

# Computer Vision Problems

Image

Class



64x64

→ Cat? (0/1)

Neural Style  
Transfer



Object detection



# Deep Learning on large images



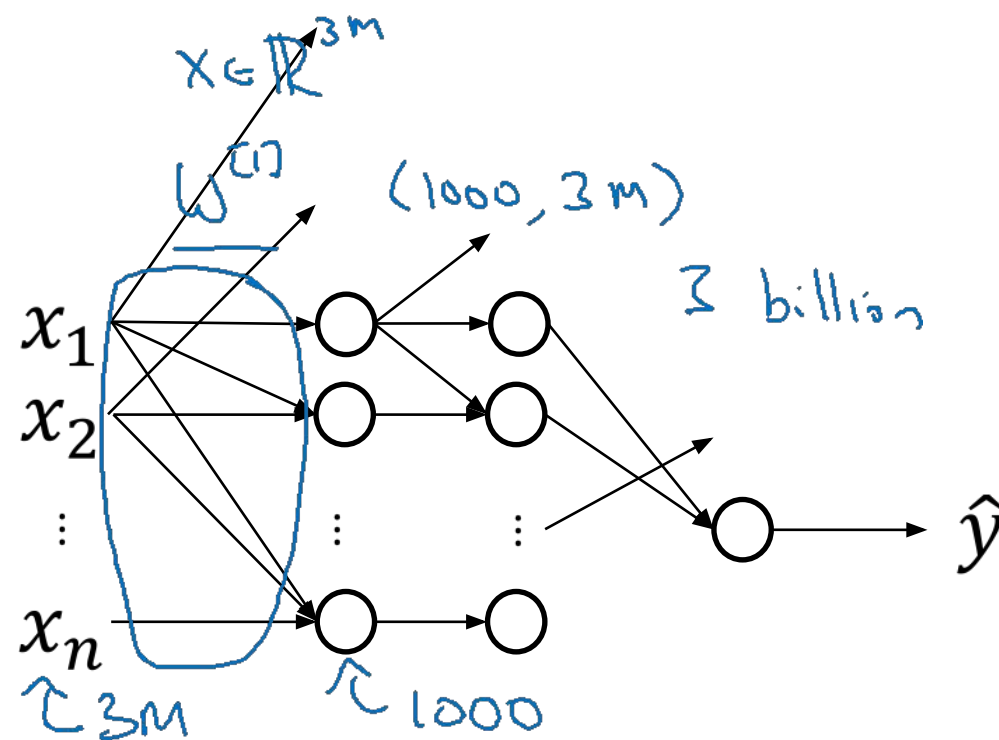
64x64 x 3

→ Cat? (0/1)

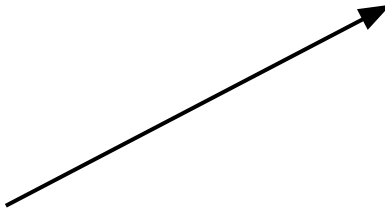
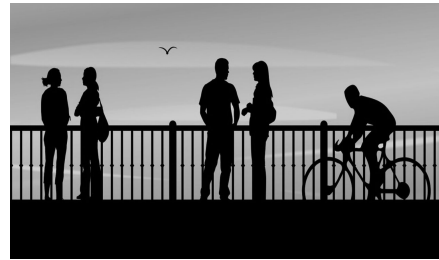
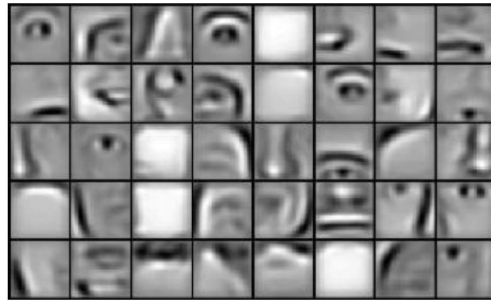
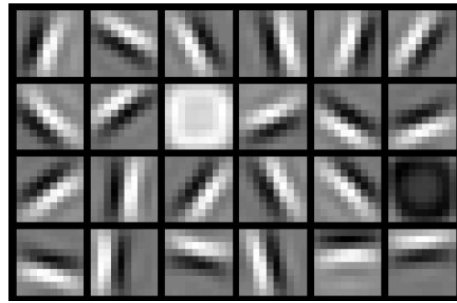
12288



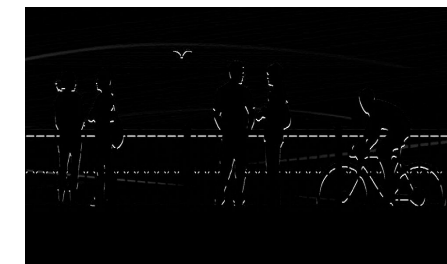
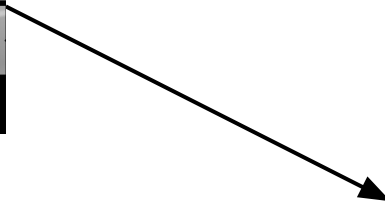
1000x1000 x 3  
= 3 million



# Computer Vision Problem



vertical edges



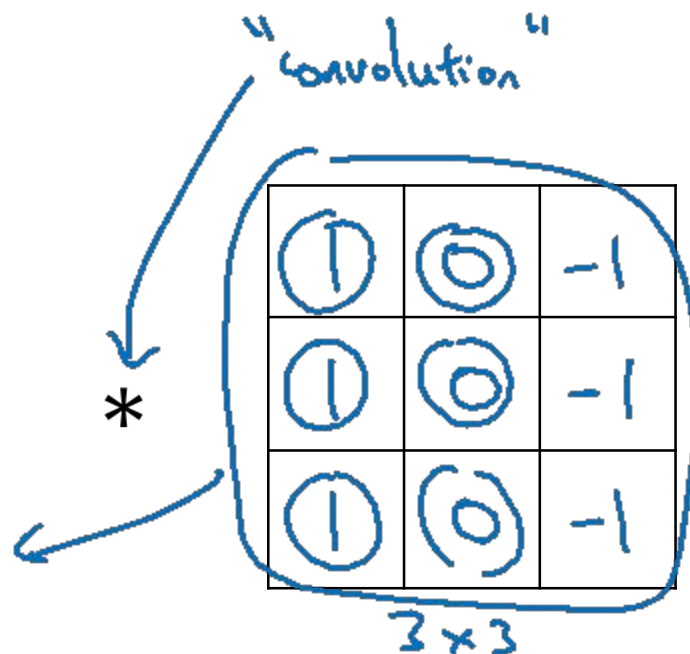
horizontal edges

# Vertical edge detection

$$\rightarrow 3 \times 1 + 1 \times 1 + 2 \times 1 + 0 \times 0 + 5 \times 0 + 7 \times 0 + 1 \times -1 + 8 \times -1 + 2 \times -1 = -5$$

3 <sup>1</sup>	0 <sup>01</sup>	1 <sup>01</sup>	2 <sup>-1</sup>	7	4
1 <sup>1</sup>	5 <sup>01</sup>	8 <sup>01</sup>	9 <sup>-1</sup>	3	1
2 <sup>1</sup>	7 <sup>01</sup>	2 <sup>01</sup>	5 <sup>-1</sup>	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6x6



=

-5	-4	0	8
-10	-2	2	3
			0

4x4

# Vertical edge detection examples

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0



\*

1	0	-1
1	0	-1
1	0	-1

=

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0



0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10



\*

1	0	-1
1	0	-1
1	0	-1

=

0	-30	-30	0
0	-30	-30	0
0	-30	-30	0
0	-30	-30	0



# Valid and Same convolutions

→ no padding

“Valid”:

$$n \times n \quad * \quad f \times f \quad \rightarrow \quad \frac{n-f+1}{1} \times n-f+1$$

$$6 \times 6 \quad * \quad 3 \times 3 \quad \rightarrow \quad 4 \times 4$$

“Same”: Pad so that output size is the same as the input size.

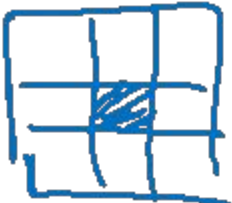
$$n + 2p - f + 1 \times n + 2p - f + 1$$

$$\cancel{n} + 2p - f + 1 = \cancel{n} \Rightarrow p = \frac{f-1}{2}$$

3x3       $p = \frac{3-1}{2} = 1$        $\left| \begin{array}{c} 5 \times 5 \\ f=5 \end{array} \right.$

f is usually odd

1x1  
3x3  
5x5  
7x7



p=2

# Summary of convolutions

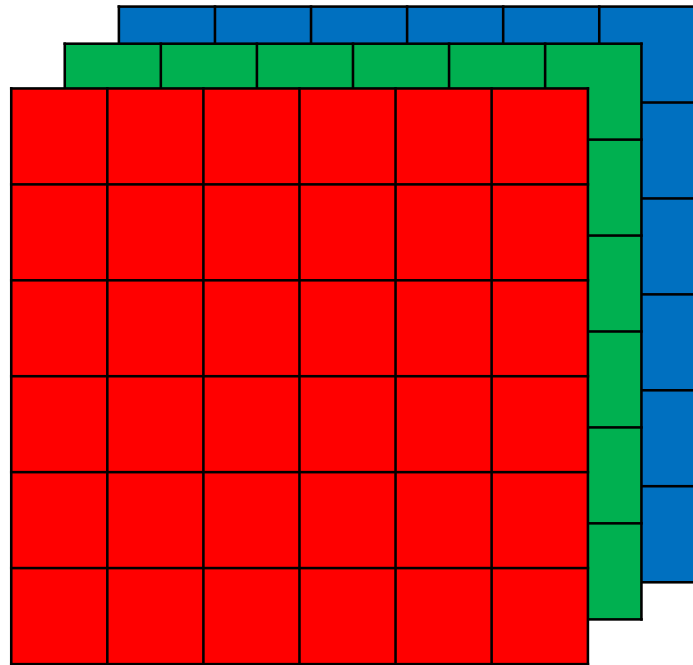
$n \times n$  image       $f \times f$  filter

padding  $p$       stride  $s$

Output Size:

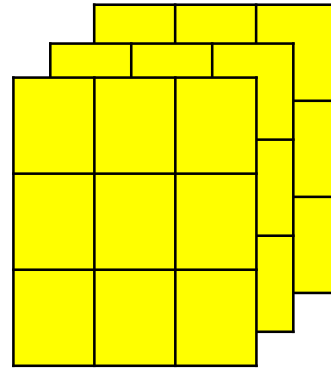
$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

# Multiple filters



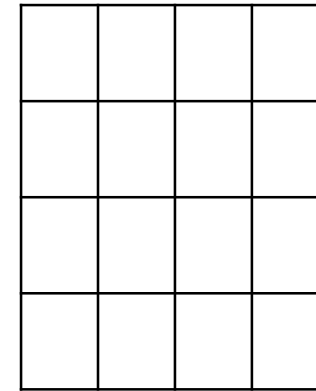
$6 \times 6 \times 3$

\*



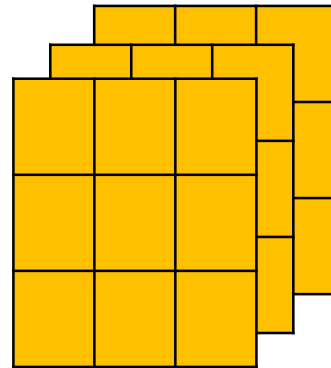
$3 \times 3 \times 3$

=



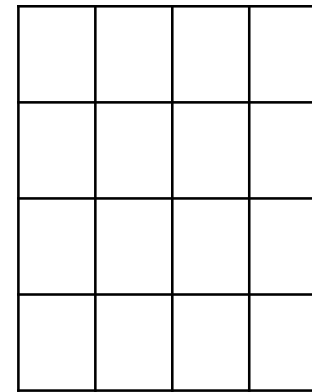
$4 \times 4$

\*



$3 \times 3 \times 3$

=



$4 \times 4$

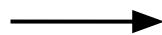


# Pooling layer: Max pooling

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2


# Pooling layer: Average pooling

1	3	2	1
2	9	1	1
1	4	2	3
5	6	1	2




3.75	1.25
4	2

$$f=2$$

$$s=2$$

$$\underline{7 \times 7} \times 1000 \rightarrow 1 \times 1 \times 1000$$

# Types of layer in a convolutional network:

- Convolution (CONV) ←
  - Pooling (POOL) ←
  - Fully connected (FC) ←
- 

# Outline

## Classic

### networks:

- LeNet-5 ←

- AlexNet ←

- VGG ←

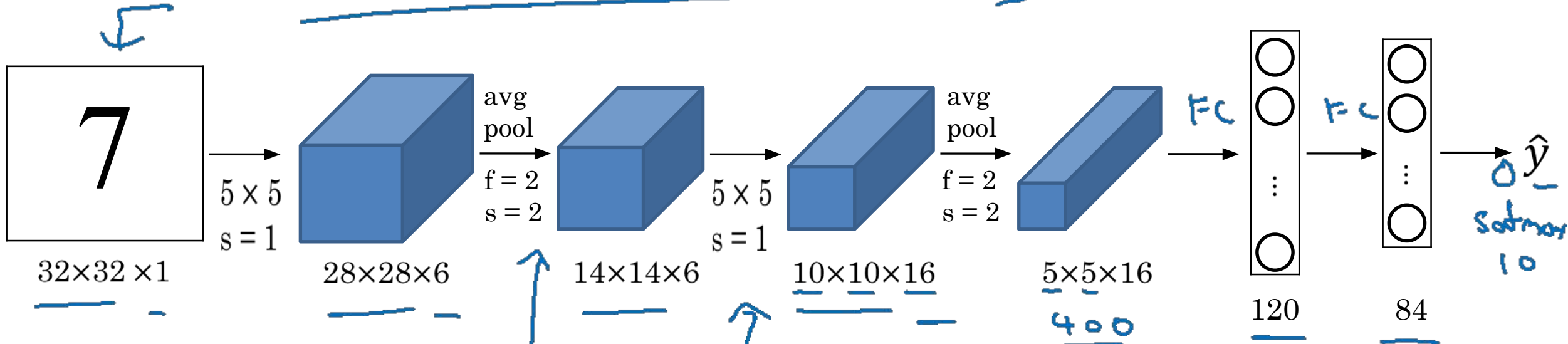
ResNet (152)

t

Inception

n

# LeNet - 5



60K parameters.

$n_H, n_w \downarrow$

$n_c \uparrow$

non-linearly  
ops pooling

$n_H \times n_w \times n_c$

$f \times f \times n_c$

conv pool

conv pool

fc

fc

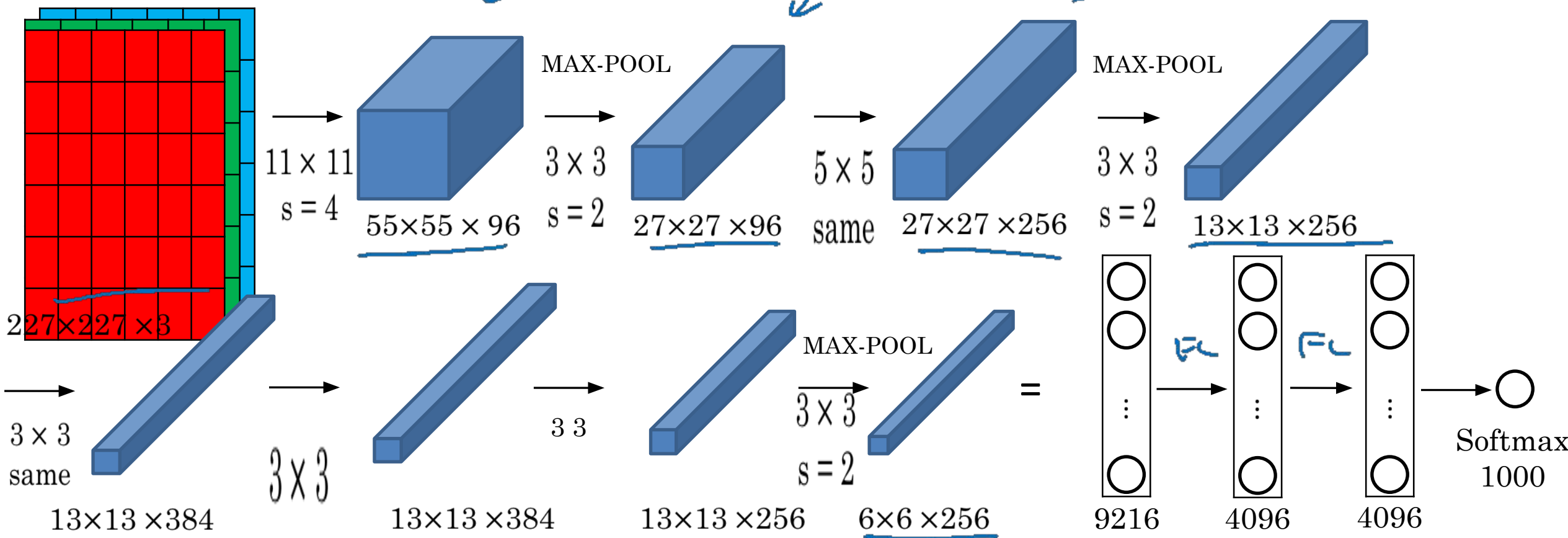
output

Activation: sigmoid/tanh

ReLU



# AlexNet

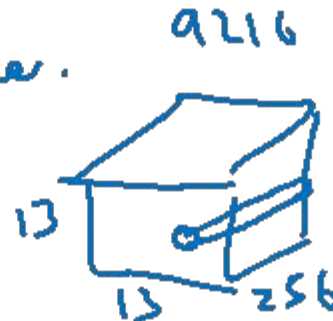


- Similar to LeNet, but much bigger.

- ReLU

- Multiple GPUs.

- Local Response Normalization (LRN)

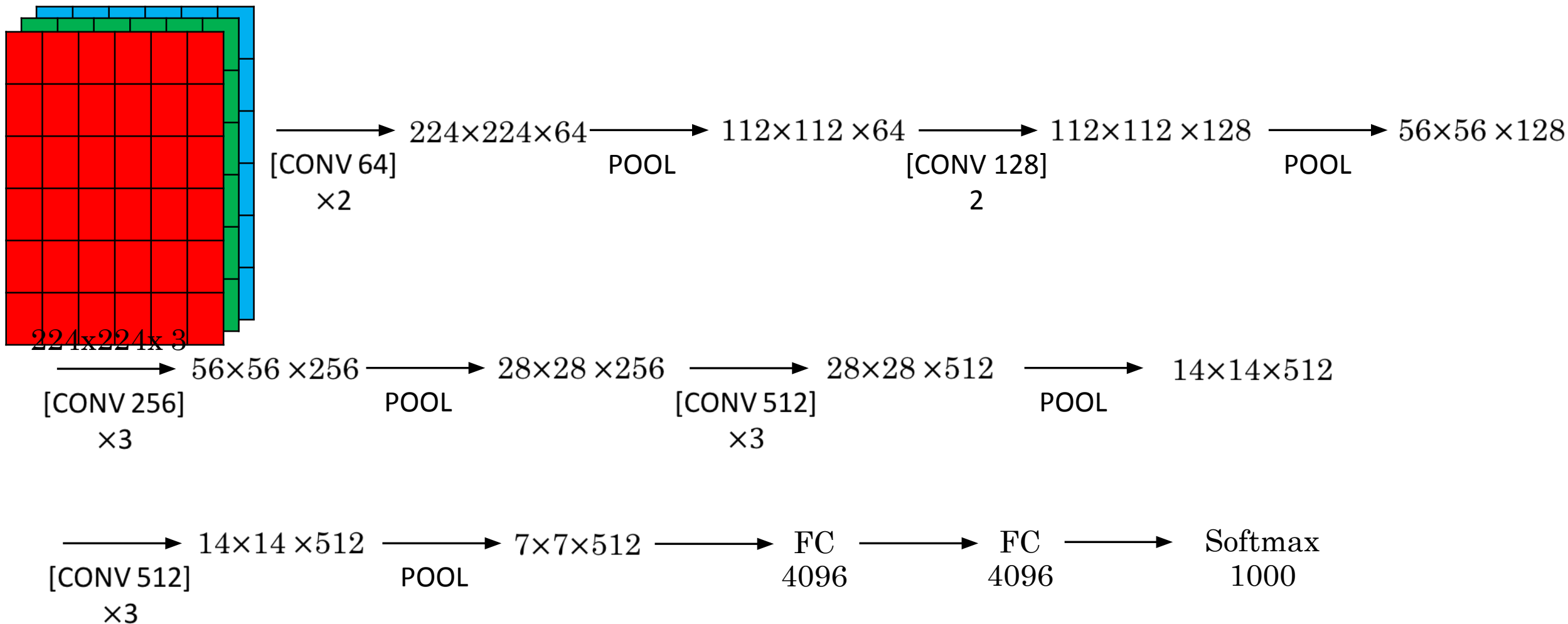


~60M parameters

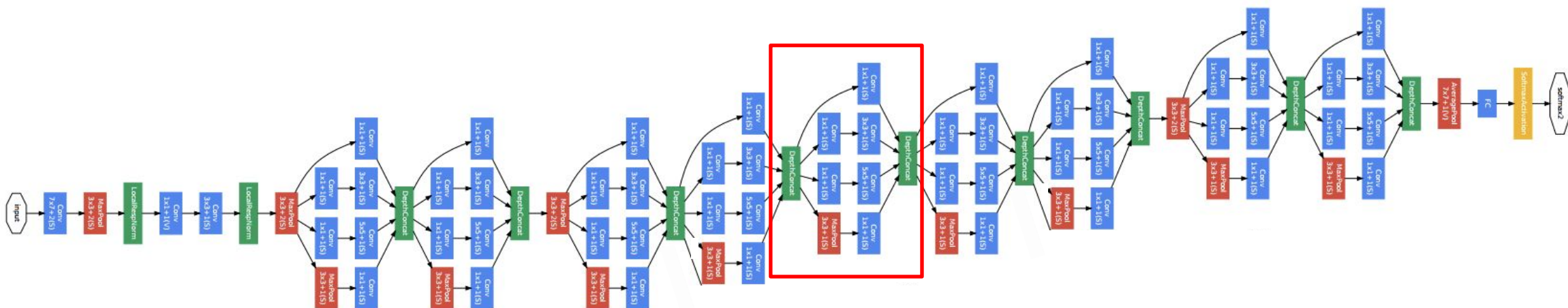
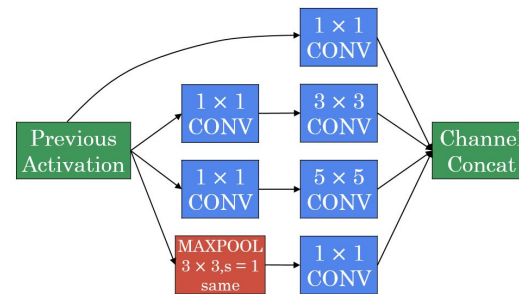
# VGG - 16

CONV = 3x3 filter, s = 1, same

MAX-POOL = 2x2, s = 2



# Inception network



GoogLeNet



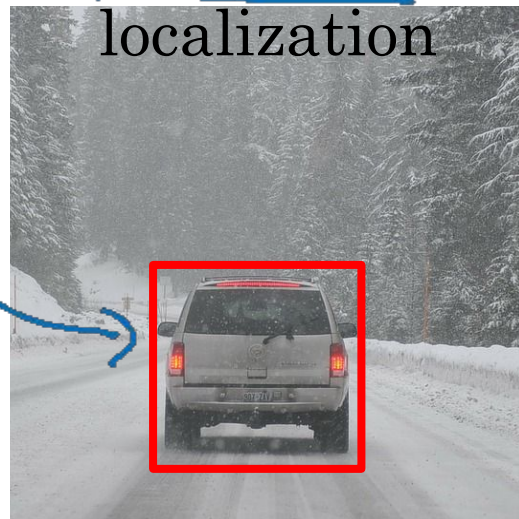
# What are localization and detection?

Image  
classification



"Car"

Classification  
with  
localization



"Car"

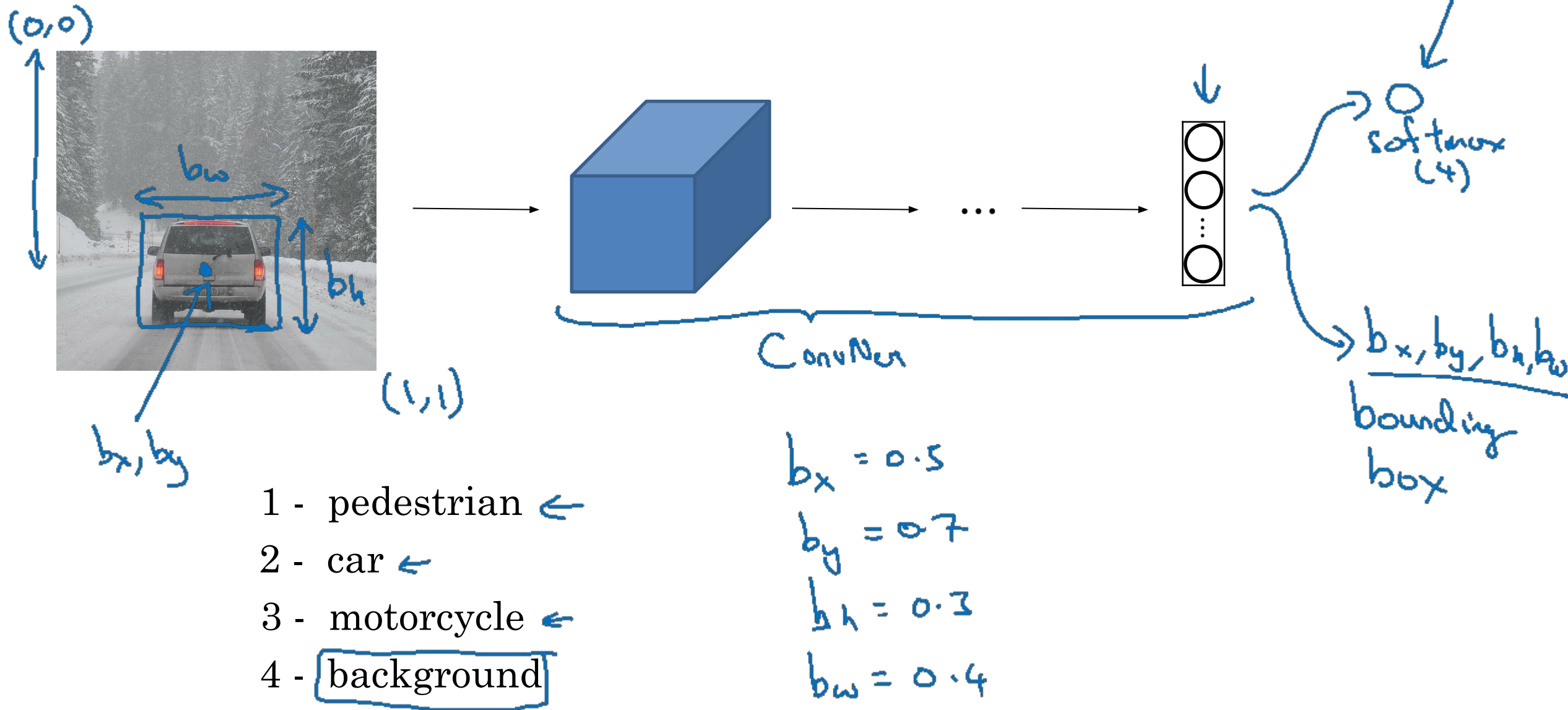
Detection



multiple  
objects

1 object

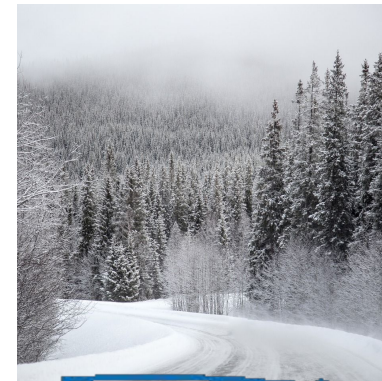
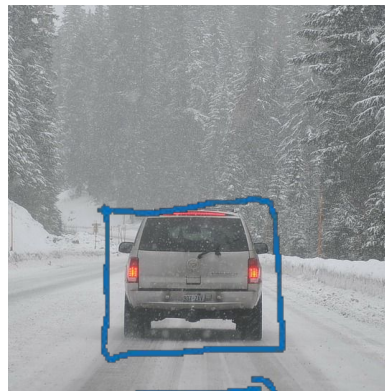
# Classification with localization



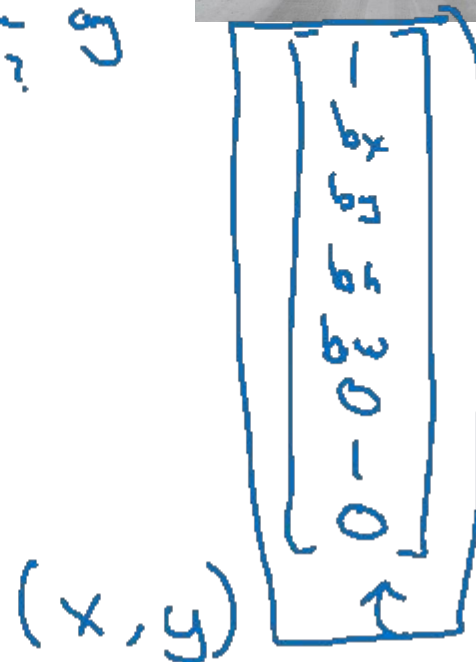
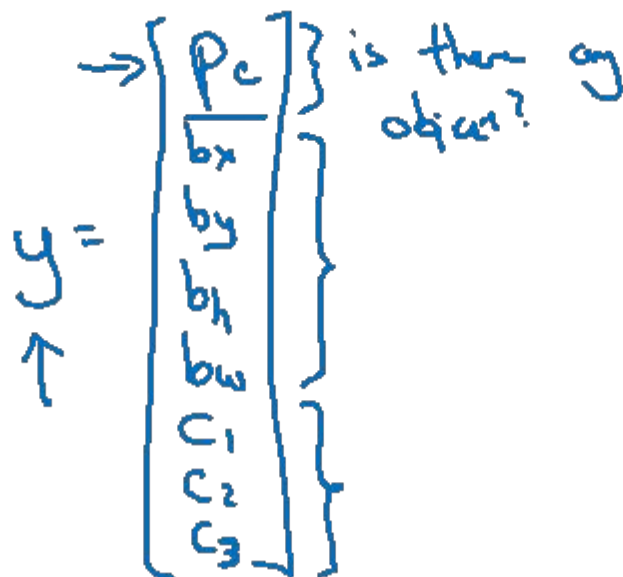
# Defining the target label $y$

- 1 - pedestrian
- 2 - car ←
- 3 - motorcycle
- 4 - background ←

Need to output class, label (1-4)  
 Need to output  $x, y, h, w$  (1-4)

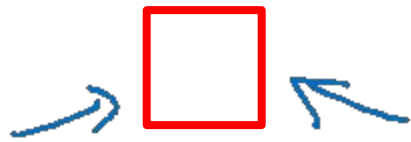
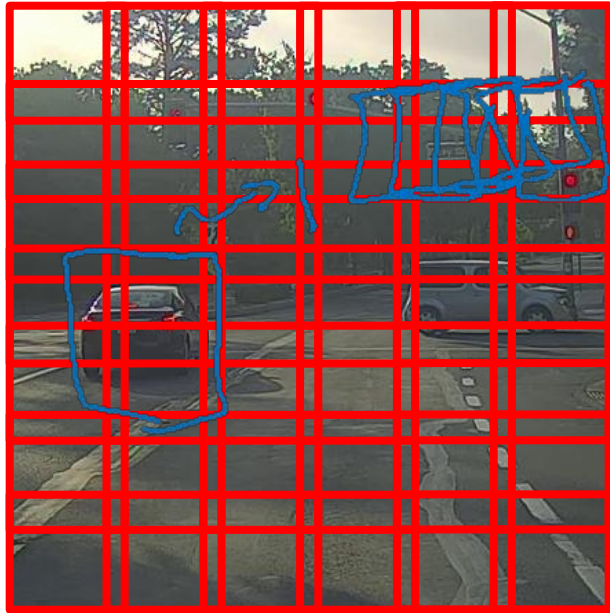


$$L(\hat{y}, y) = \begin{cases} (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + \dots + (\hat{y}_n - y_n)^2 & \text{if } y_1 = 1 \\ (\hat{y}_1 - y_1)^2 & \text{if } y_1 = 0 \end{cases}$$

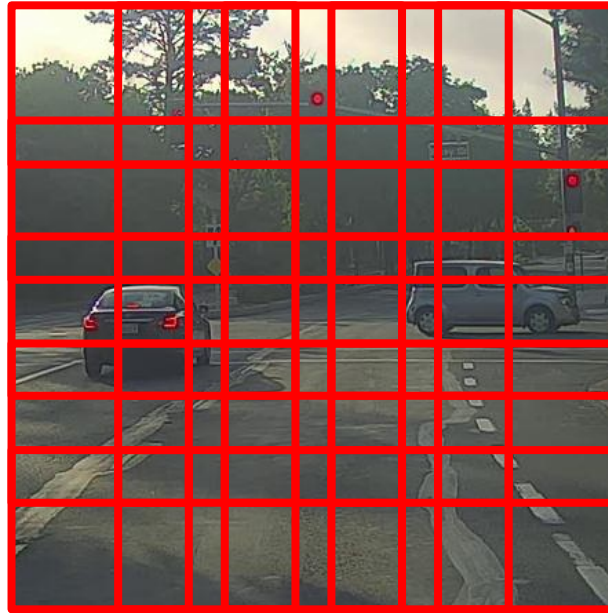


# Sliding windows detection

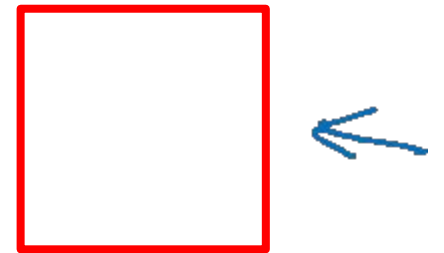
ConvNet → 0



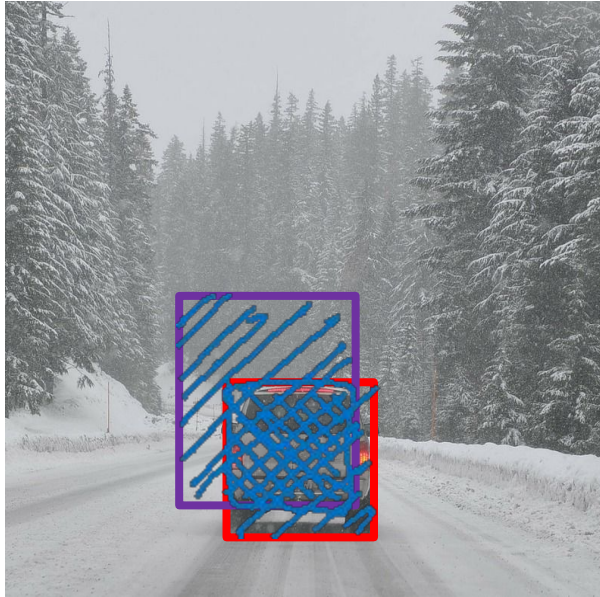
ConvNet



Computation cost



# Evaluating object localization



Intersection over Union (IoU)

$$= \frac{\text{Size of } \text{[shaded box]}}{\text{Size of } \text{[union box]}}$$

“Correct” if IoU 0.5

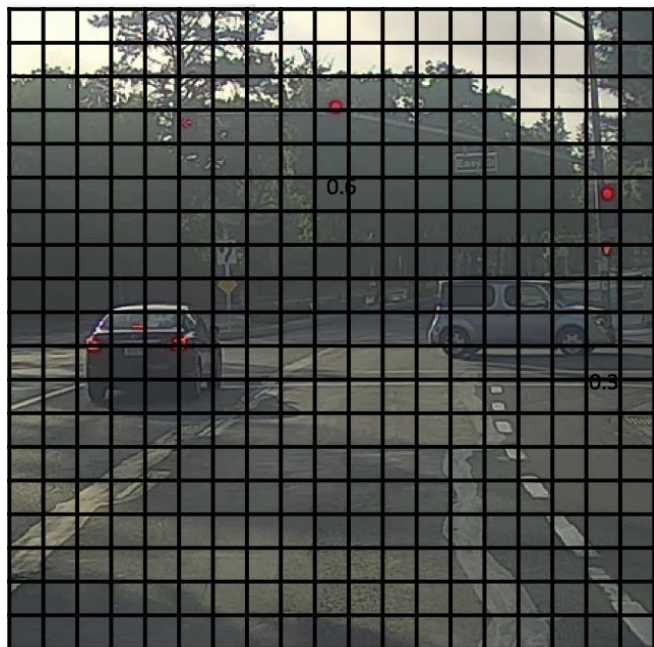
0.6

More generally, IoU is a measure of the overlap between two bounding boxes.

# Non-max suppression example

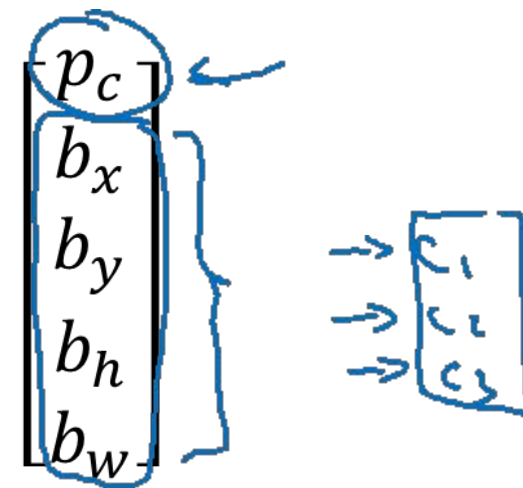


# Non-max suppression algorithm



19x19

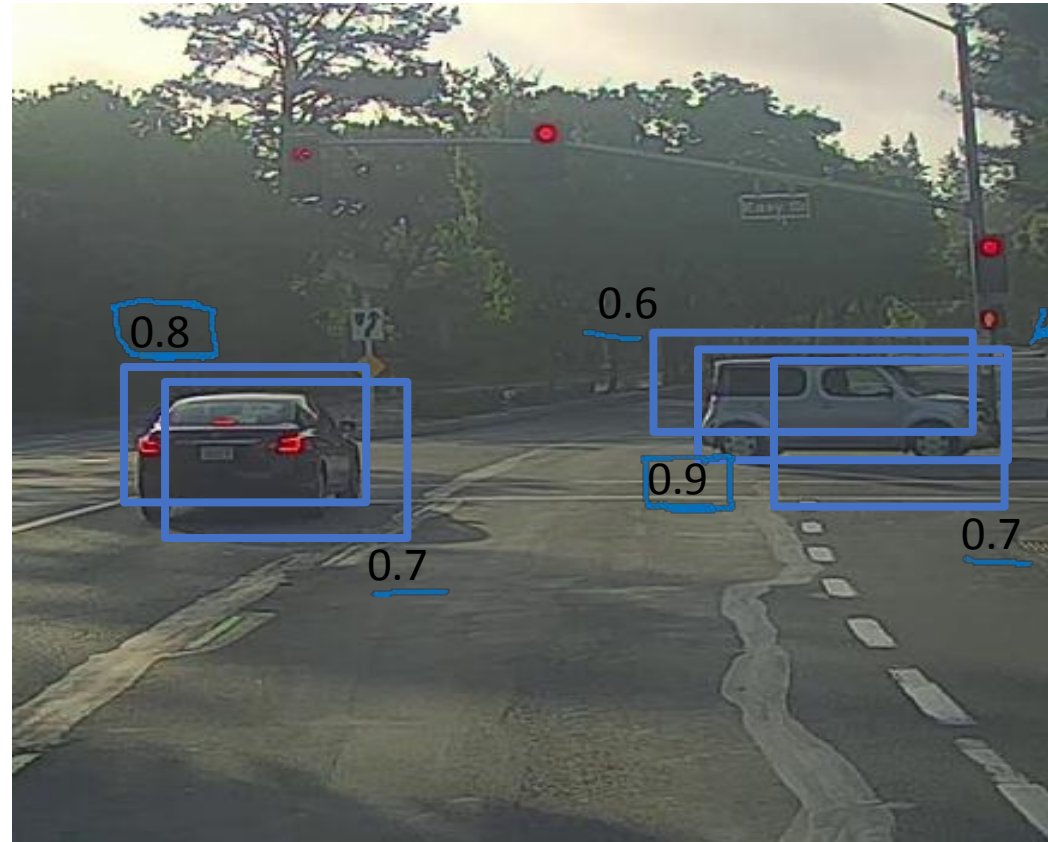
Each output prediction is:



Discard all boxes with                     

- While there are any remaining boxes:
- Pick the box with the largest Output that as a prediction.
  - Discard any remaining box with IoU with the box output in the previous step

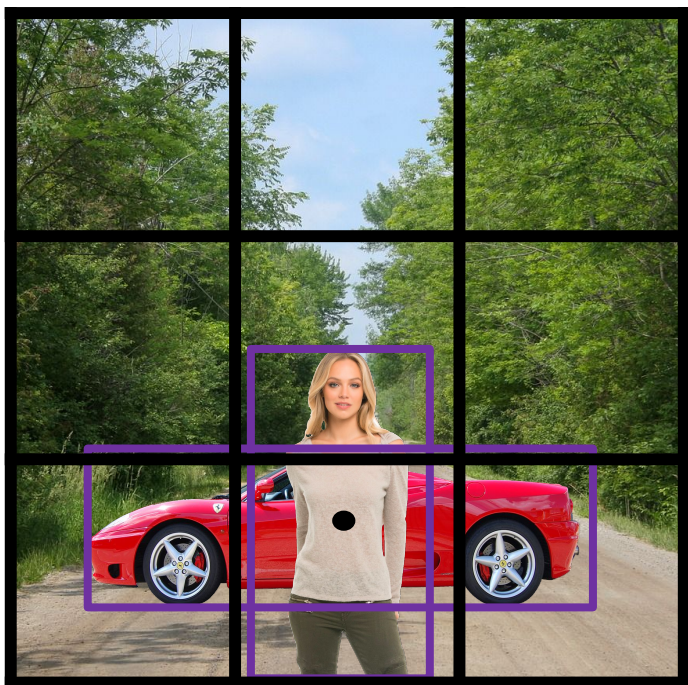
# Non-max suppression example



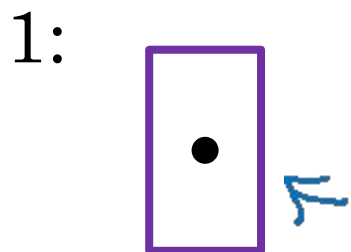
$P_c$



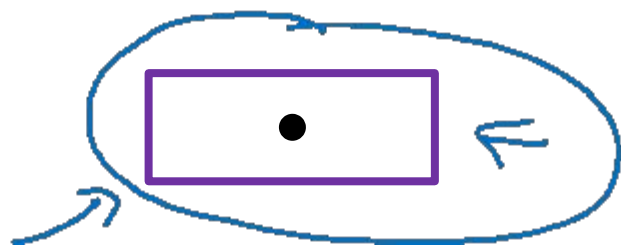
# Anchor box example



Anchor box 1:



Anchor box 2:



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

