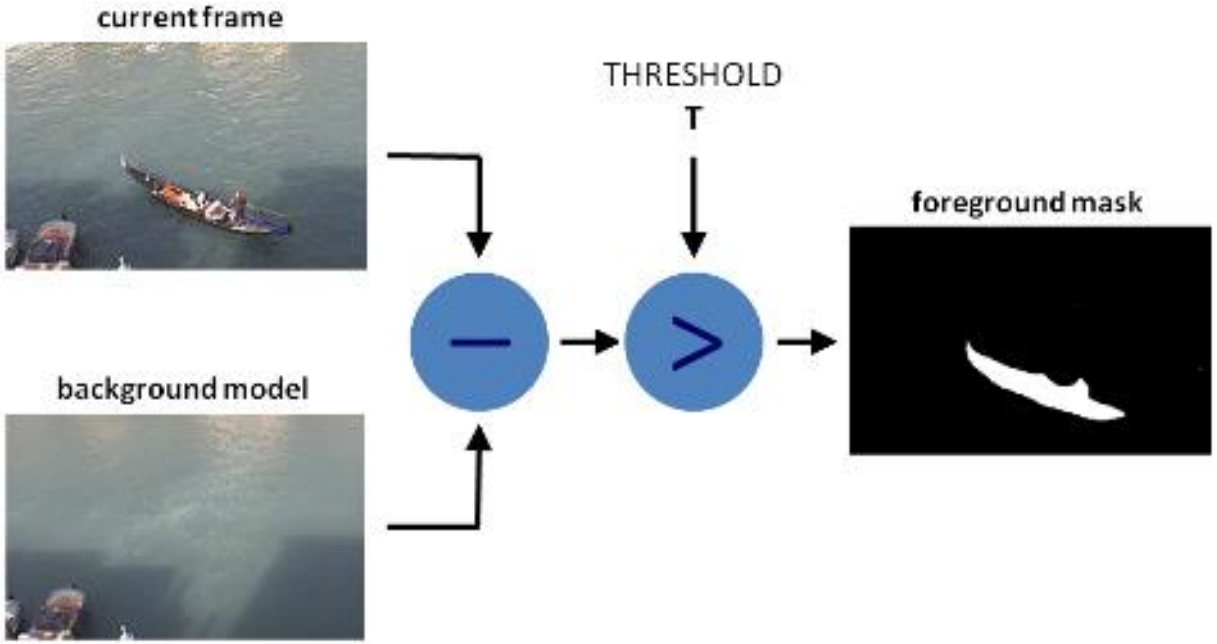
Computer Vision (1/2)

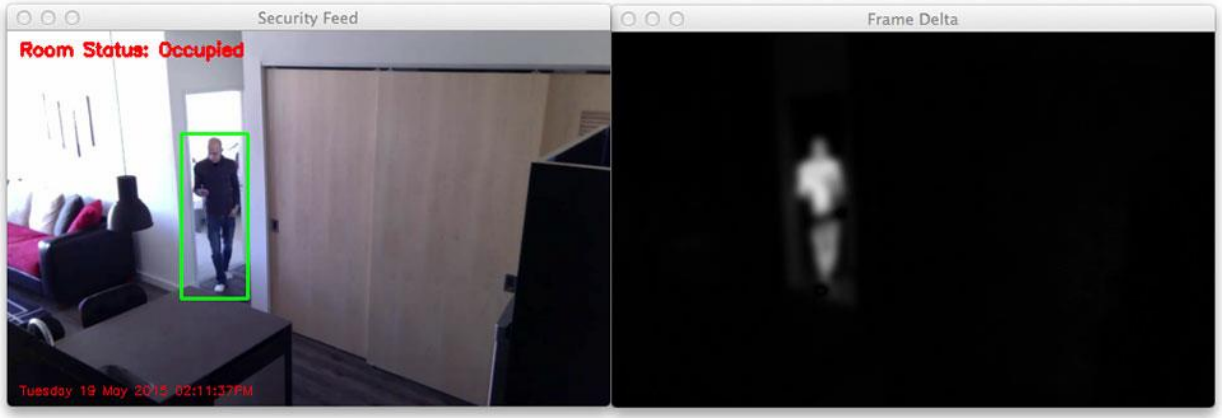
From what to where:



Computer Vision (2/2)

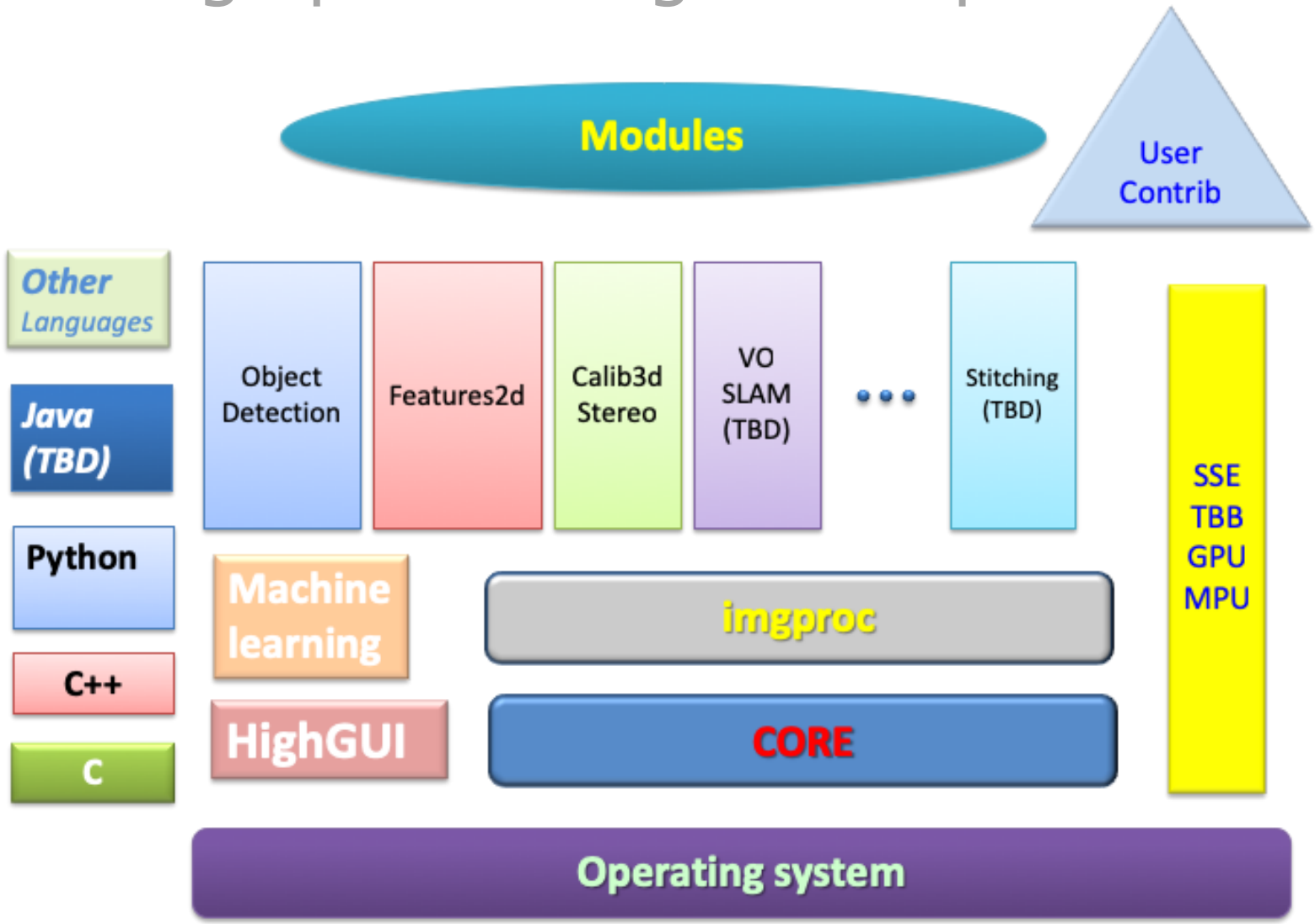
From images to videos:





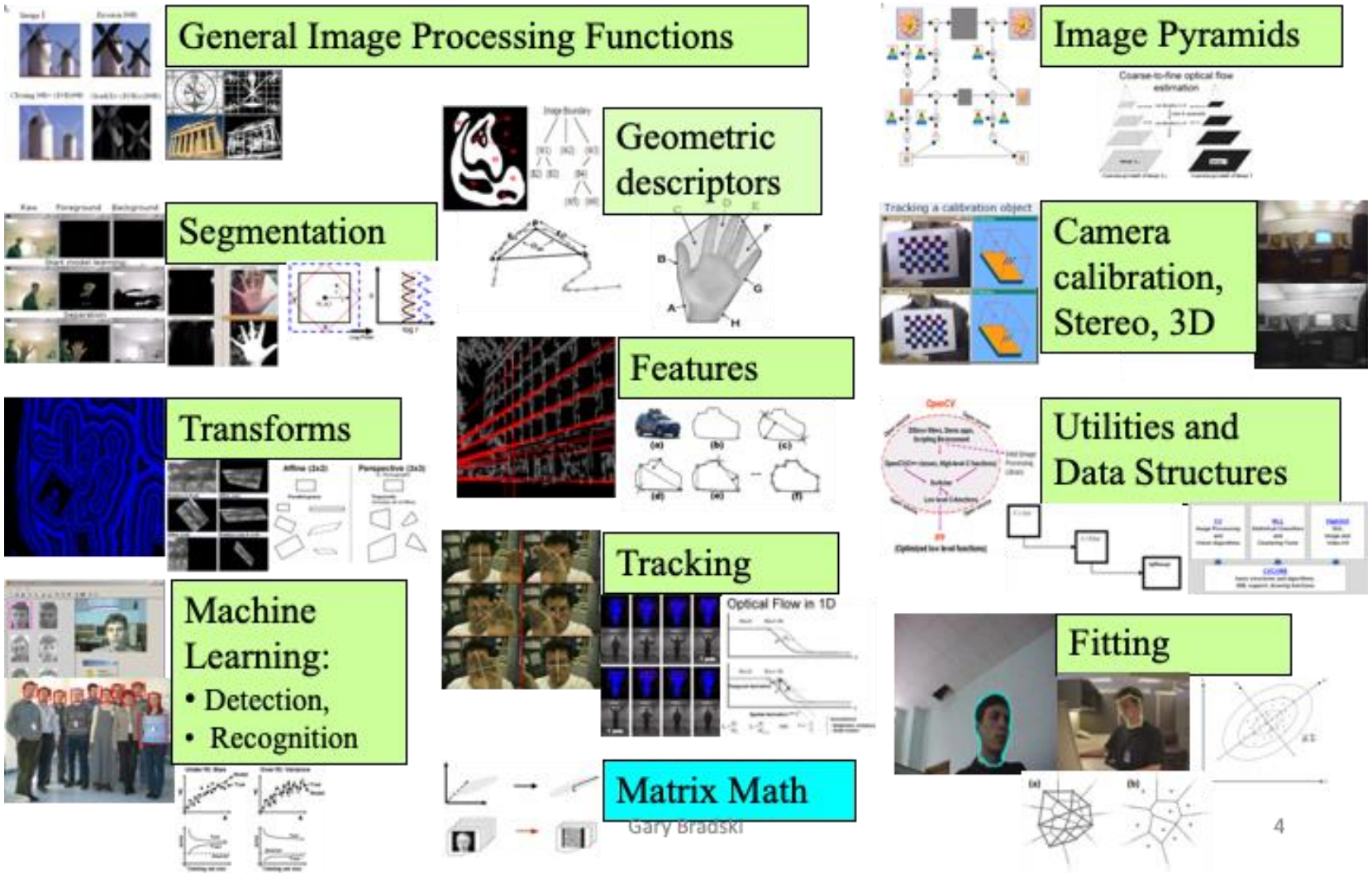
OpenCV (1/2)

A library for image processing + computer vision:



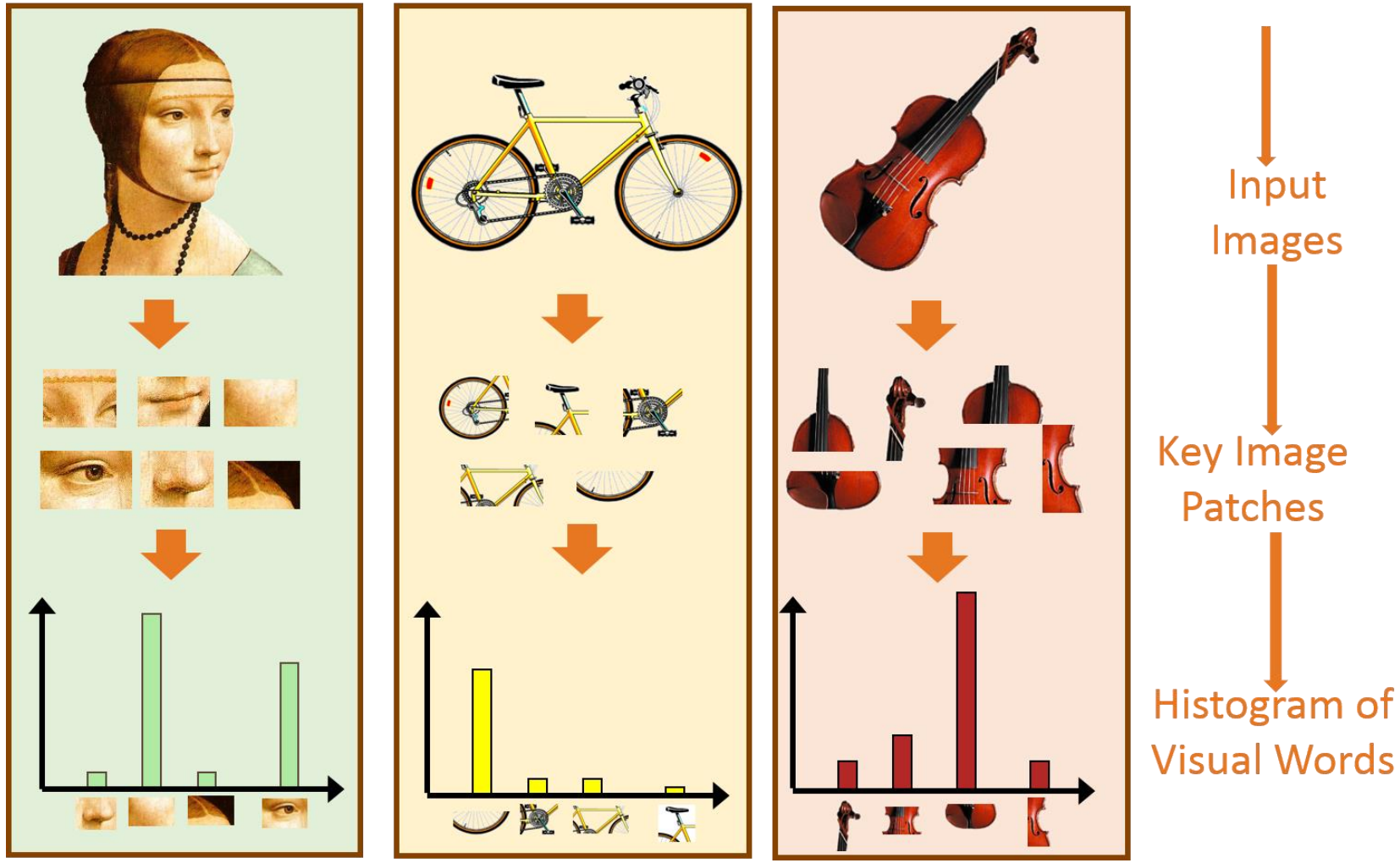
OpenCV (2/2)

More than 2,500 algorithms:



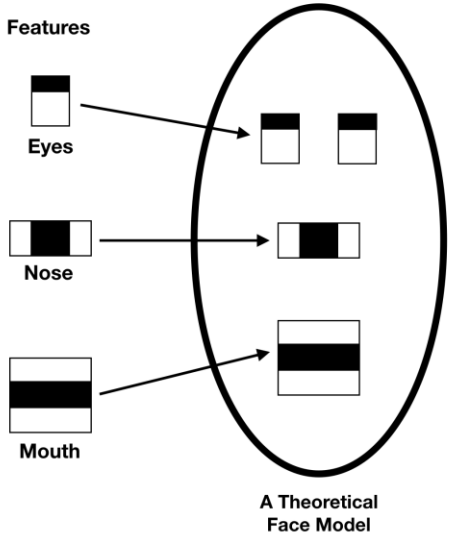
Features for Object Recognition

Bag of Visual Words (BOVW) is just one example:

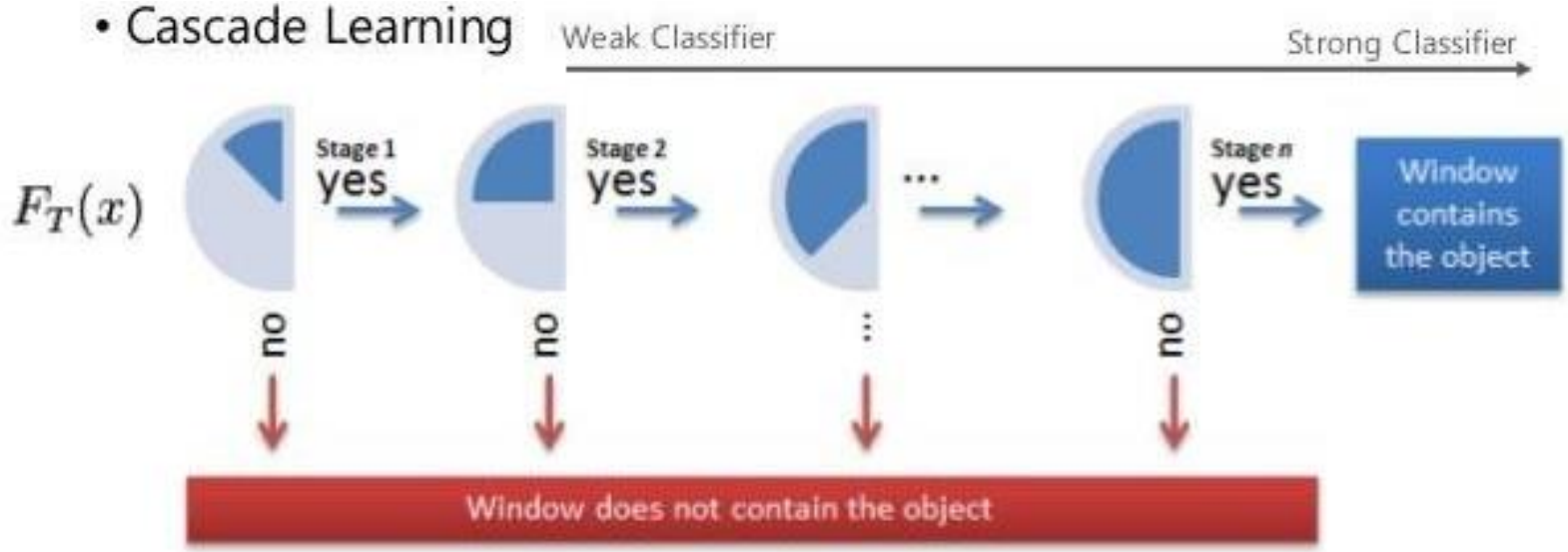


Face Recognition: Haar Cascades

AdaBoost on basic edge/line features:



• Cascade Learning



Topics for today

Image Processing

The processing unit is a pixel

Traditional Computer Vision

The processing unit is an object

DL-based Computer Vision

The foundation is the convolutional NN

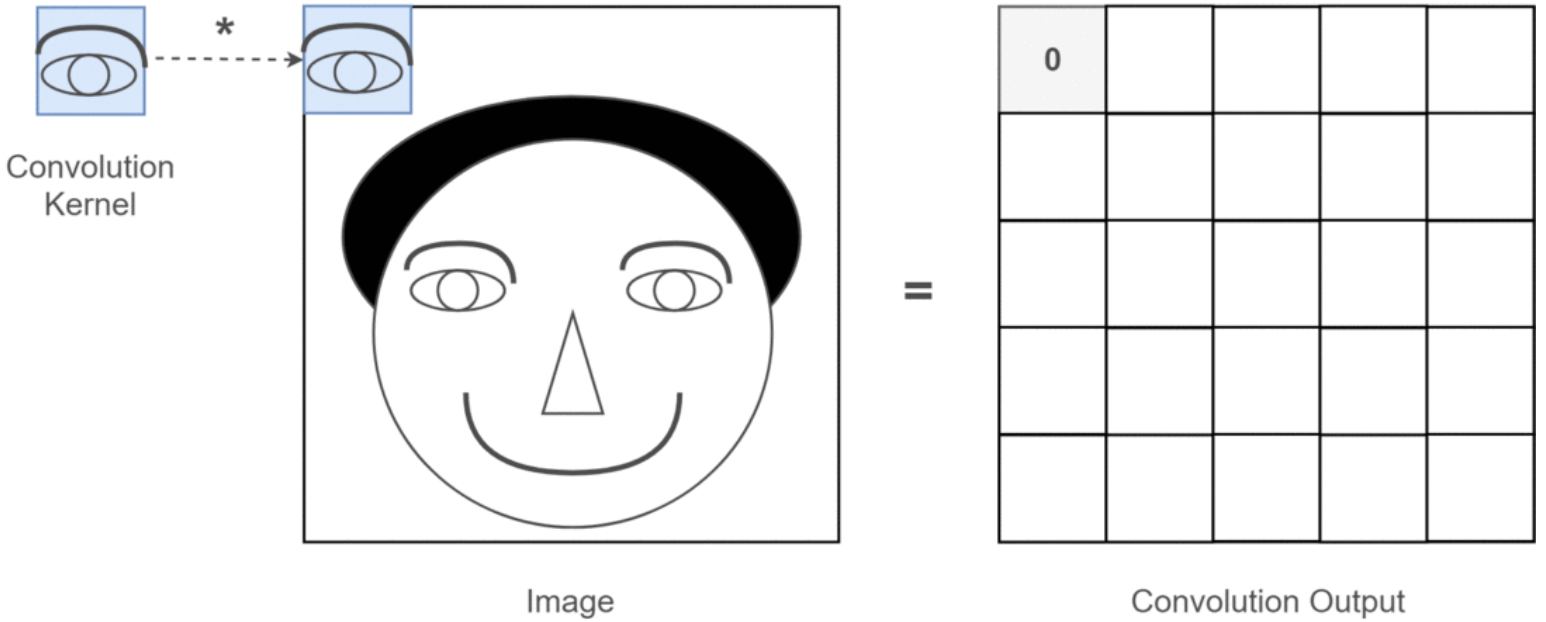
2D Convolution/Filtering

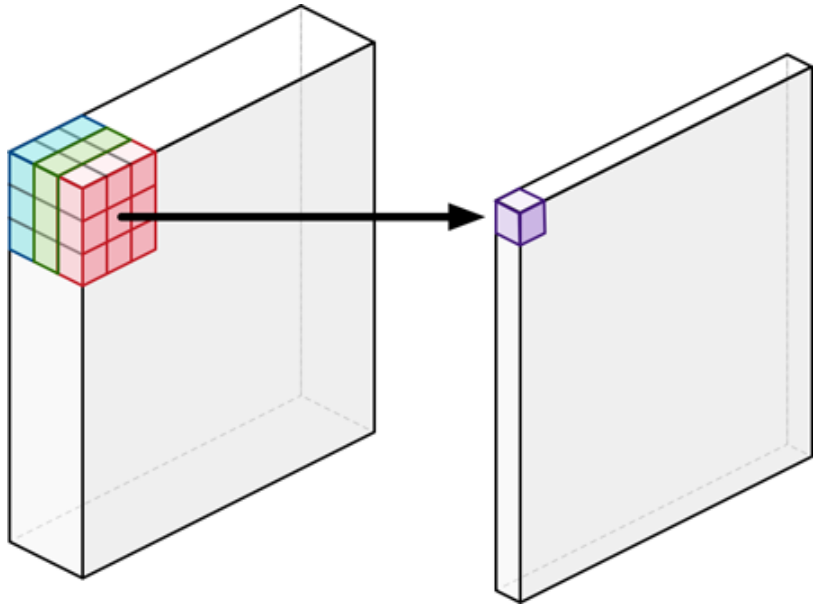
Conv2D(img, krnl)=filter2D(img,上下顛倒左右相反(krnl))

X_{11}	X_{12}	X_{13}
X_{21}	X_{22}	X_{23}
X_{31}	X_{32}	X_{33}

W_{11}	W_{12}
W_{21}	W_{22}

Kernel size=3*3*3

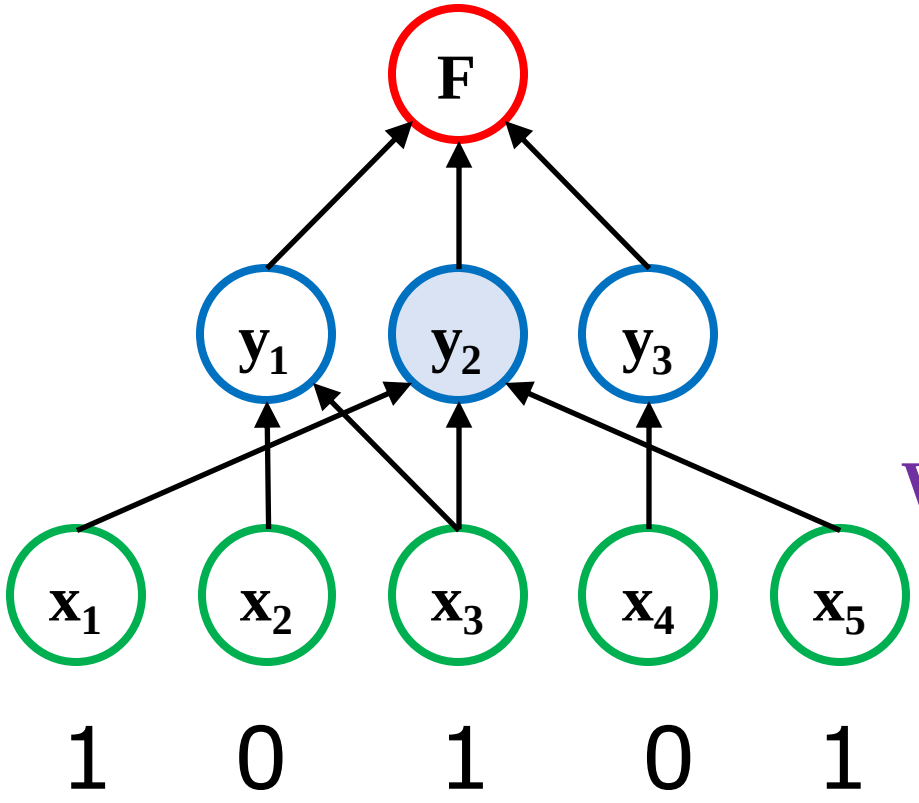


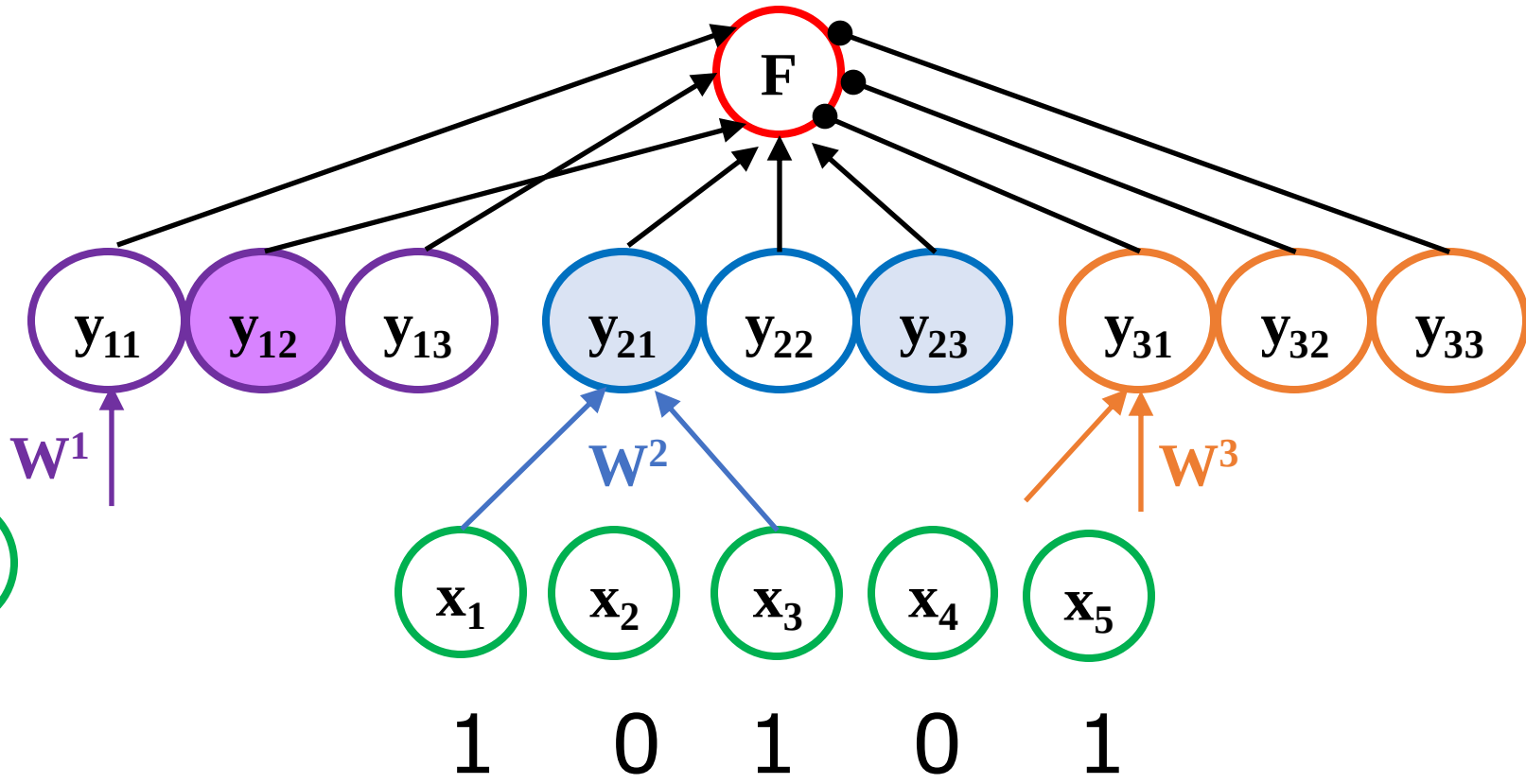


Non-shared vs. shared weights

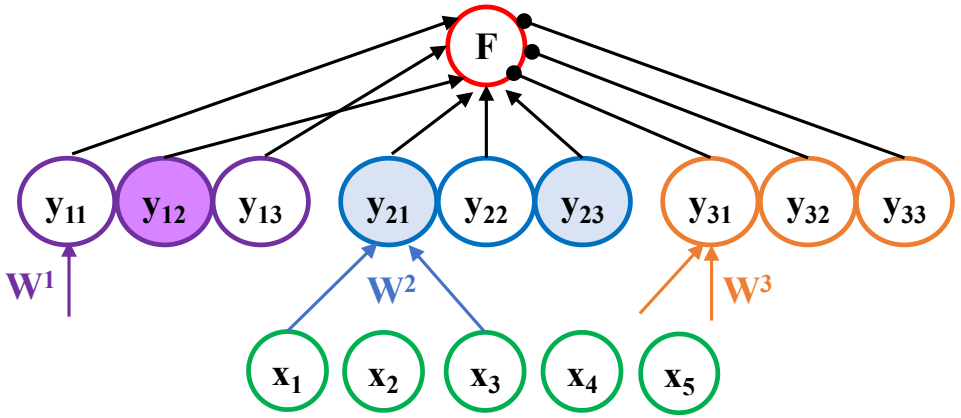
y_1 detects [0 1 1 0 0] globally
y_2 detects [1 0 1 0 1] globally
y_3 detects [0 0 0 1 0] globally

W^1 detects [0 1 0] locally
W^2 detects [1 0 1] locally
W^3 detects [1 1 0] locally





Convolutional Lingo



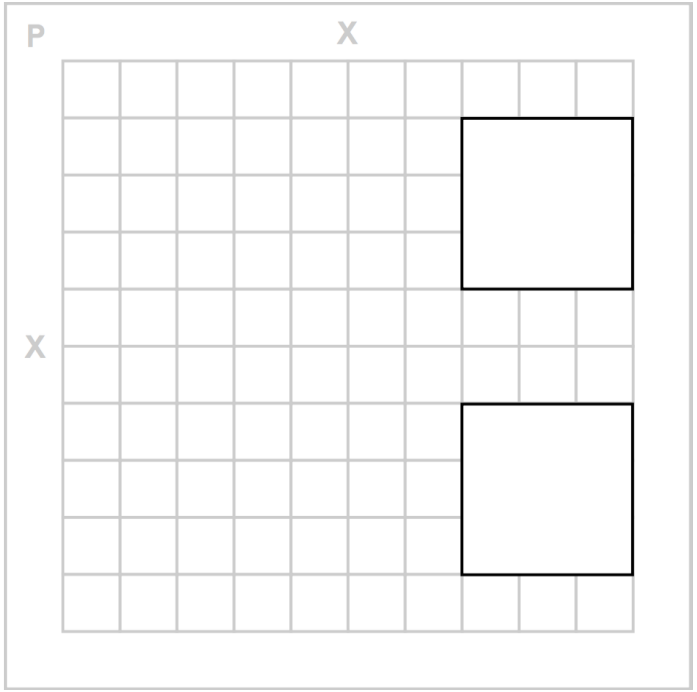
$$1 + \frac{X - F + 2P}{S}$$

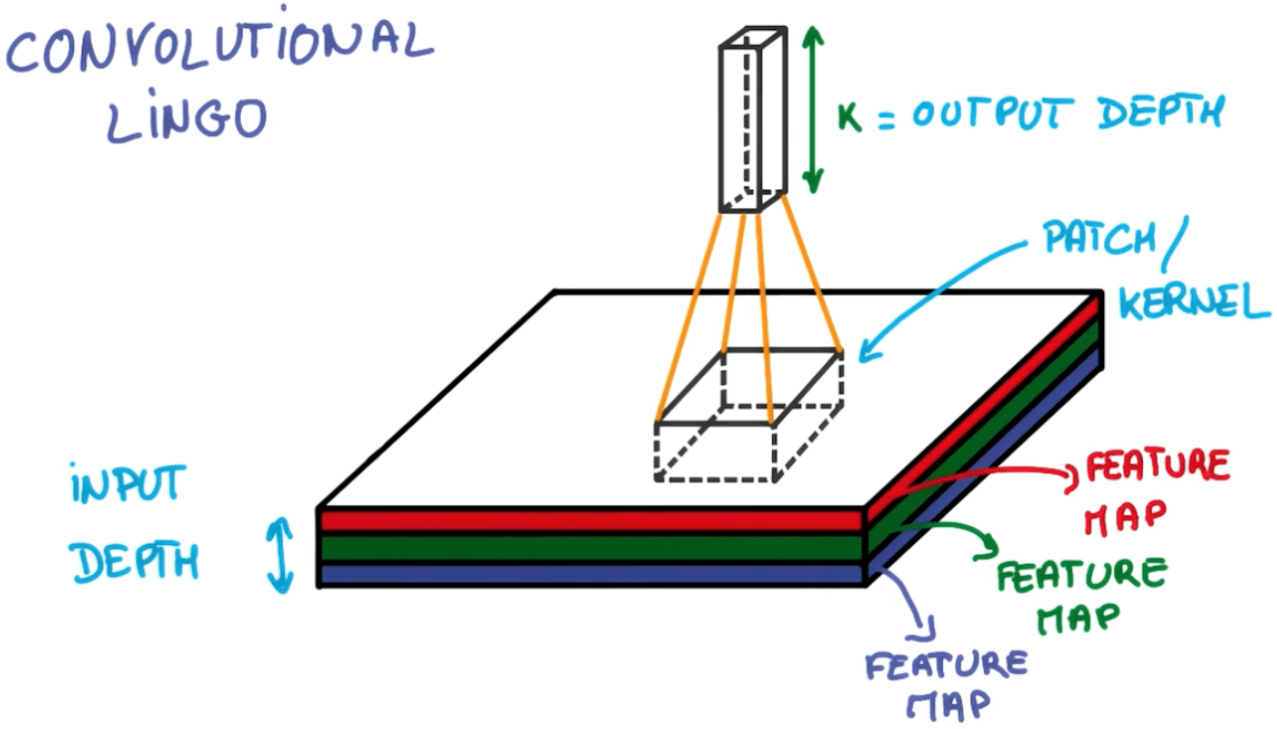
X = image size

F = filter size

P = padding

S = stride

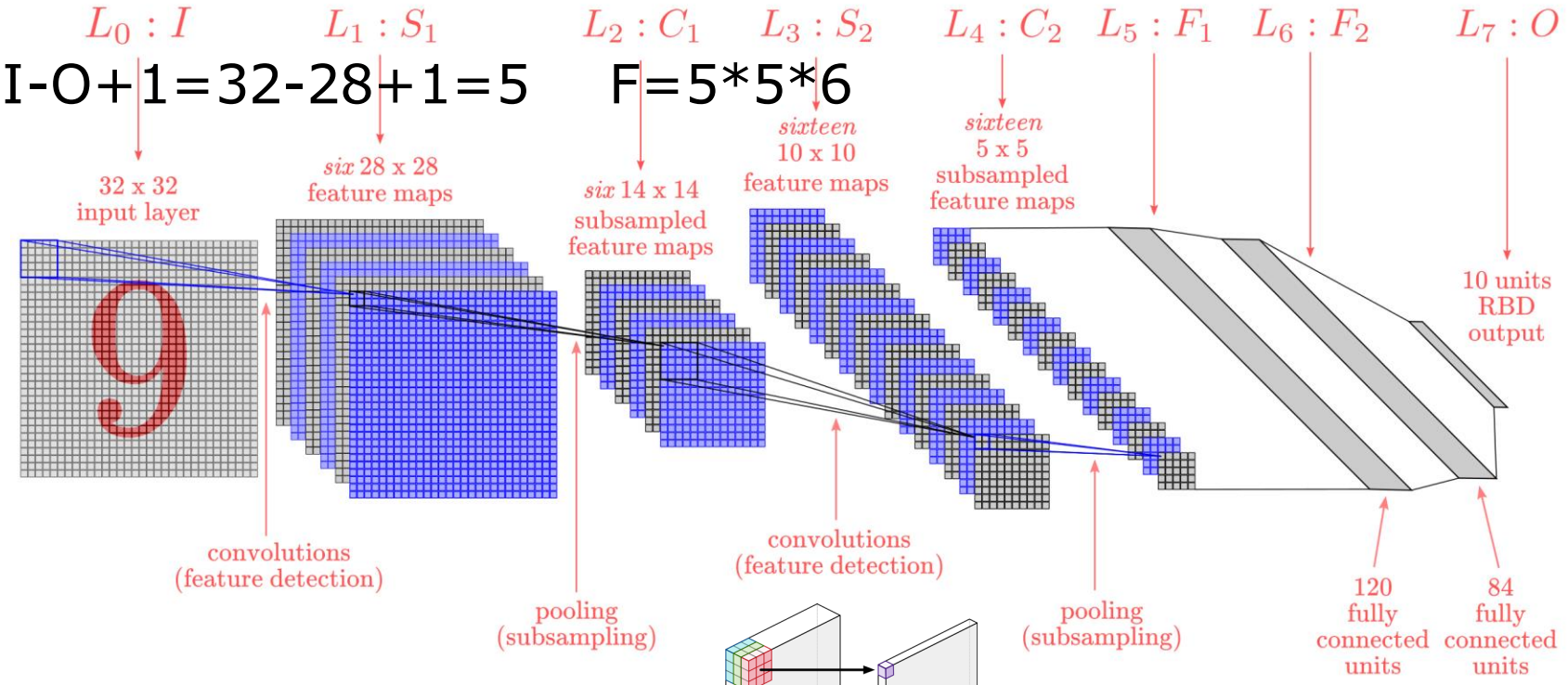


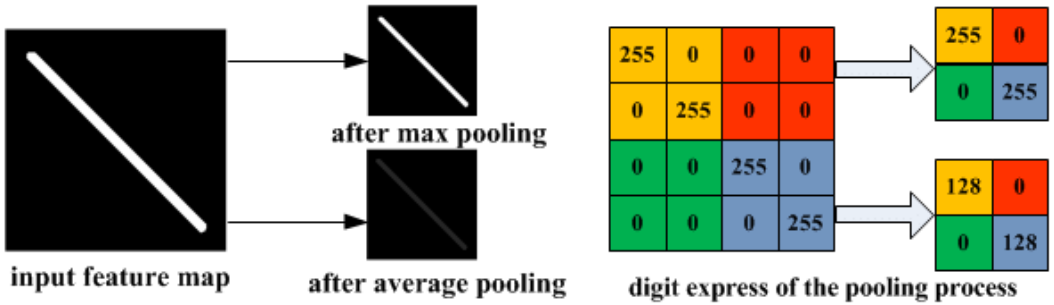


If (P,S)=(0,1),
then O=1+I-F
⇒ F=I-O+1

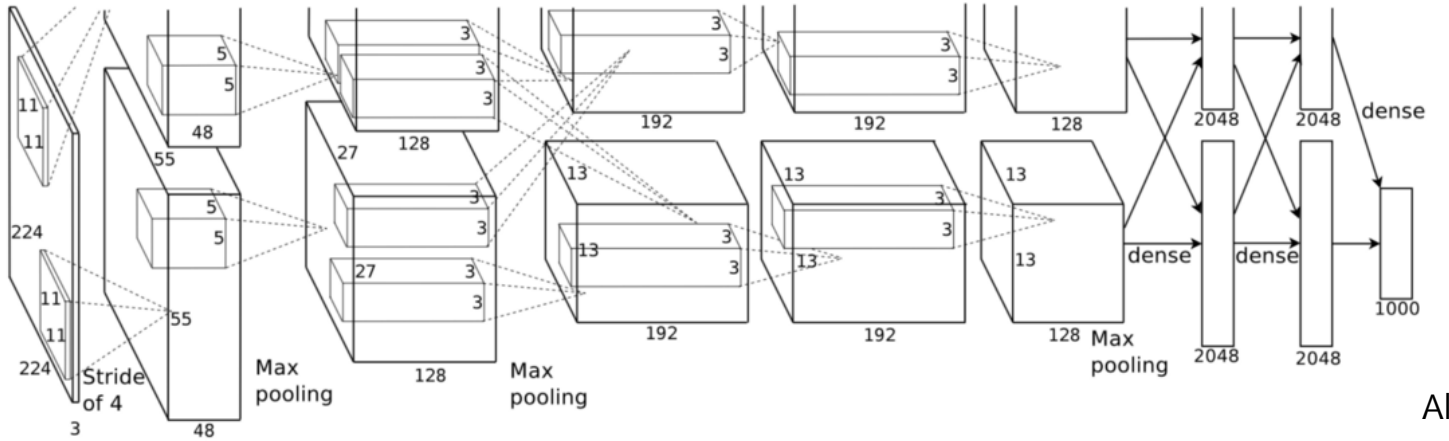
Basic CNN: LeNet

F=I-O+1=32-28+1=5 F=5*5*6

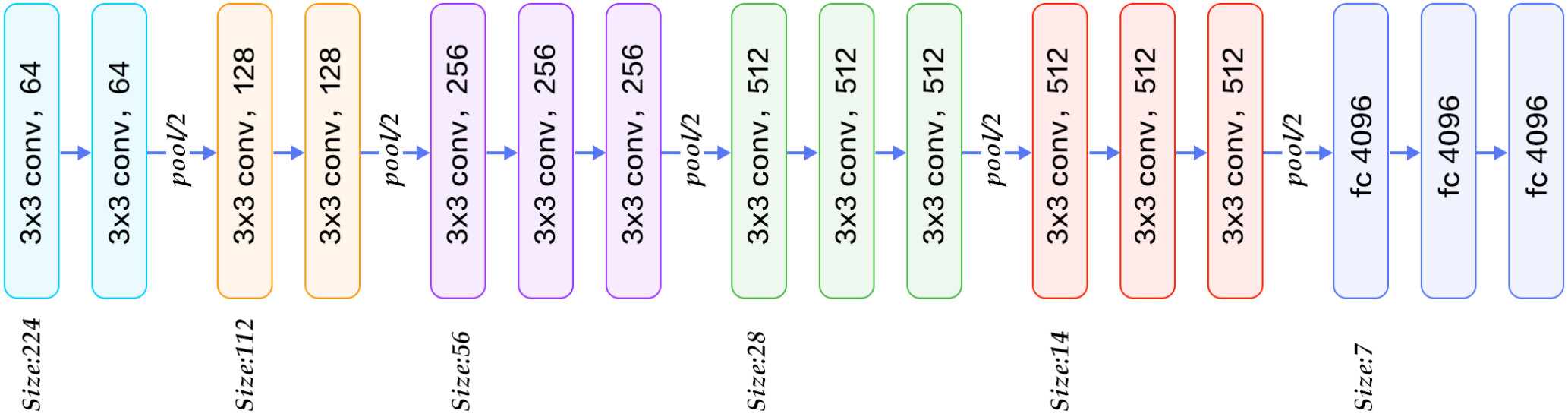




Evolution of CNNs (1/3): GPU+Deeper

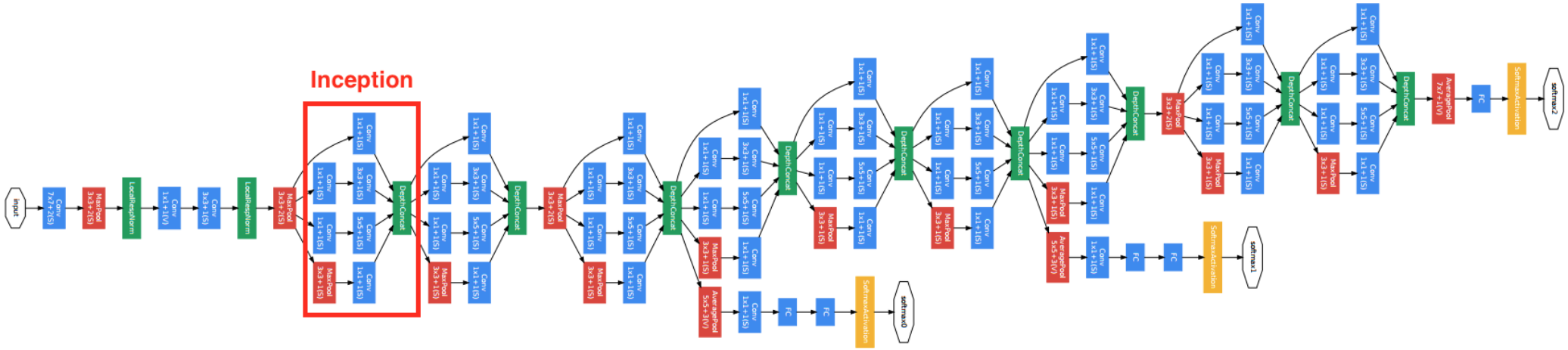


AlexNet (2012)

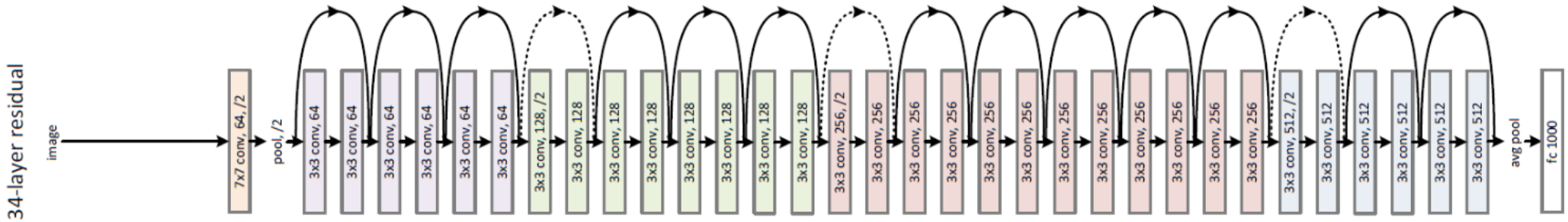


VGG (2014)

Evolution of CNNs (2/3): Wider/Ensemble



GoogLeNet (2014)

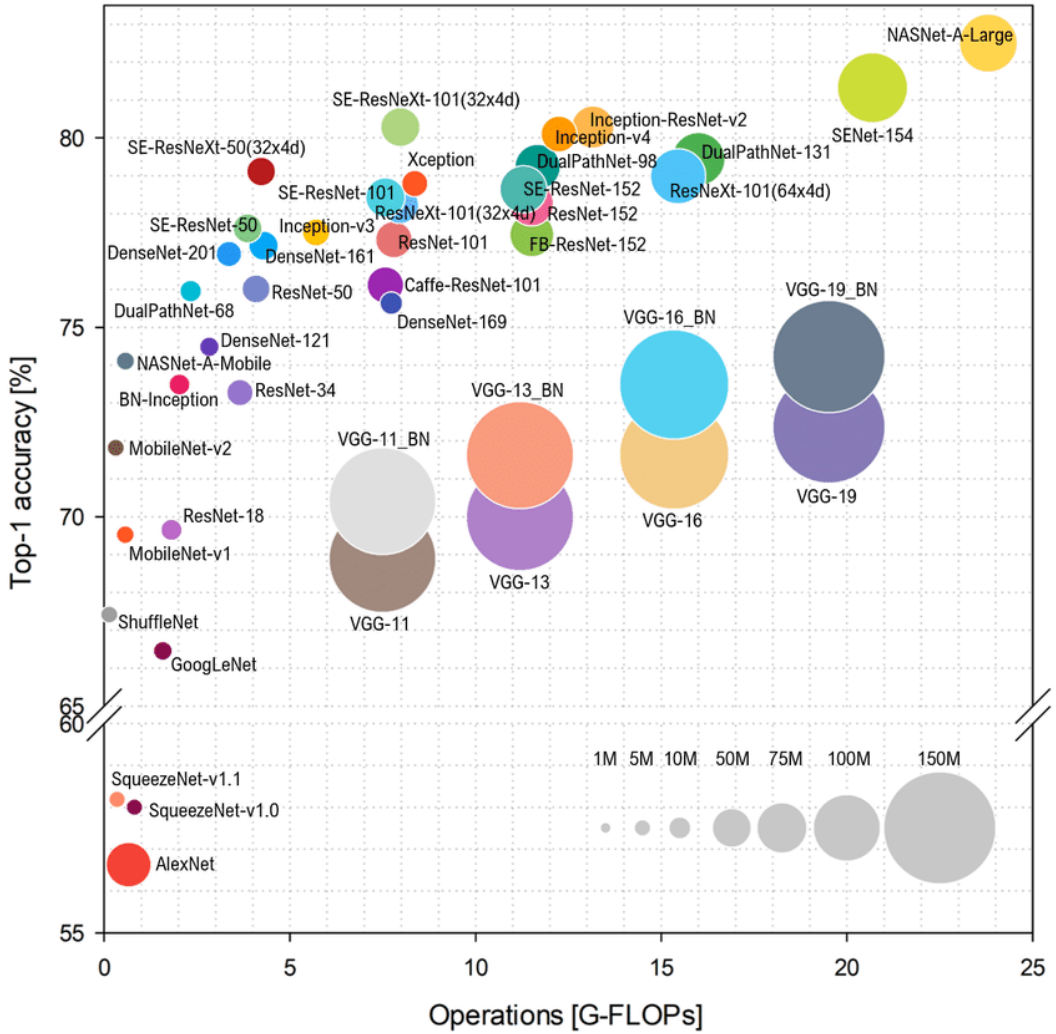


ResNet (2015)

Evolution of CNNs (3/3)

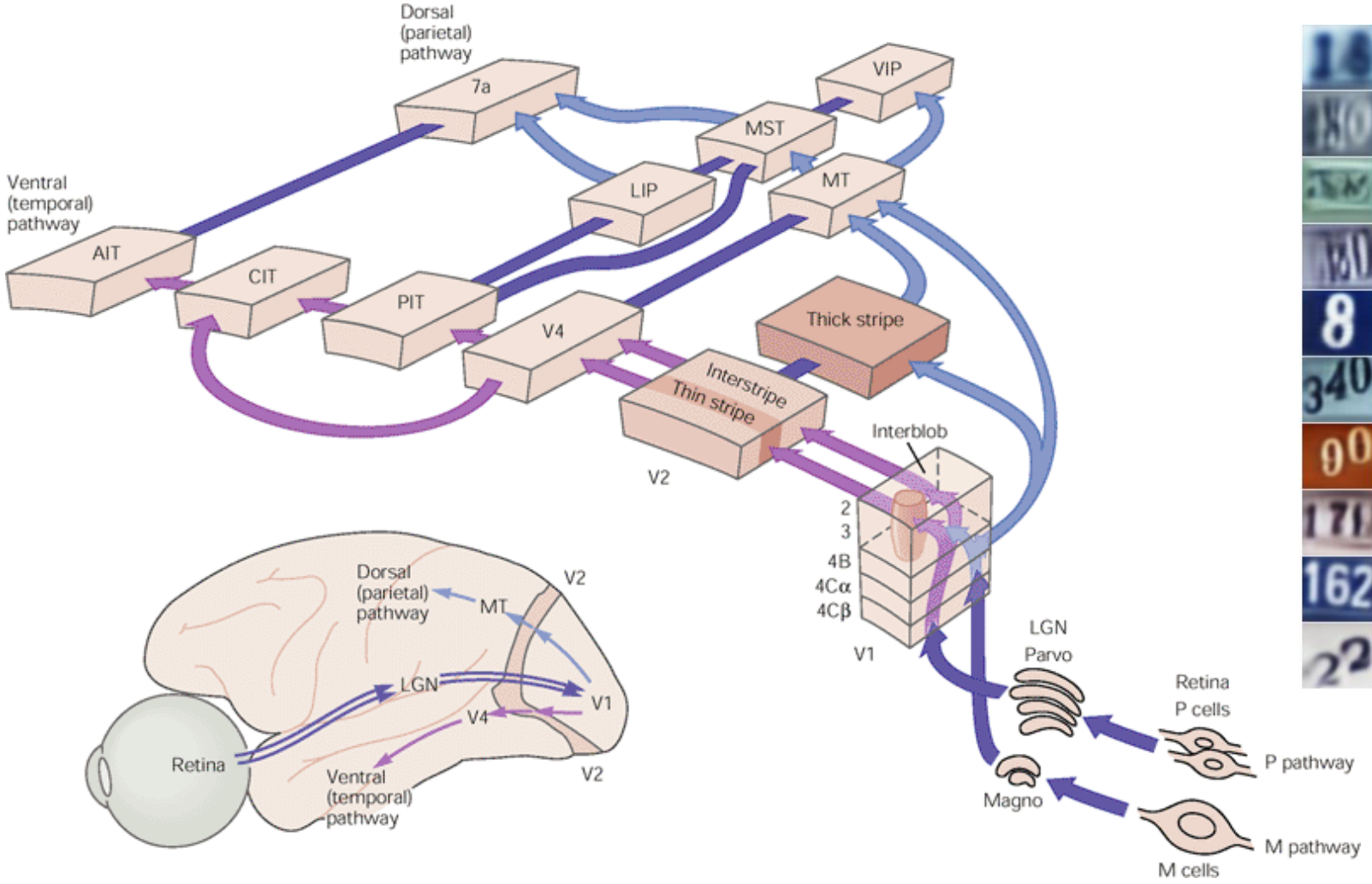
More complex models ≠ Better models

≠

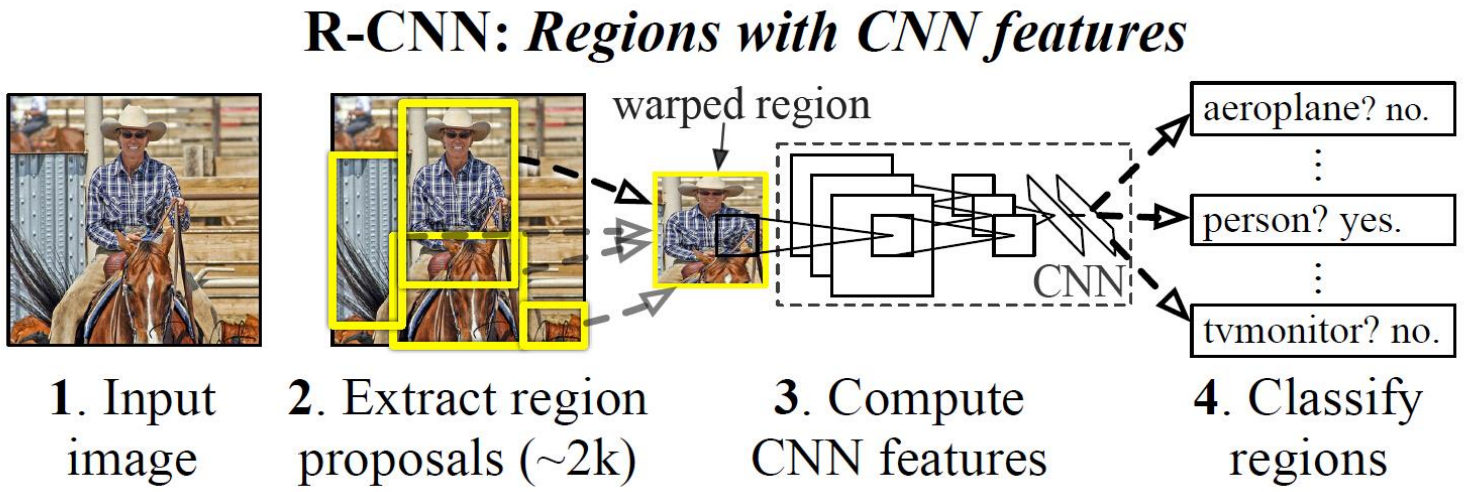


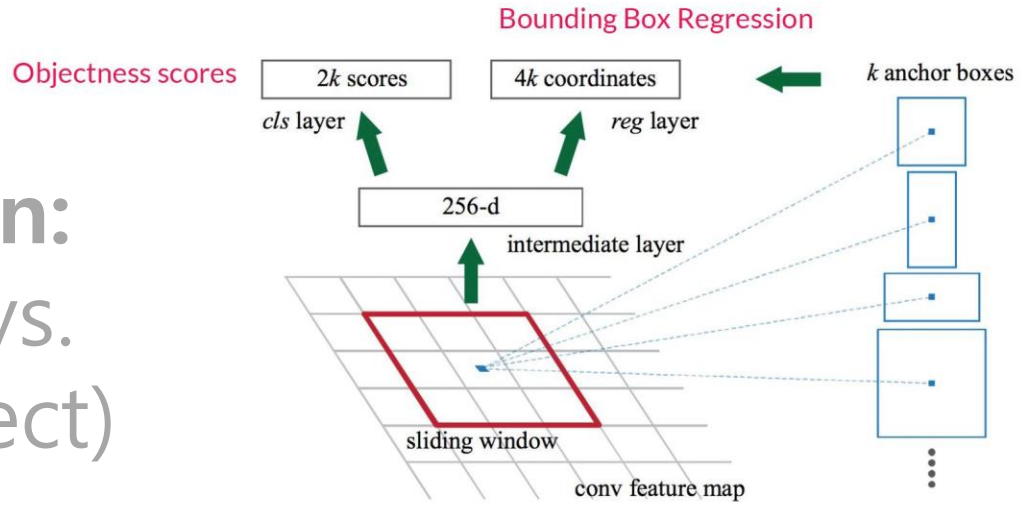
From “What” to “Where” Information

CV is incomplete & inefficient w/o processing “Where” info



Region-based CNN (Faster RCNN)

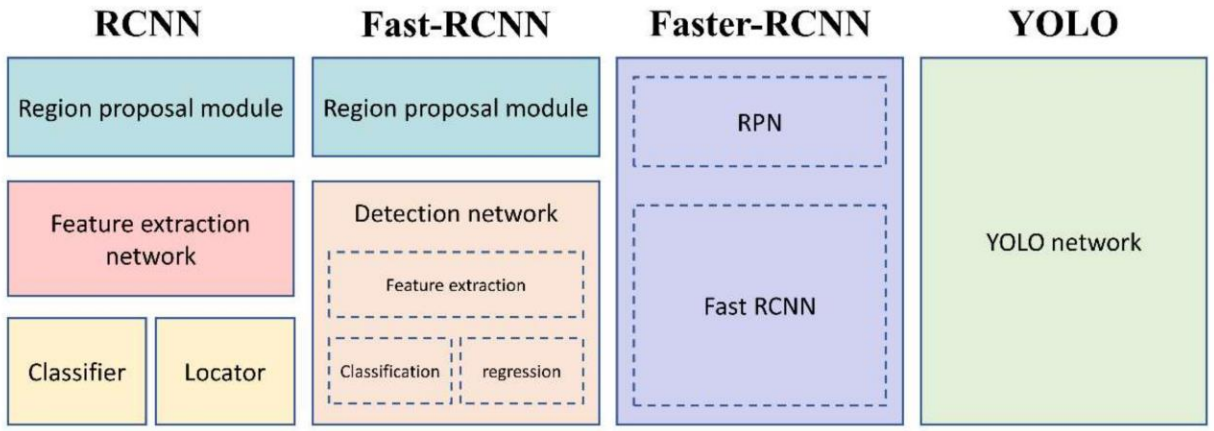


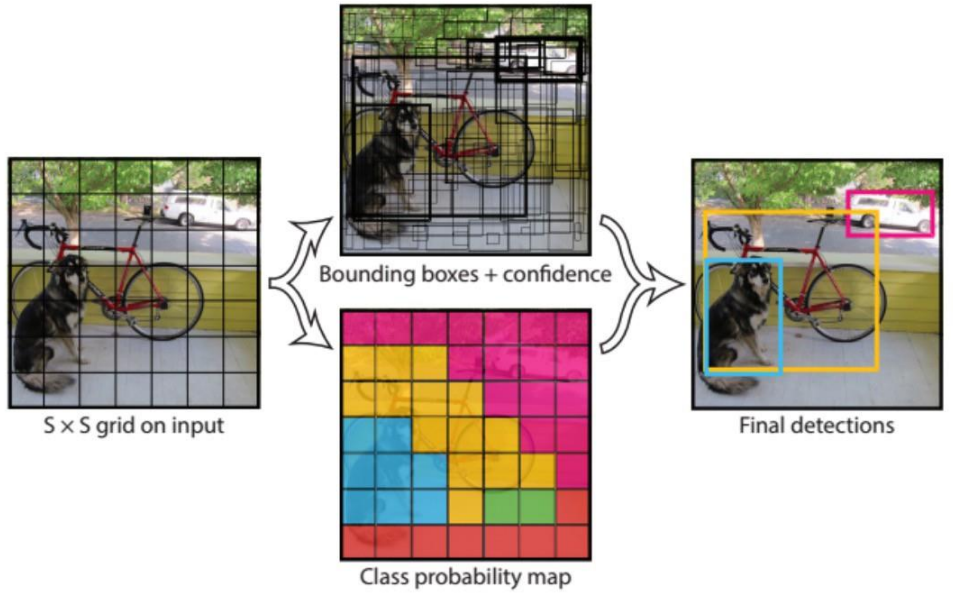


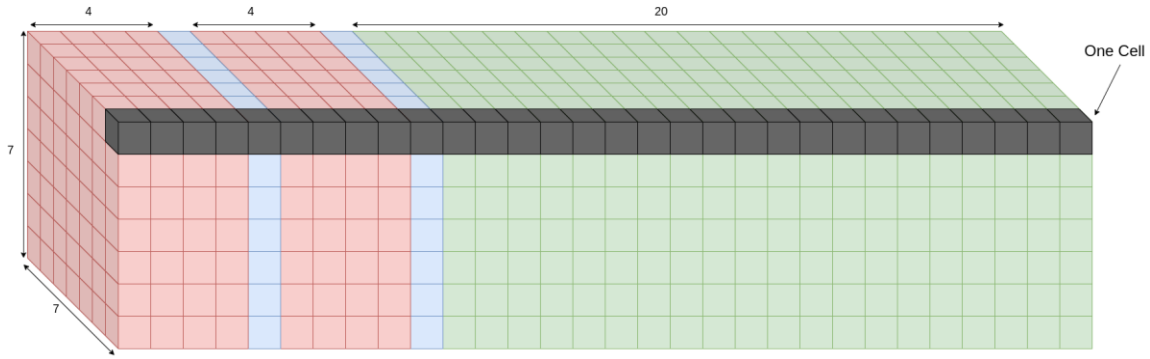
Objectness classification:
Foreground (an object) vs.
Background (not an object)

From RCNN to YOLO (U Only Look Once)

YOLO simultaneously predicts a object class &
its potential bounding boxes at the same time!







Topics for today

Image Processing

The processing unit is a pixel

Traditional Computer Vision

The processing unit is an object

DL-based Computer Vision

The foundation is the convolutional NN

Game Over