Detection Overview

Keyword

- Edge

- Classification,
  Classification with localization,
  Detection

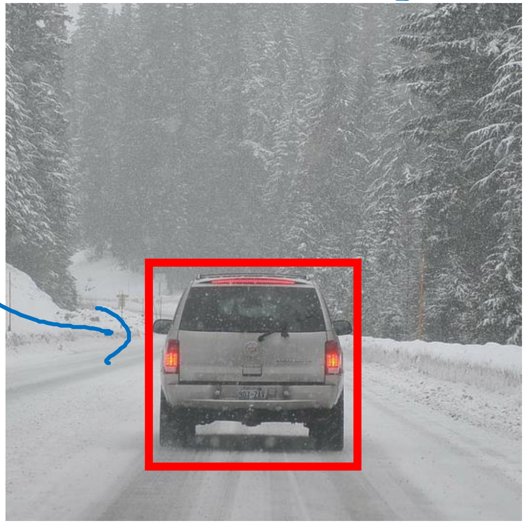
- Sliding window,
  Anchor,
  Intersection of union ,
  Non-maximum suppression

What are localization and detection?

Image classification

"Car"

Classification with localization



"Car"

1 object

Detection

multiple objects

Classification

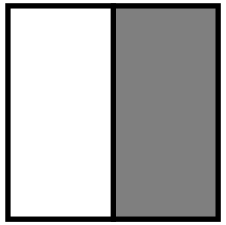
- Problem: classify the objects
- How do the people recognize the objects?
- **Edge**, color …
- How do the computers recognize the objects using edges?

Edge definition

- Discontinuity in image brightness or contrast
- Usually, edges occur on the boundary of two regions
- Abrupt changes in the intensity of pixels
- How does the computer recognize edges?

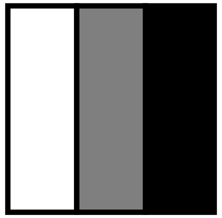
Vertical edge detection

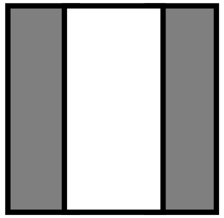
$$
\begin{bmatrix}
10 & 10 & 10 & 0 & 0 & 0 \\
10 & 10 & 10 & 0 & 0 & 0 \\
10 & 10 & 10 & 0 & 0 & 0 \\
10 & 10 & 10 & 0 & 0 & 0 \\
10 & 10 & 10 & 0 & 0 & 0 \\
10 & 10 & 10 & 0 & 0 & 0
\end{bmatrix}
*
\begin{bmatrix}
1 & 0 & -1 \\
1 & 0 & -1 \\
1 & 0 & -1
\end{bmatrix}
=
\begin{bmatrix}
0 & 30 & 30 & 0 \\
0 & 30 & 30 & 0 \\
0 & 30 & 30 & 0 \\
0 & 30 & 30 & 0
\end{bmatrix}
$$



$*$

Convolution (cross-correlation)



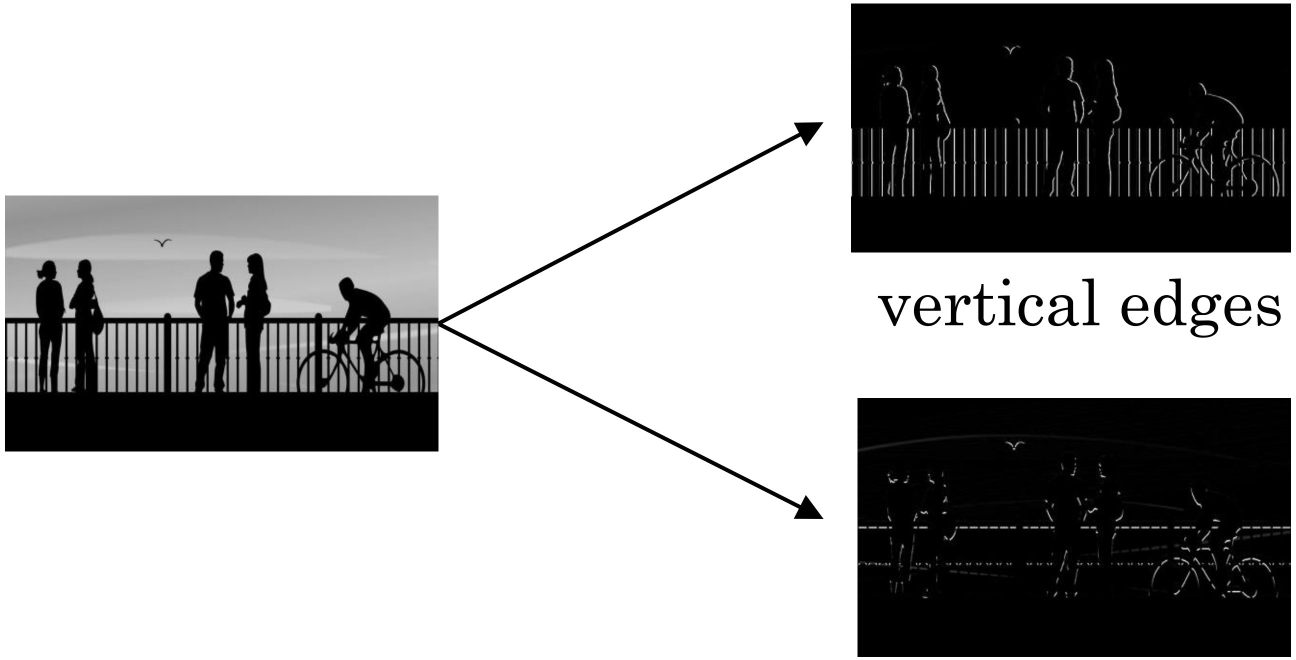


Edge Detection

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

Horizontal

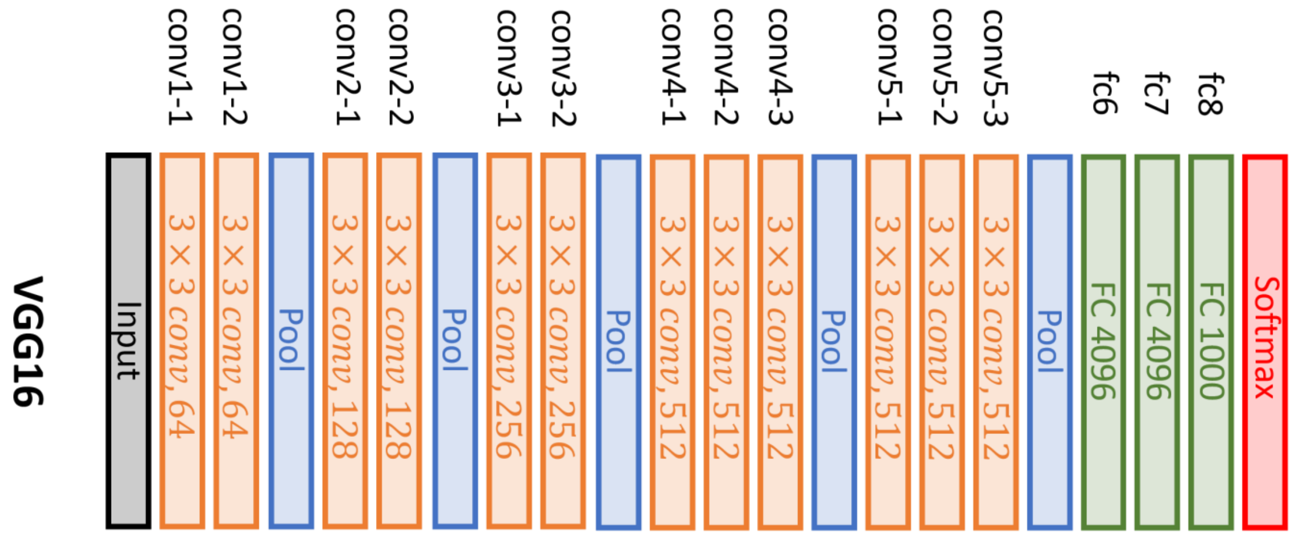
Computer Vision Problem



vertical edges

horizontal edges

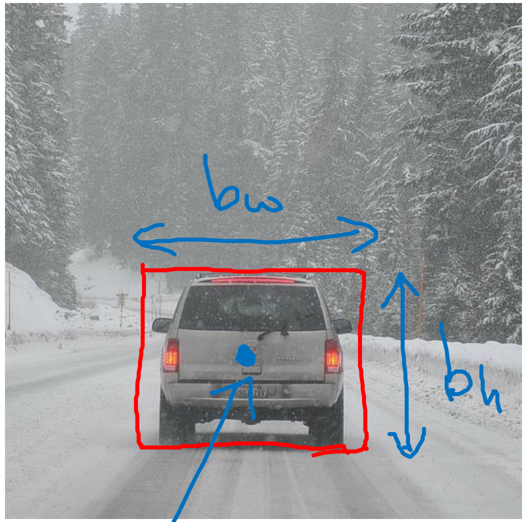
CNN visualization



Gradient descent can optimize the weights in the convolution filters

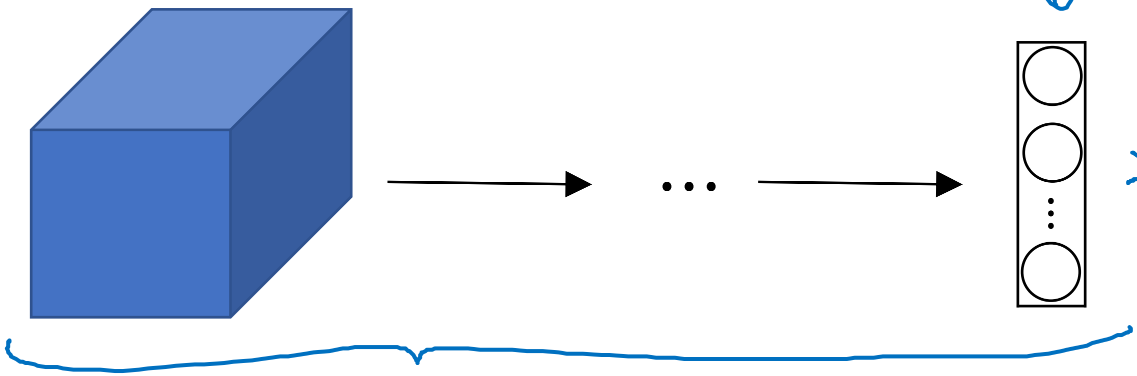
Classification with localization

(0,0)



(1,1)

b_x, b_y



softmax (4)

b_x, b_y, b_h, b_w

bounding box

1 - pedestrian

2 - car

3 - horse

4 - $\boxed{\text{background}}$

$b_x = 0.5$

$b_y = 0.7$

$b_h = 0.3$

$b_w = 0.4$

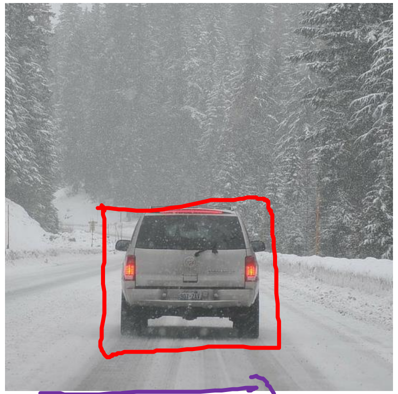
Classifier + Regressor

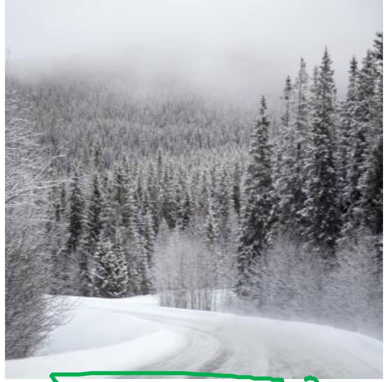
Defining the target label y

1 - pedestrian
2 - car
3 - horse
4 - background

Need to output b_x, b_y, b_h, b_w, class label (1-4)

$x =$





$$\mathcal{L}(\hat{y}, y) = \begin{cases} (\hat{y}_1 - y_1)^2 + (\hat{y}_2 - y_2)^2 + \dots + (\hat{y}_8 - y_8)^2 & \text{if } y_1 = 1 \\ (\hat{y}_1 - y_1)^2 & \text{if } y_1 = 0 \end{cases}$$

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

is there any object?

$$\begin{bmatrix} 1 \\ b_x \\ b_y \\ b_h \\ b_w \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad \begin{bmatrix} 0 \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \\ ? \end{bmatrix}$$

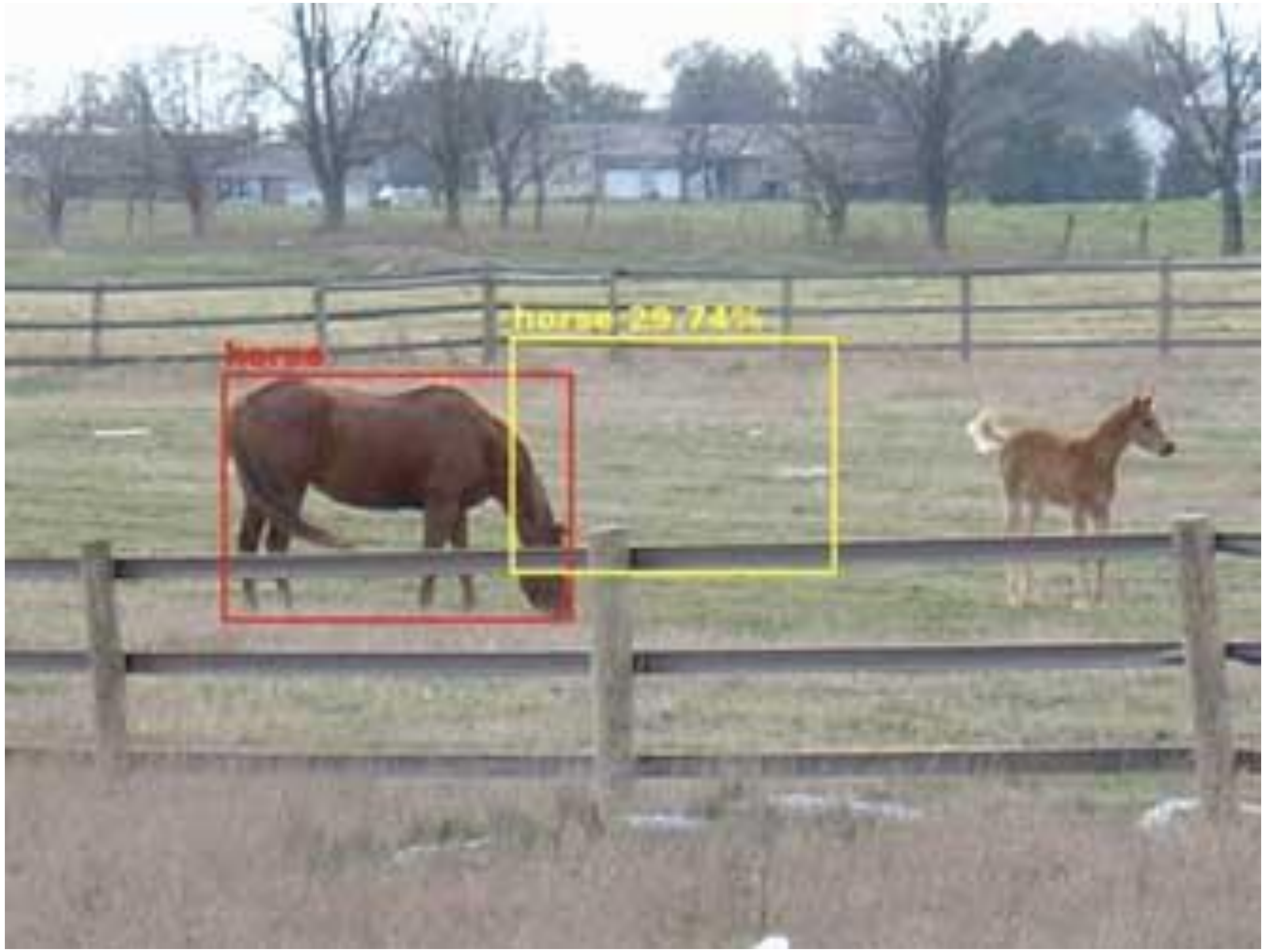
(x, y)

p_c

"don't care"

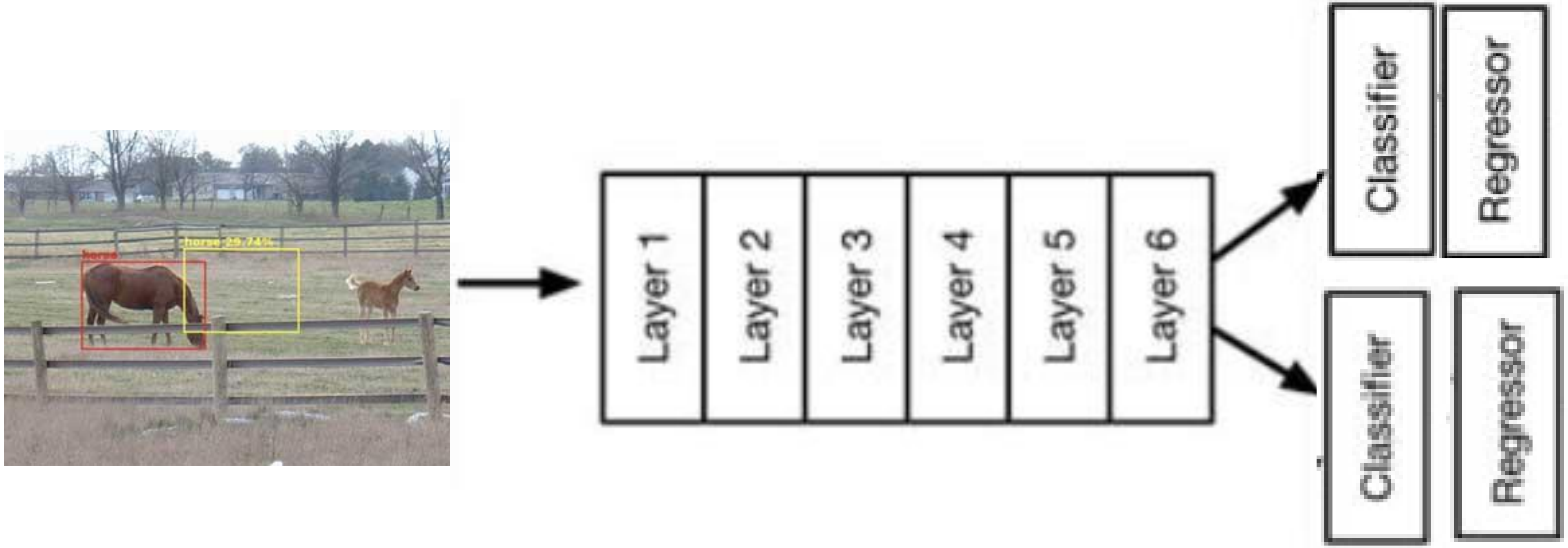
Andrew Ng

Detection Problem



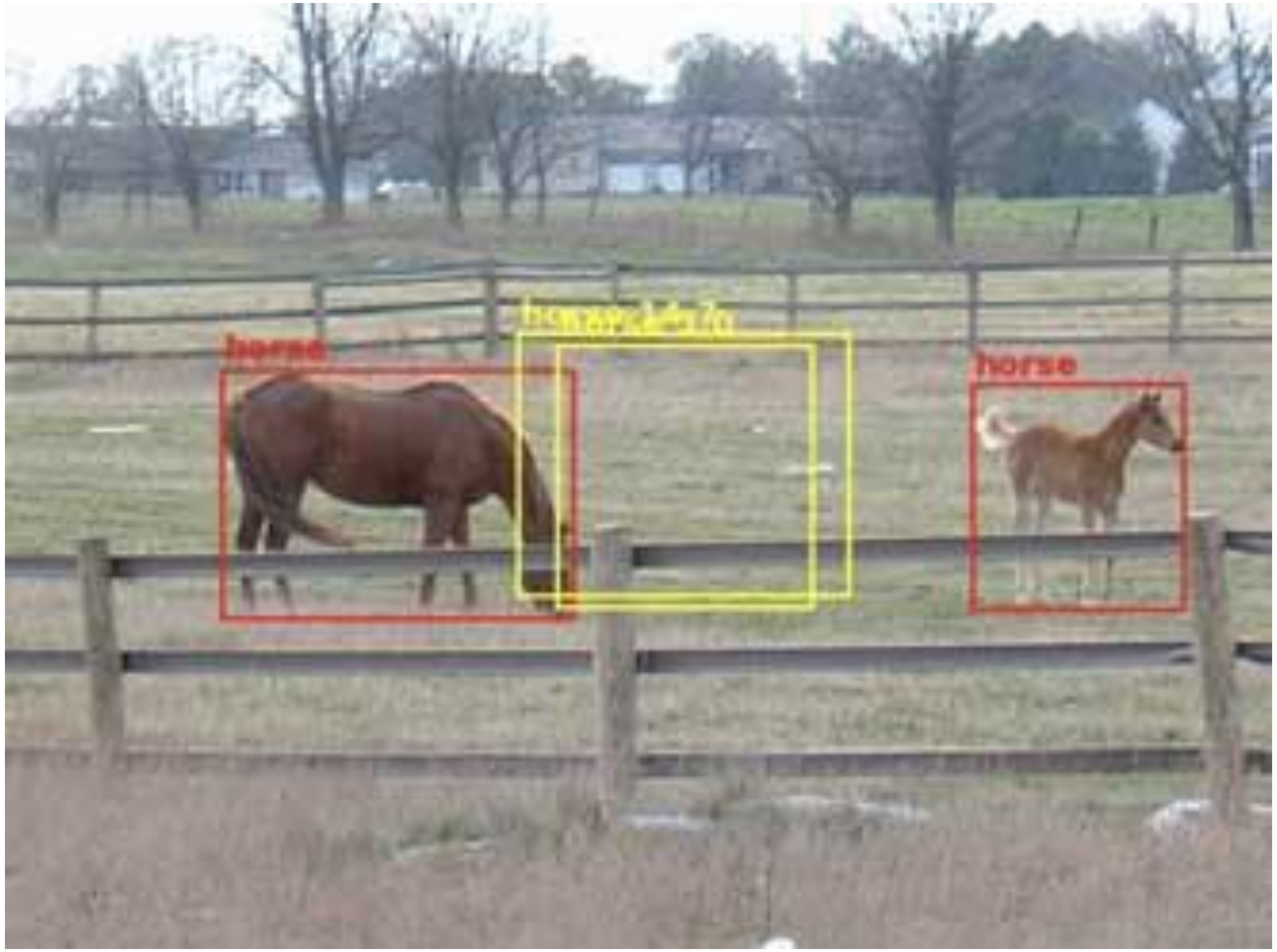
- Problems arise when using images with multiple objects.
- Alternative plan: add more classifier & regressor for multiple objects

Detection Problem



Let’s add more classifier & regressor for multiple objects!

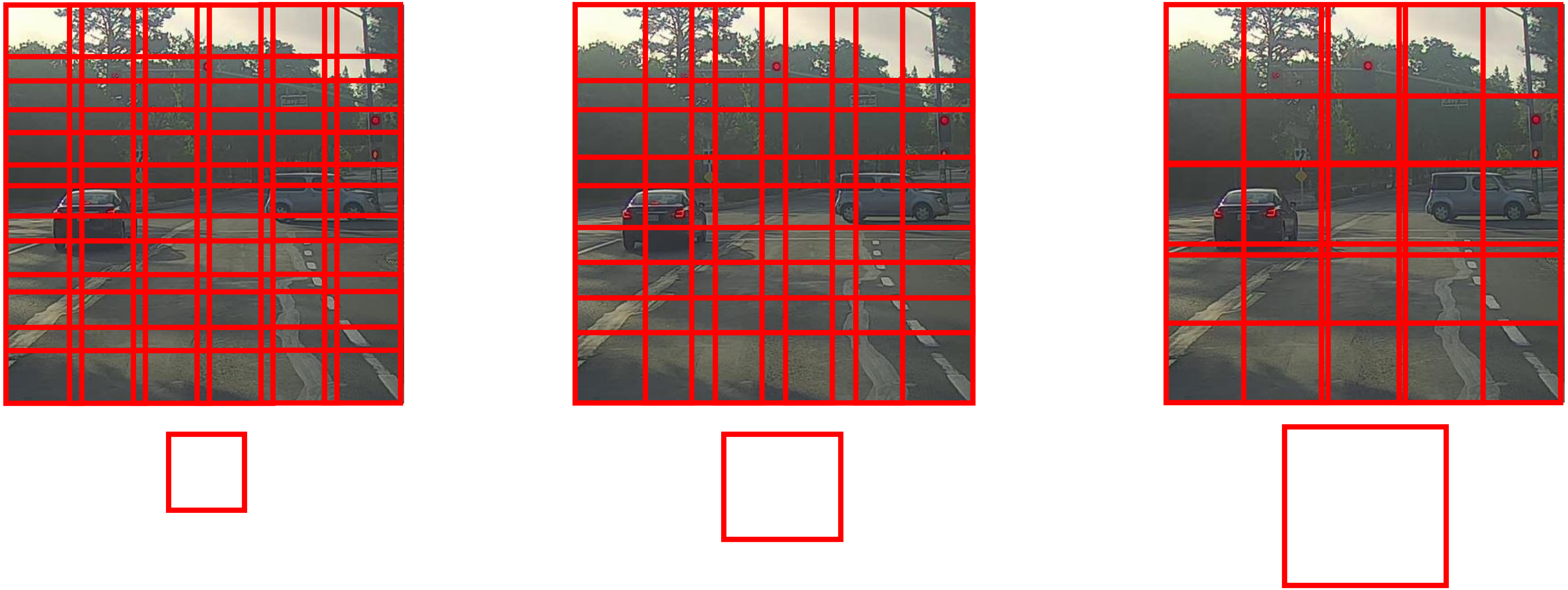
Detection Problem



- it seems not to work properly

- Solution:
  Let's use sliding window

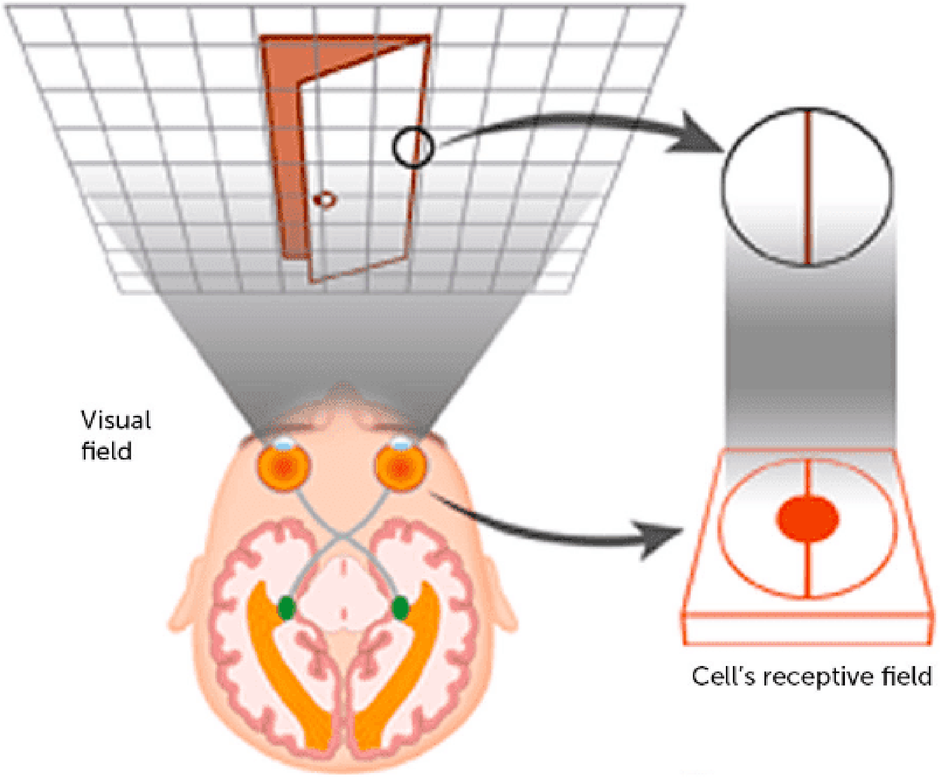
Sliding windows detection : Naïve version

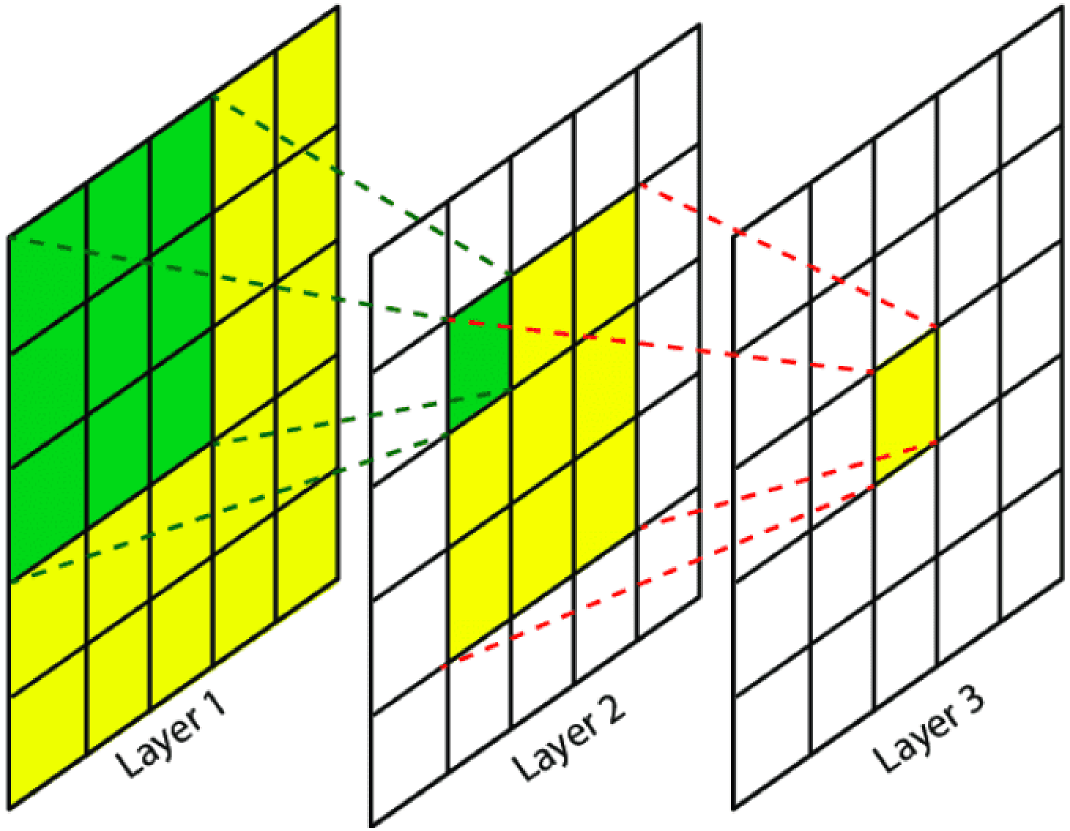


Too slow for inferring each window!

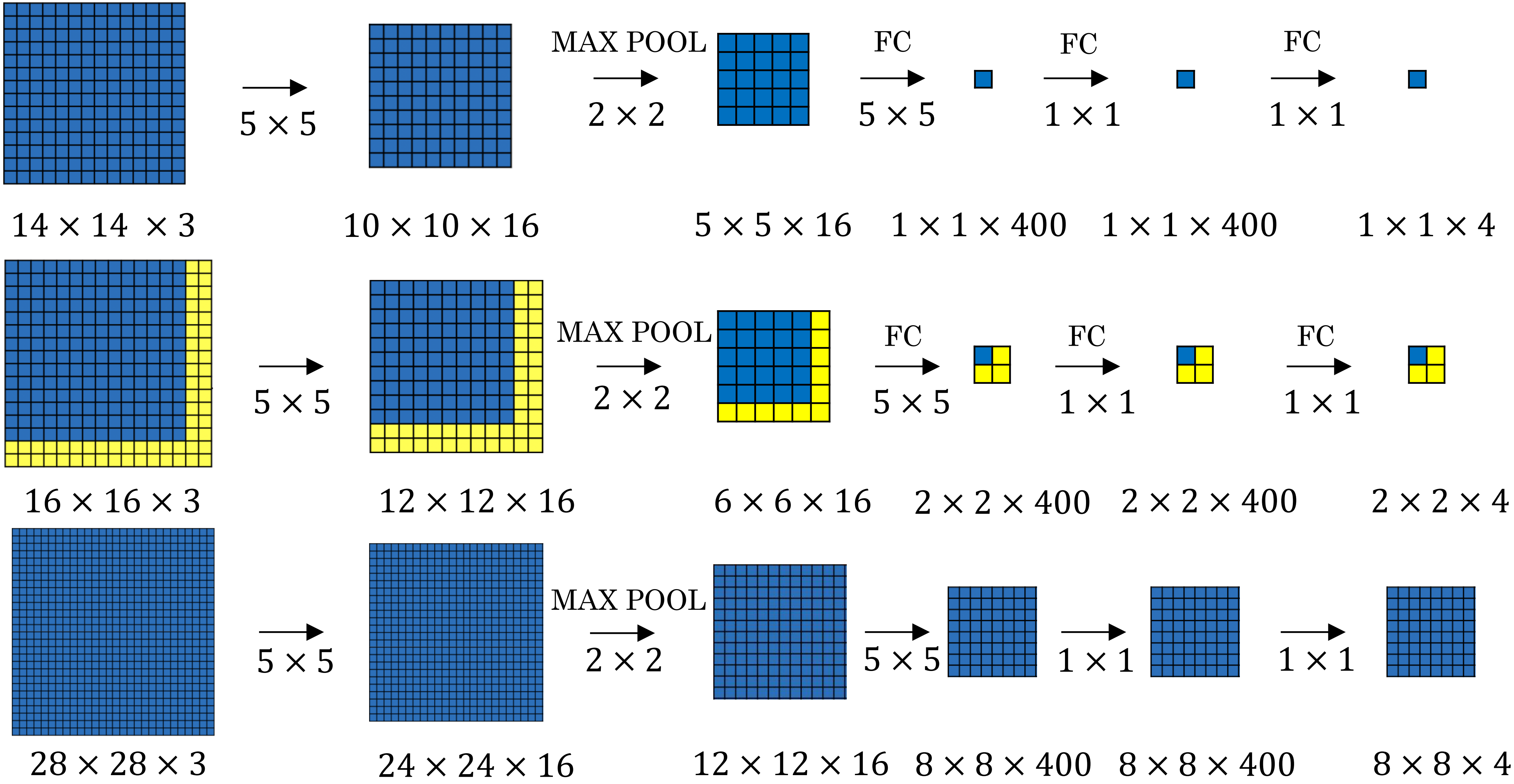
Andrew Ng

Receptive field



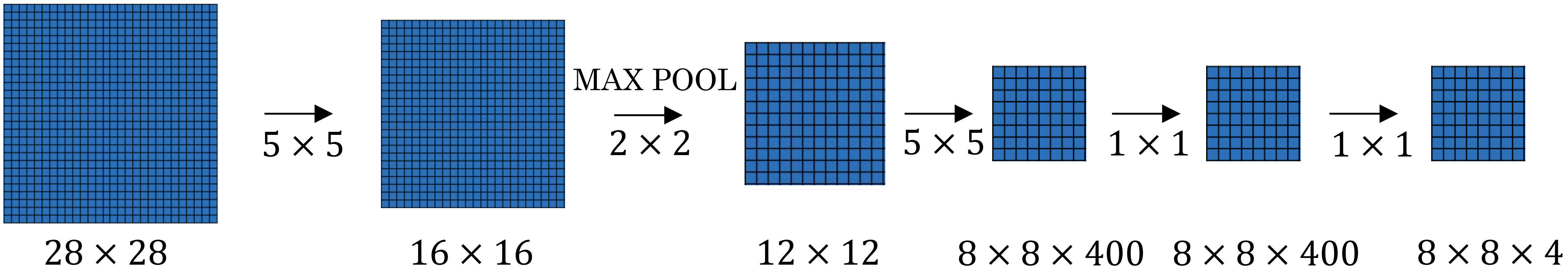


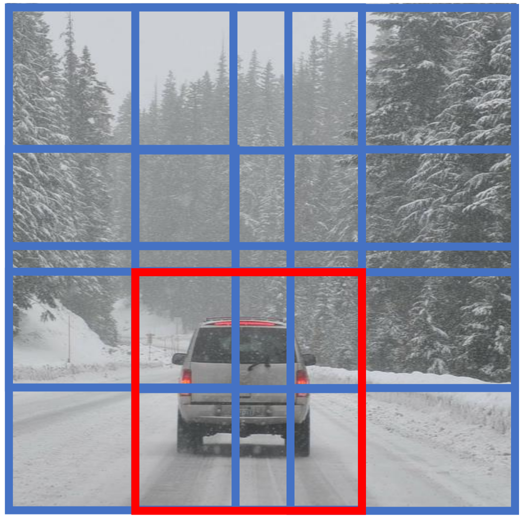
Convolution implementation of sliding windows

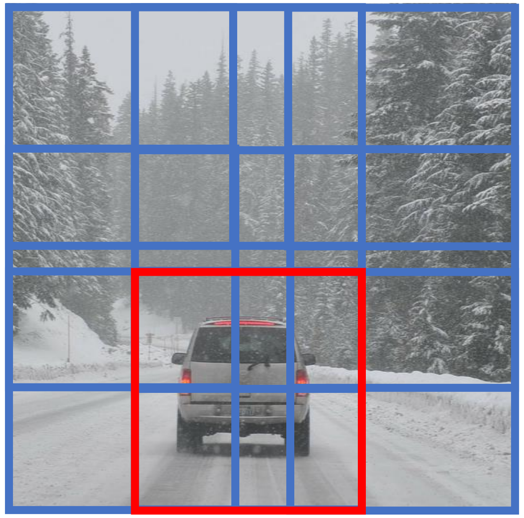


[Sermanet et al., 2014, OverFeat: Integrated recognition, localization and detection using convolutional networks]

Convolution implementation of sliding windows

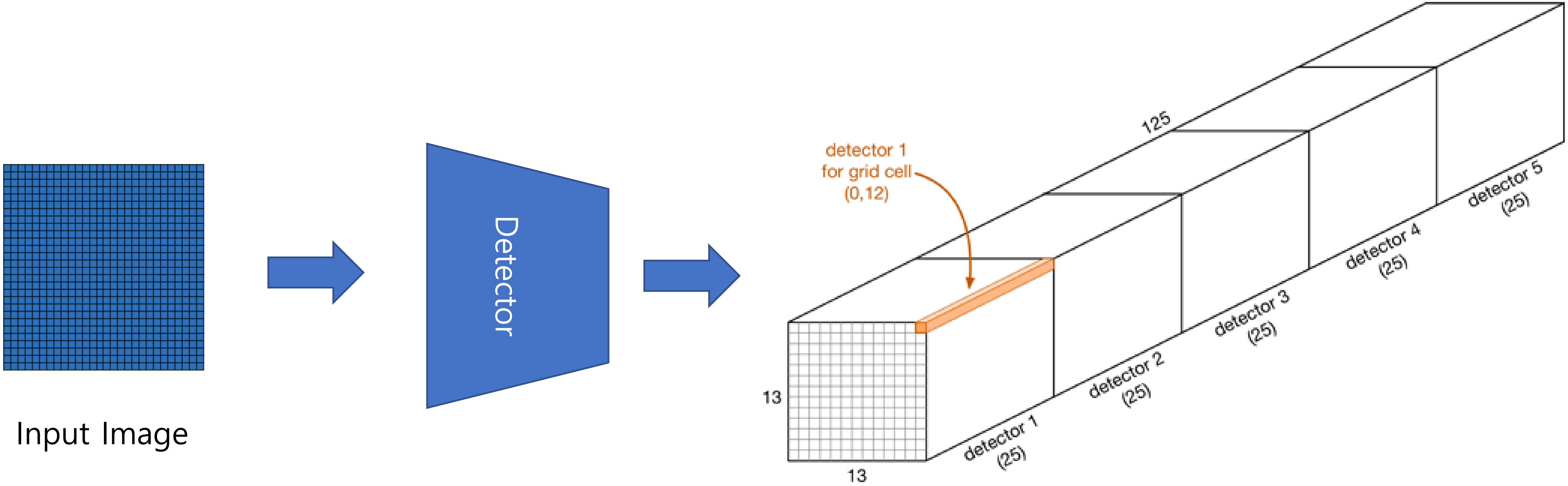






Andrew Ng

Improved Model



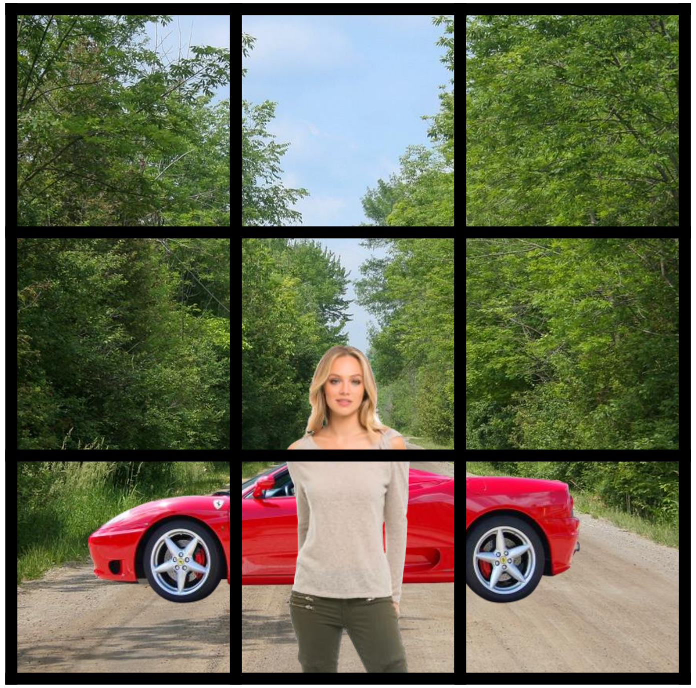
Input Image

- 4 bounding box coordinates (center x, center y, width, height)
- 1 confidence score
- 20 numbers containing the class probabilities

Loss function review

- when the confidence score is too low
- when the coordinates are wrong
- when the class is wrong
- we want to penalize it

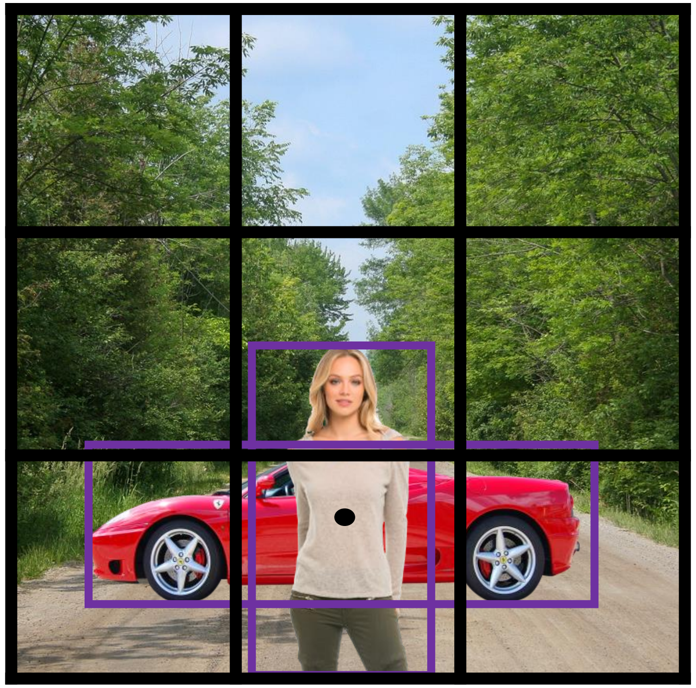
Overlapping objects



- Problem:
  If there are objects that have center point close to each other, does the model work well?

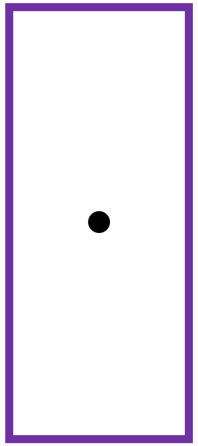
- Solution:
  Anchor Box (predefined bounding box)

Overlapping objects:

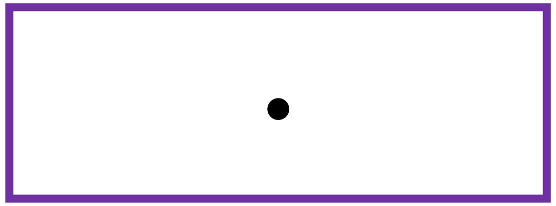


$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Anchor box 1:



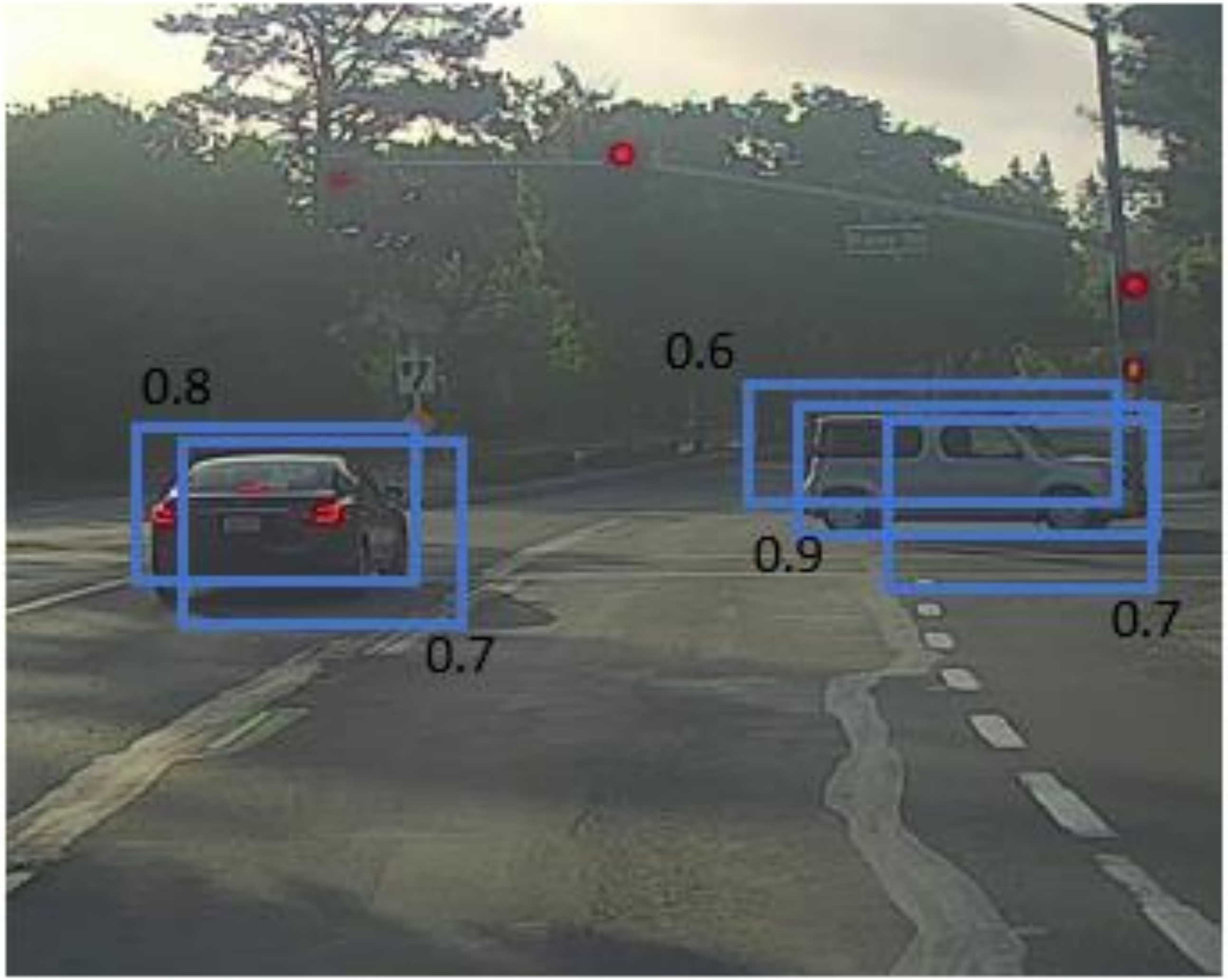
Anchor box 2:



$$y = \begin{bmatrix} \left.\begin{matrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{matrix}\right\} \text{anchor box 1} \\ \left.\begin{matrix} p_c \\ b_x \\ \vdots \\ c_3 \end{matrix}\right\} \text{anchor box 2} \end{bmatrix}$$

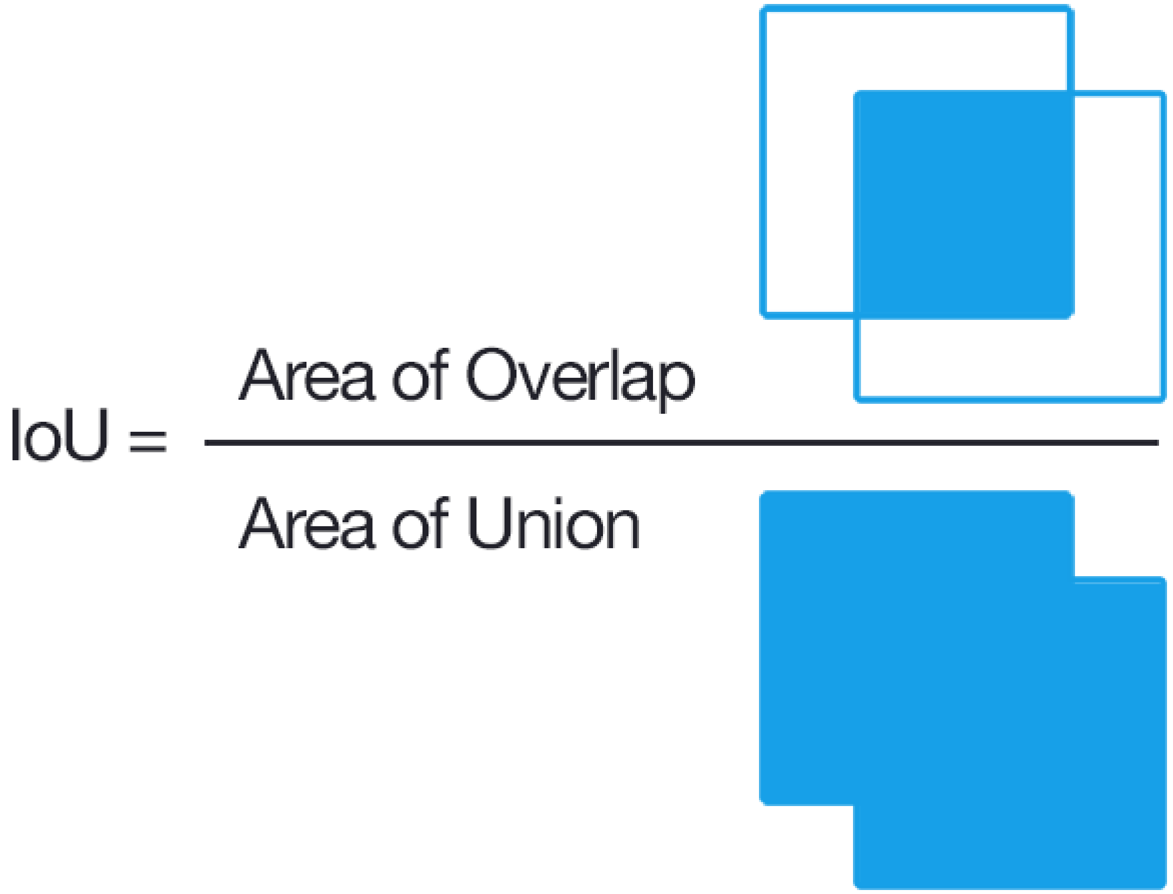
[Redmon et al., 2015, You Only Look Once: Unified real-time object detection]

Visualization Problem

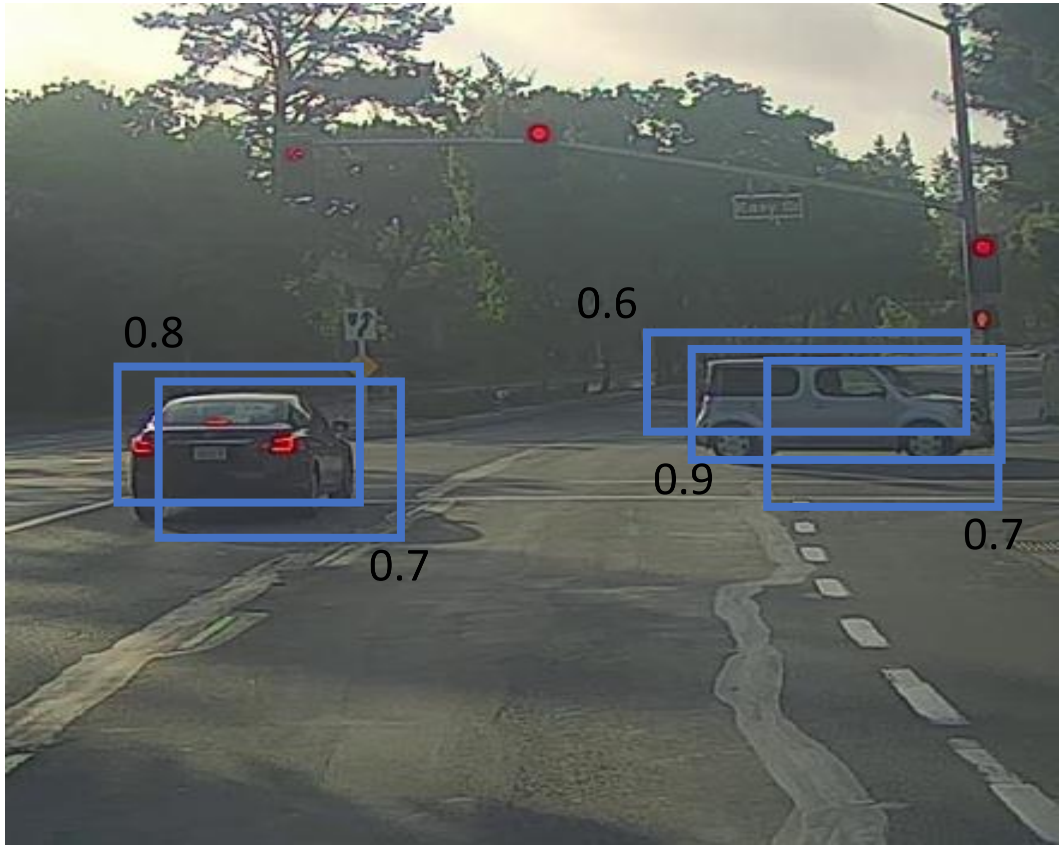


- Problem:
  Several boxes on single object
- Solution:
  Non-max suppression
- First, we need to know intersection of union

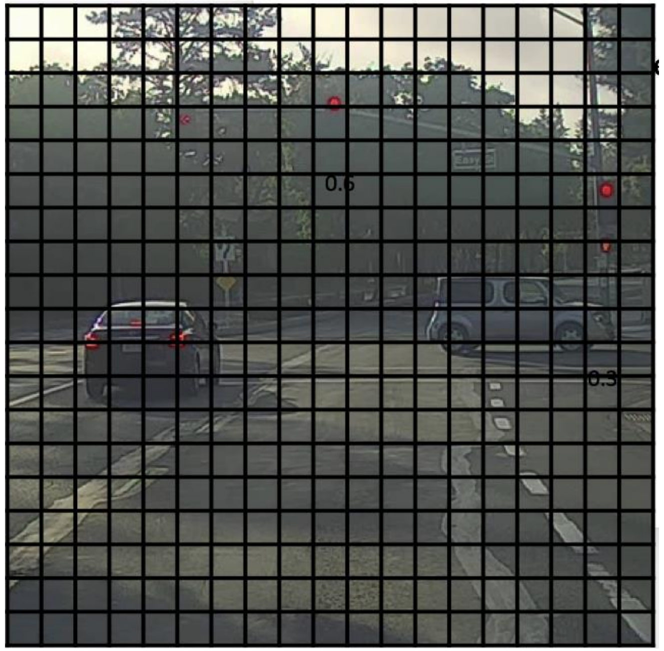
Intersection of Union



Non-max suppression example



Non-max suppression algorithm



19× 19

Each output prediction is: $\begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \end{bmatrix}$

Discard all boxes with $p_c \leq 0.6$

While there are any remaining boxes:

- Pick the box with the largest p_c
  Output that as a prediction.
- Discard any remaining box with
  IoU $\geq$ 0.5 with the box output
  in the previous step

Andrew Ng

Reference

- Deep Learning | Coursera
- One-stage object detection (machinethink.net)
- Understanding the receptive field of deep convolutional networks | AI Summer (theaisummer.com)
- Intersection over Union (IoU) for object detection - PyImageSearch