Object Detection using Deep Learning

Dr. Rakesh Kumar Sanodiya

Indian Institute of Information Technology Sri City
rakesh.pcs16@gmail.com

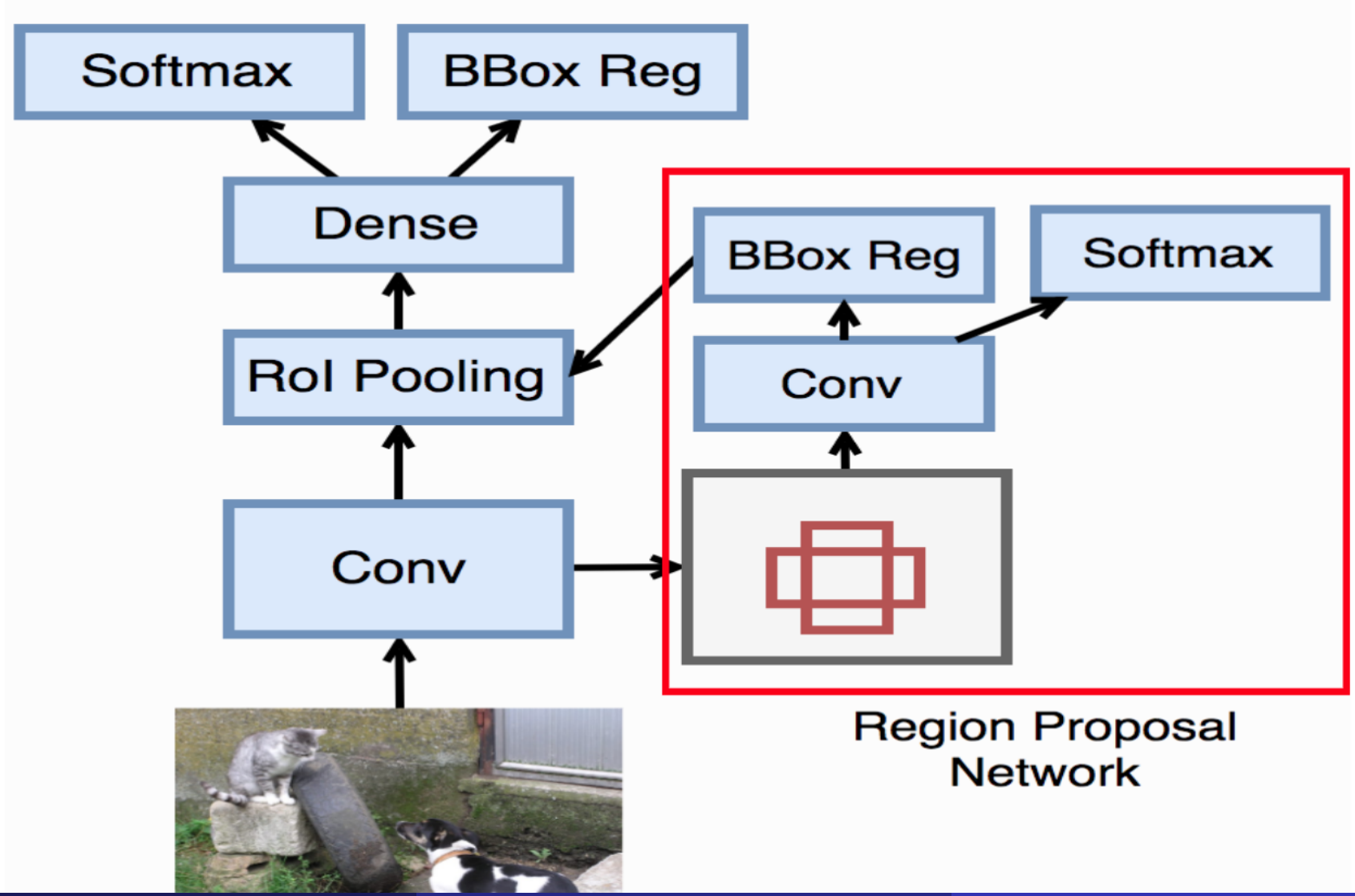
CSE Department, IIIT Sri City
April 19, 2024

GAN:What, why, and how

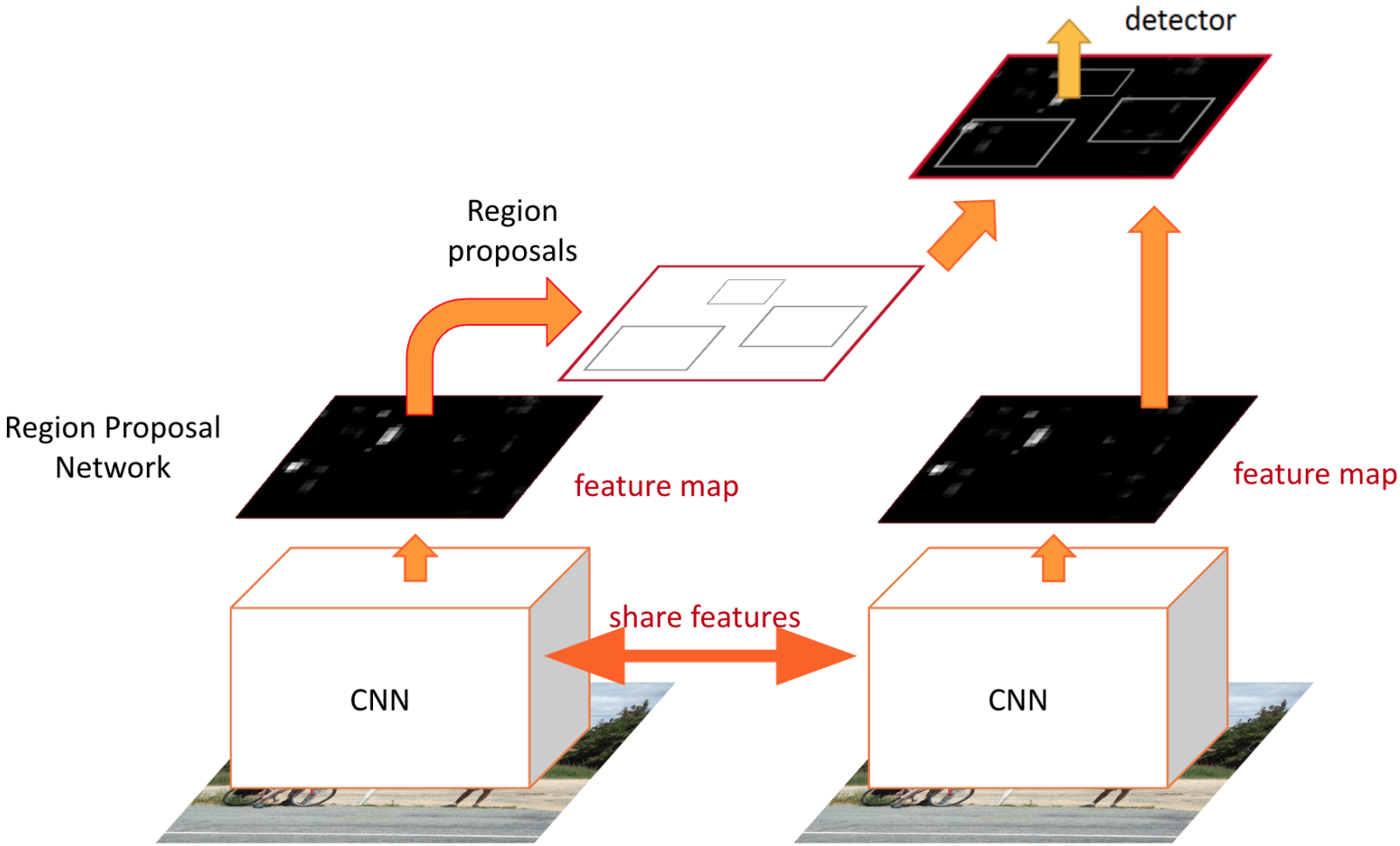
1. In the last class

Faster R-CNN (Ren et al. NIPS 2015

- Solution
  - Why not generate region proposals using CNN??

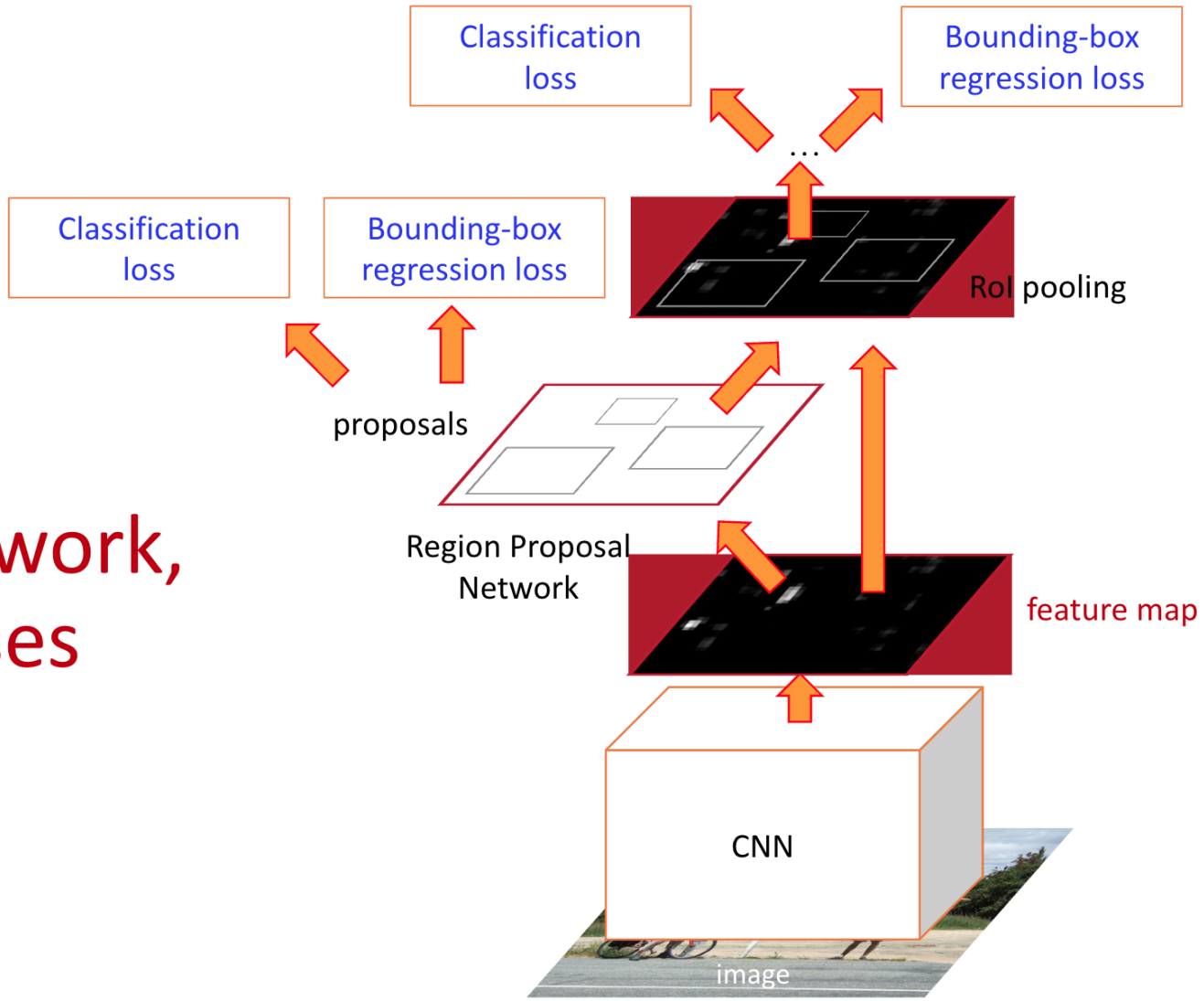


Faster R-CNN (Ren et al. NIPS 2015



Faster R-CNN (Ren et al. NIPS 2015

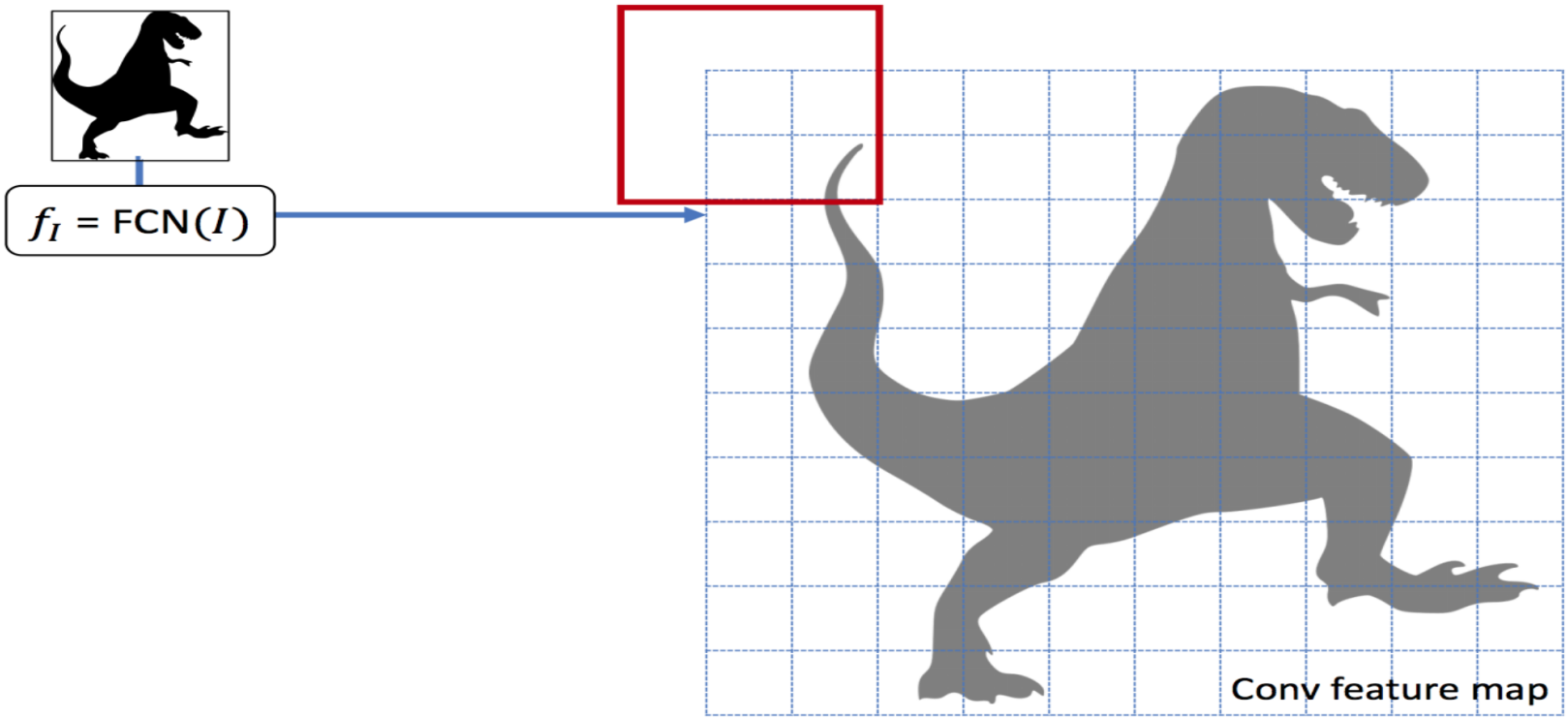
One network,
four losses



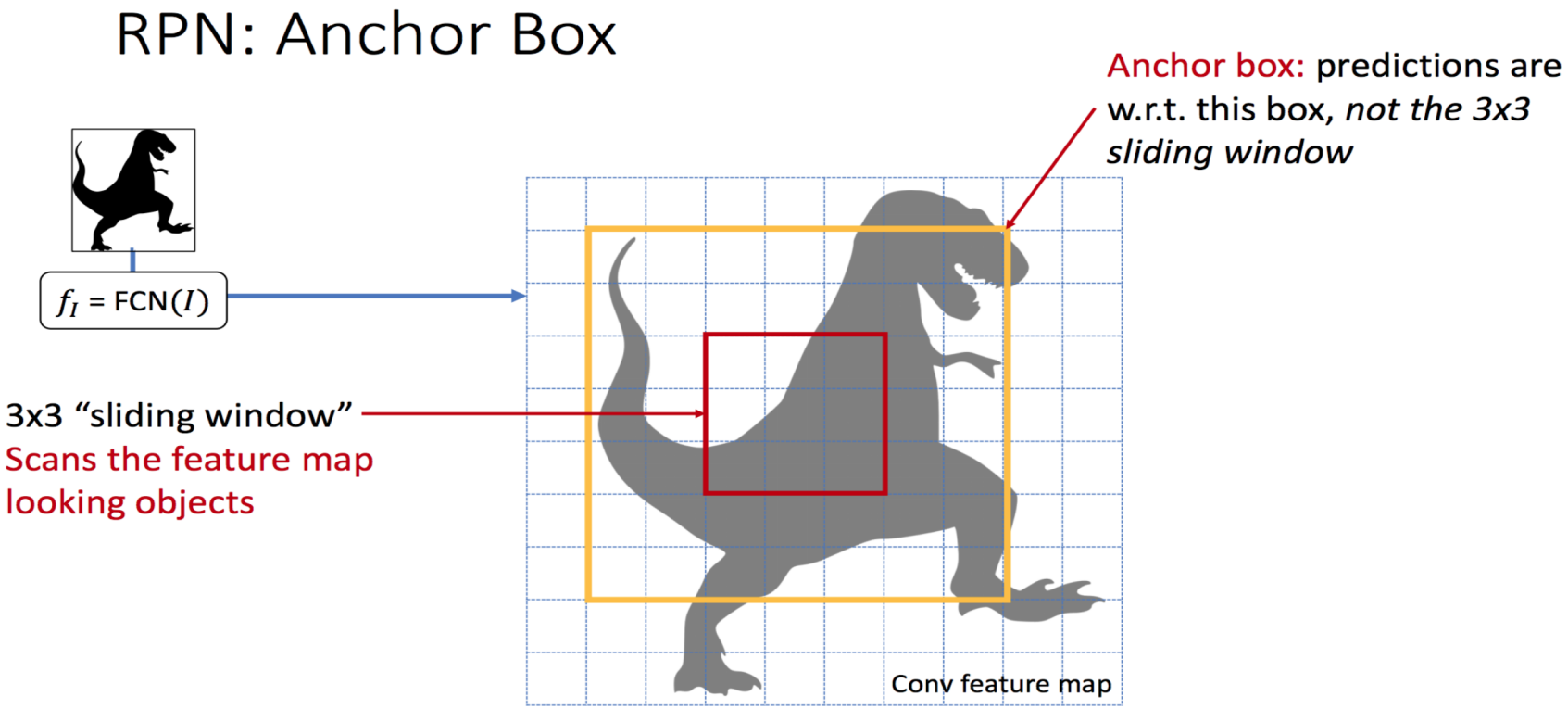
Faster R-CNN (Ren et al. NIPS 2015

	Fast R-CNN	R-CNN
Train time (h)	**9.5**	84
- Speedup	**8.8x**	
Test time / image	**0.32s**	47.0s
- Test speedup	**146x**	
mAP	**66.9%**	66.0%

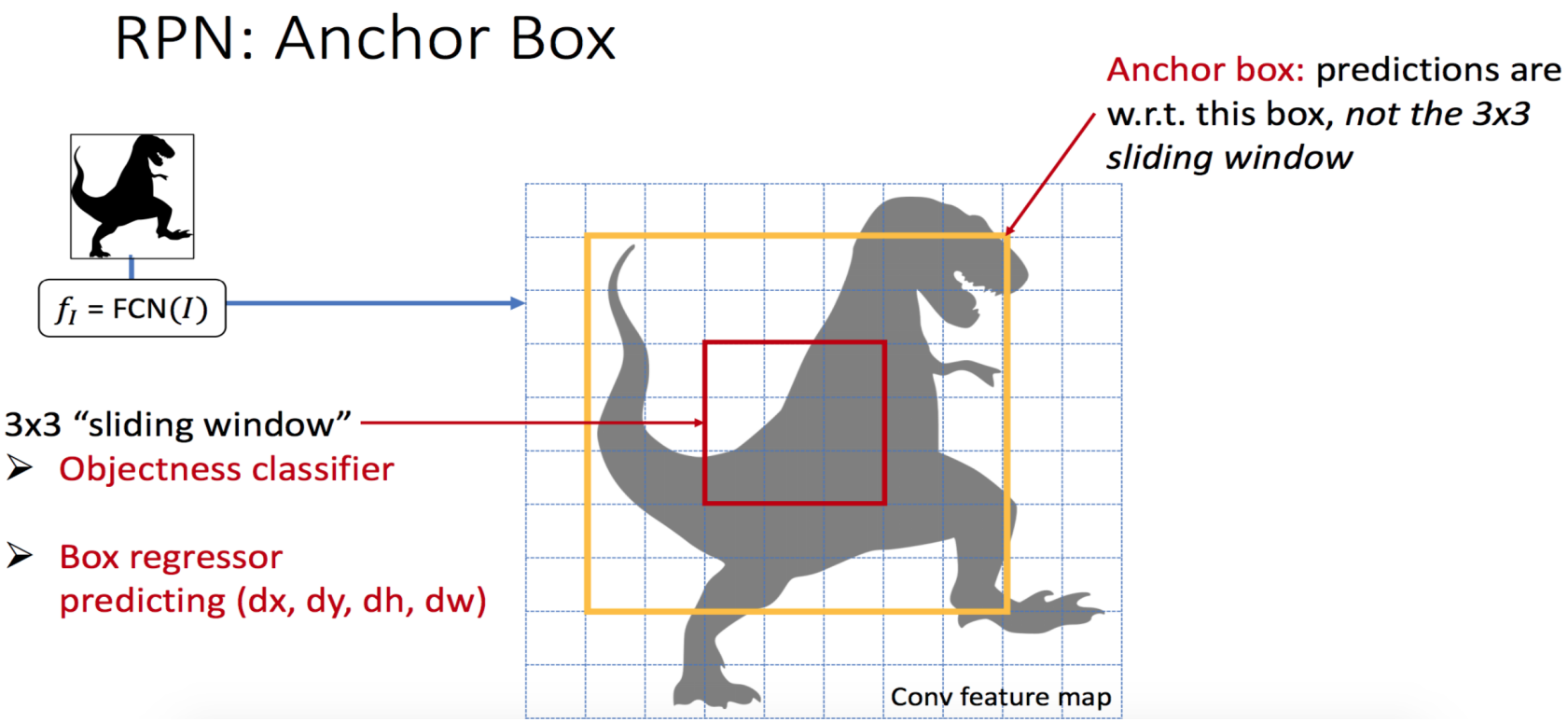
Faster R-CNN (Ren et al. NIPS 2015



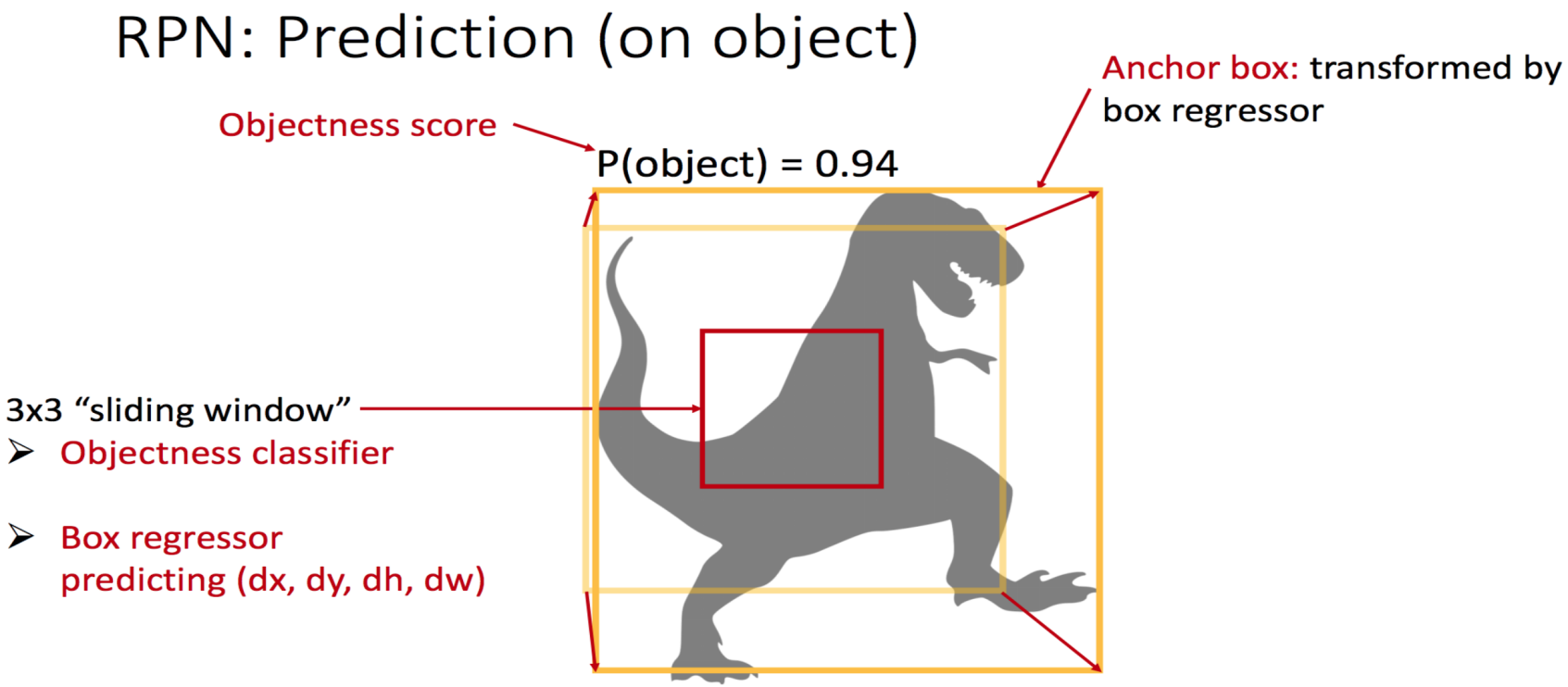
Faster R-CNN (Ren et al. NIPS 2015



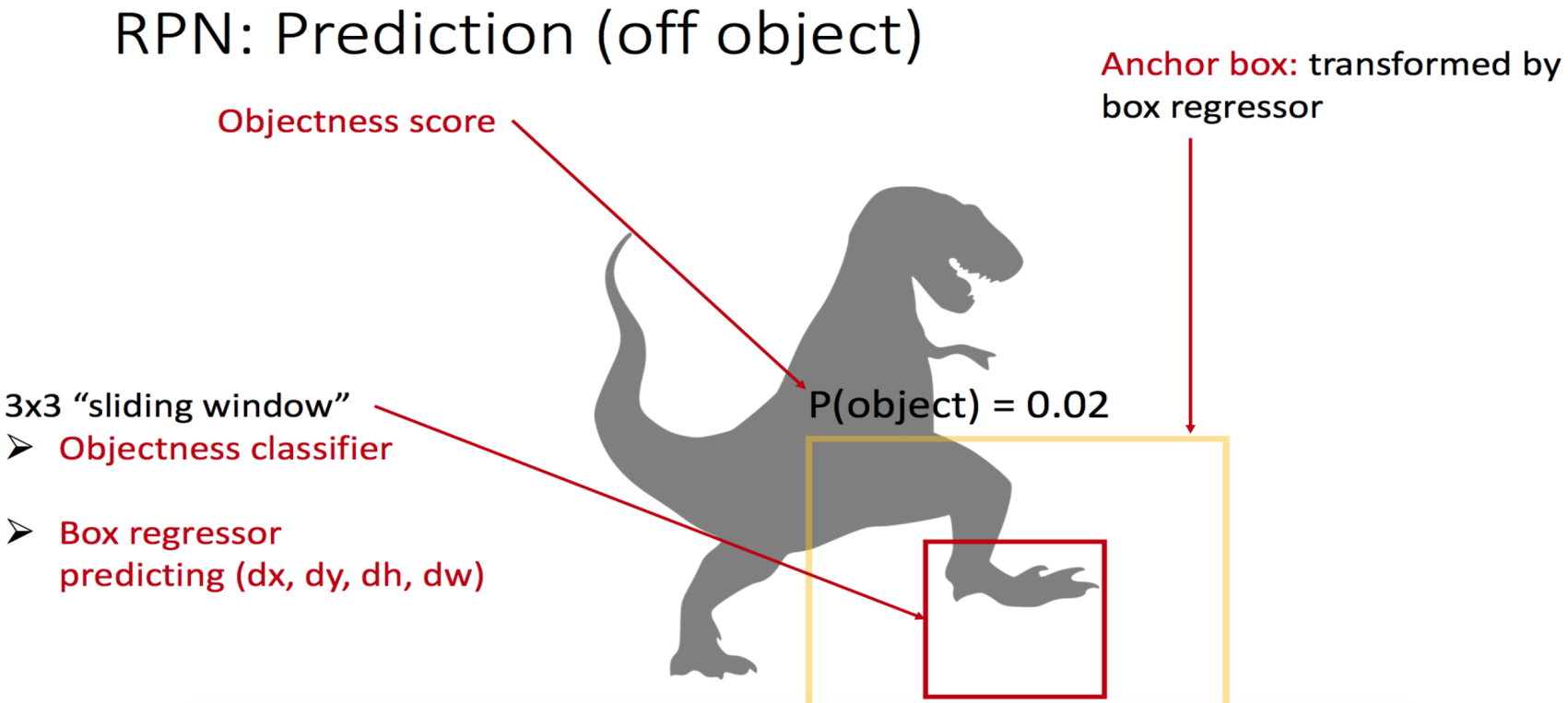
Faster R-CNN (Ren et al. NIPS 2015



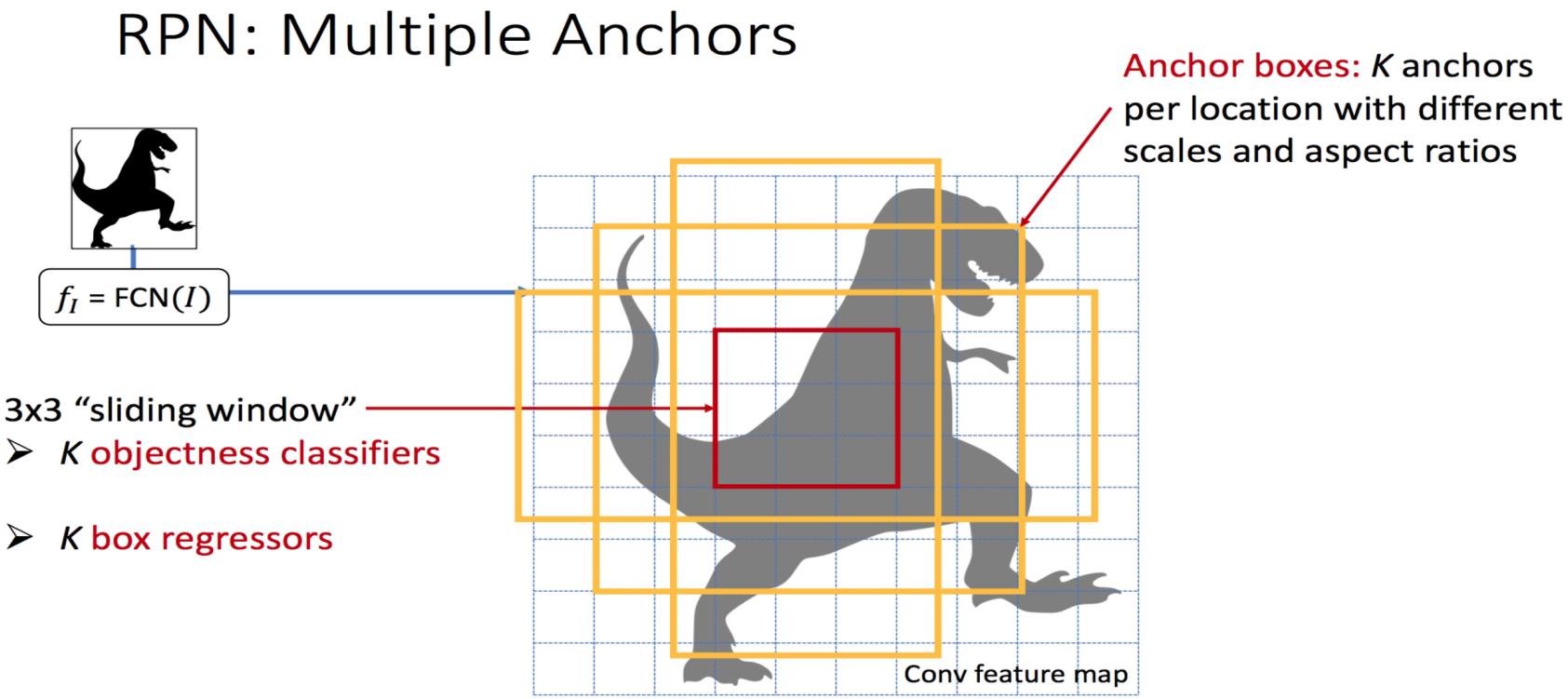
Faster R-CNN (Ren et al. NIPS 2015



Faster R-CNN (Ren et al. NIPS 2015

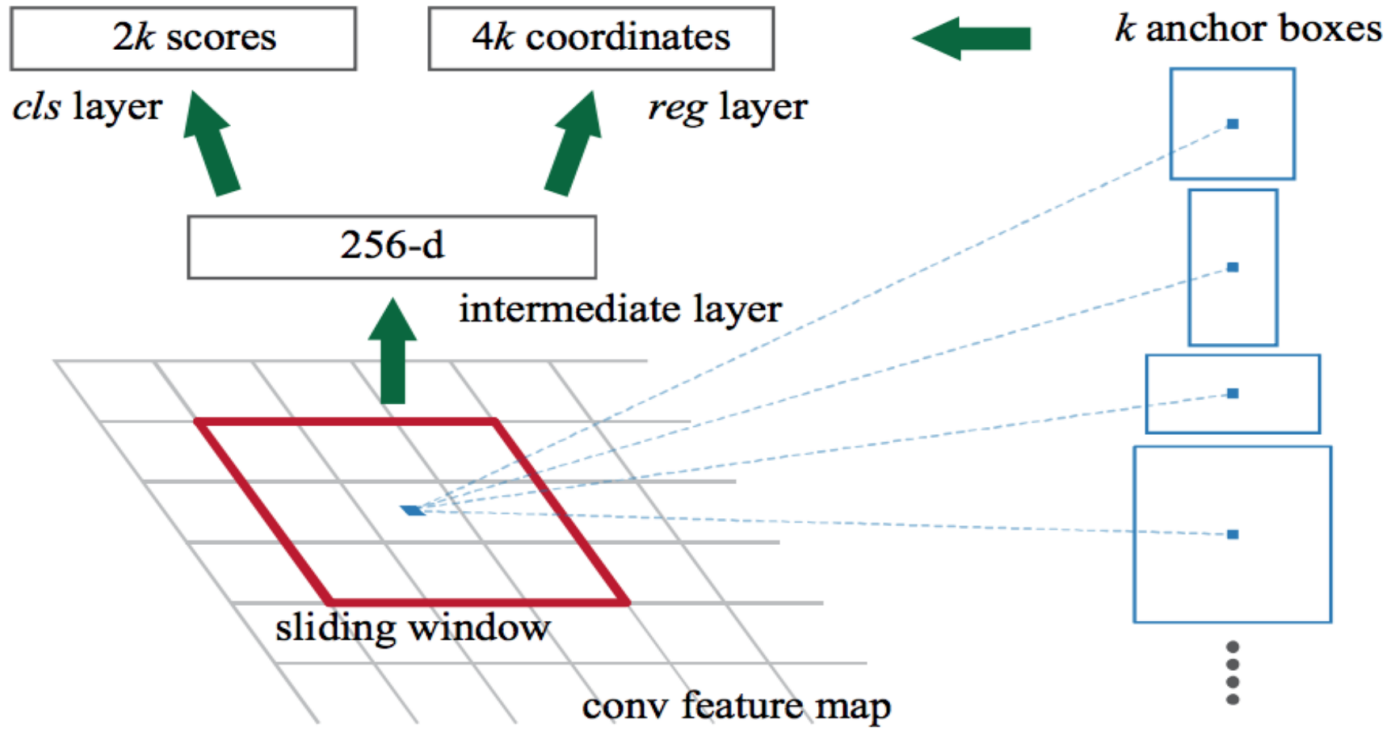


Faster R-CNN (Ren et al. NIPS 2015



Faster R-CNN (Ren et al. NIPS 2015

Region proposal network



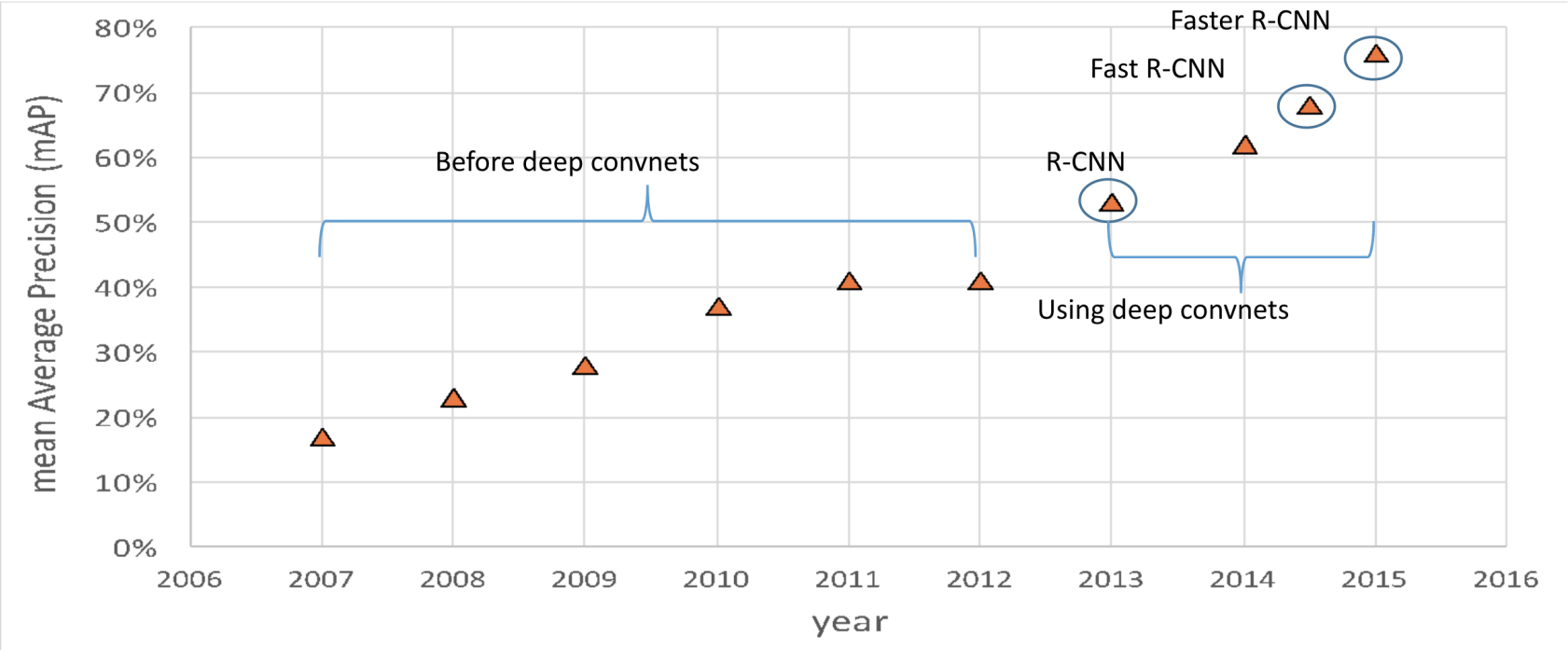
Faster R-CNN (Ren et al. NIPS 2015

Faster R-CNN results

system	time	07 data	07+12 data
R-CNN	~50s	66.0	-
Fast R-CNN	~2s	66.9	70.0
Faster R-CNN	198ms	**69.9**	**73.2**

detection mAP on PASCAL VOC 2007, with VGG-16 pre-trained on ImageNet

Faster R-CNN (Ren et al. NIPS 2015



Faster R-CNN (Ren et al. NIPS 2015)

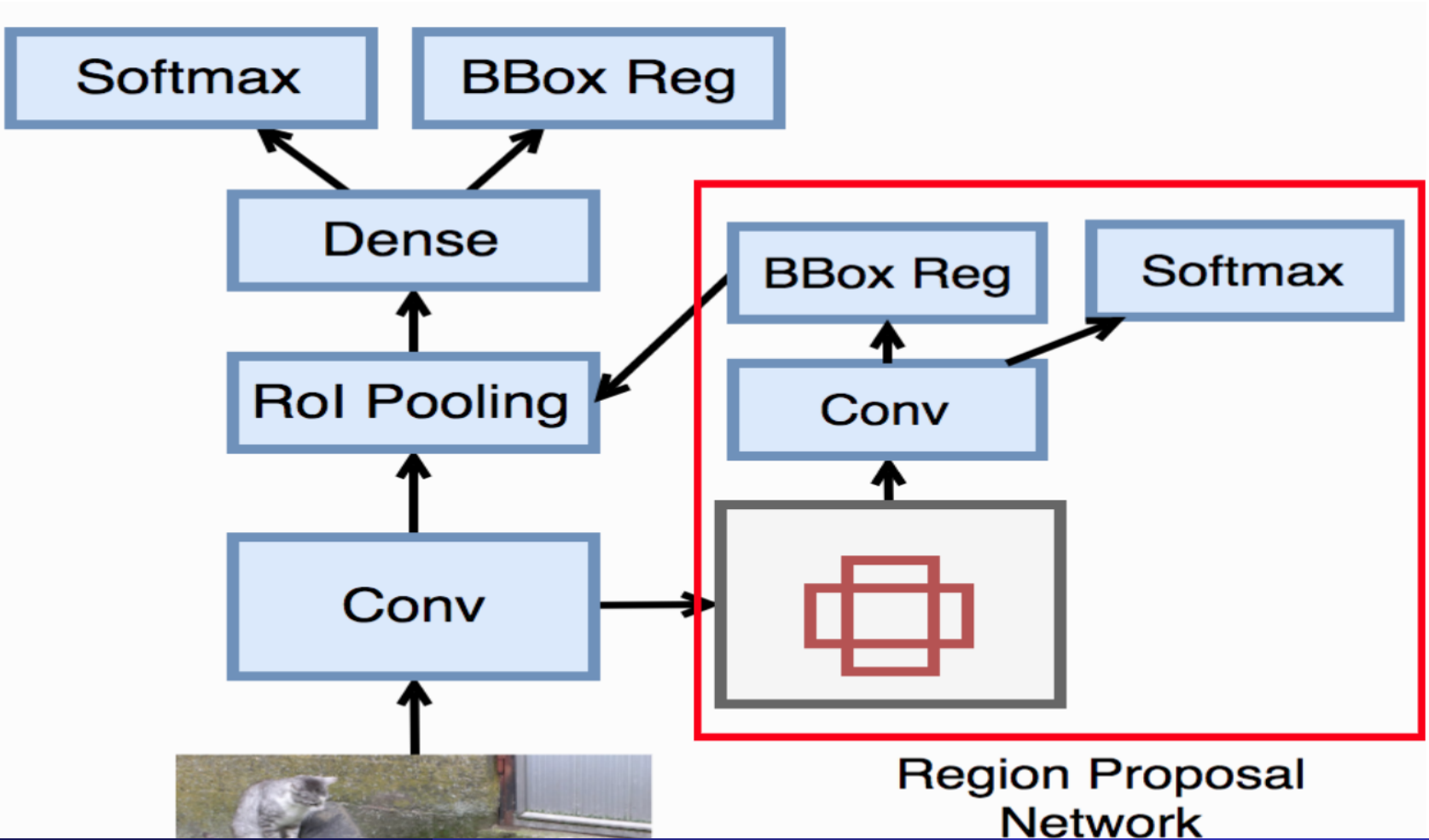
- What could be the problems

Faster R-CNN (Ren et al. NIPS 2015)

- What could be the problems
  - Two-stage detection pipeline is still too slow to apply on real-time images and videos

One-stage detection

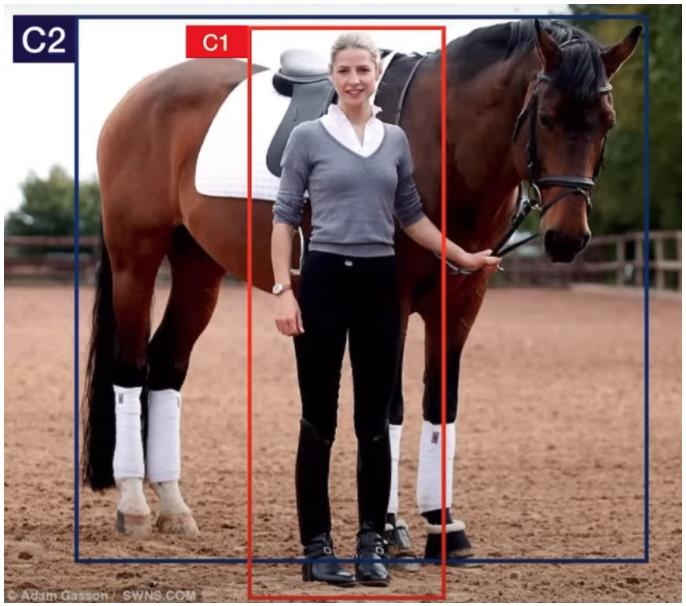
- Solution
  - Don’t generate object proposals!
  - Consider a tiny subset of the output space by design; directly classify this small set of boxes



Object Detection

- Input:
  - Image
- Output:
  - $b_1, b_2, \ldots, b_n$ bounding boxes of n detected objects
  - $c_1, c_2, \ldots, c_n$ class labels of all detected objects

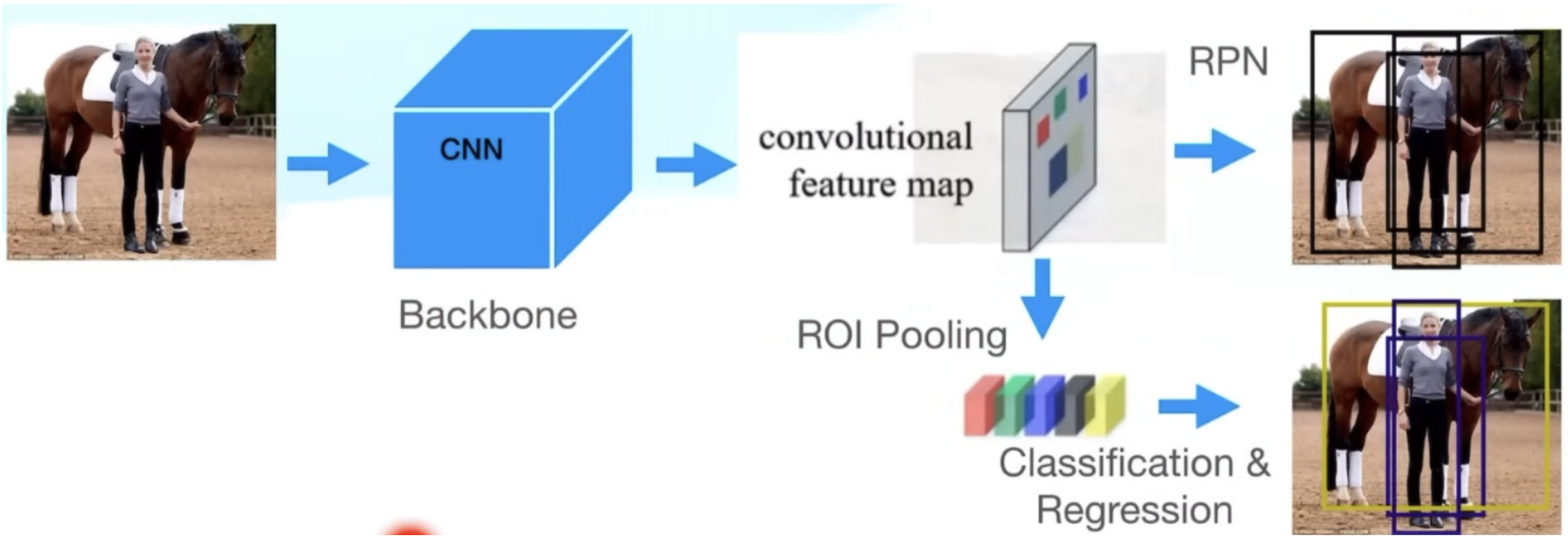
Object Detection



- Input:
  - Image
- Output:
  - $b_1, b_2, \ldots, b_n$ bounding boxes of n detected objects
  - $c_1, c_2, \ldots, c_n$ class labels of all detected objects

Object Detection

- Most algorithms repurpose classifiers to perform object detection.
- Faster-RCNN: Uses RPN to predict boxes and classifier to predict class probabilities.



Drawbacks

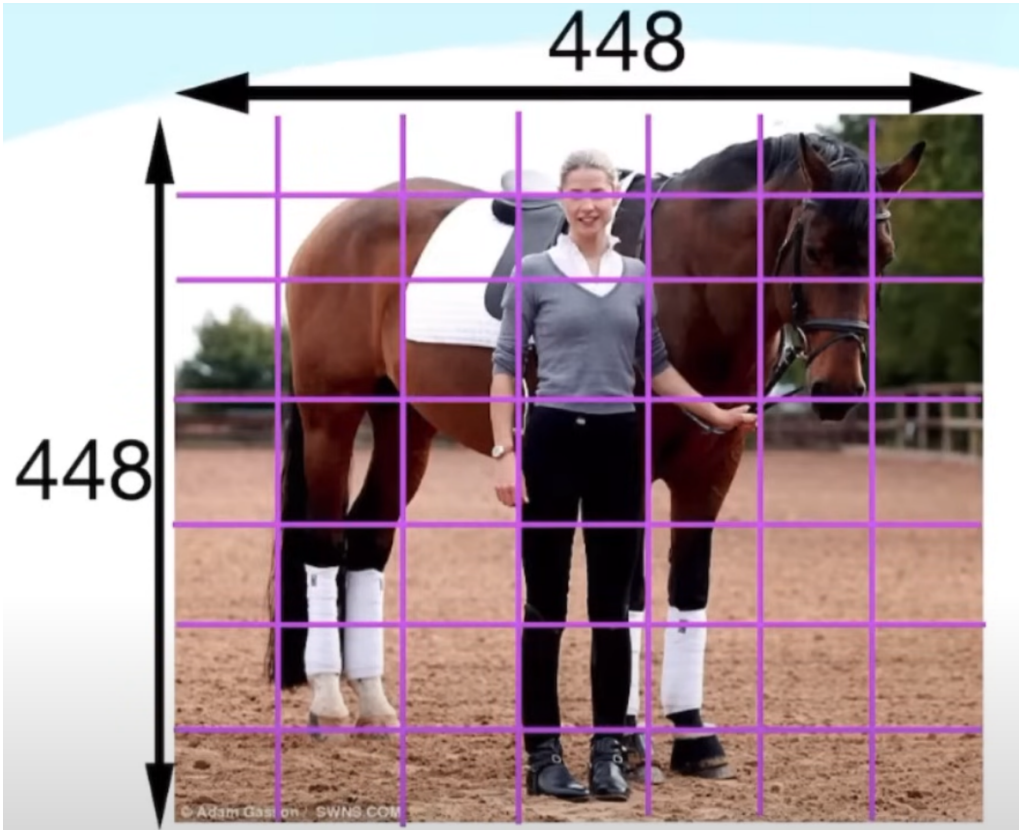
- Multi-Stage pipelines
- Each component trained separately
- Complex and not useful for real-time applications
- Not generalisable to other domains

Idea of YOLO

- Reframe object detection as a single stage regression problem

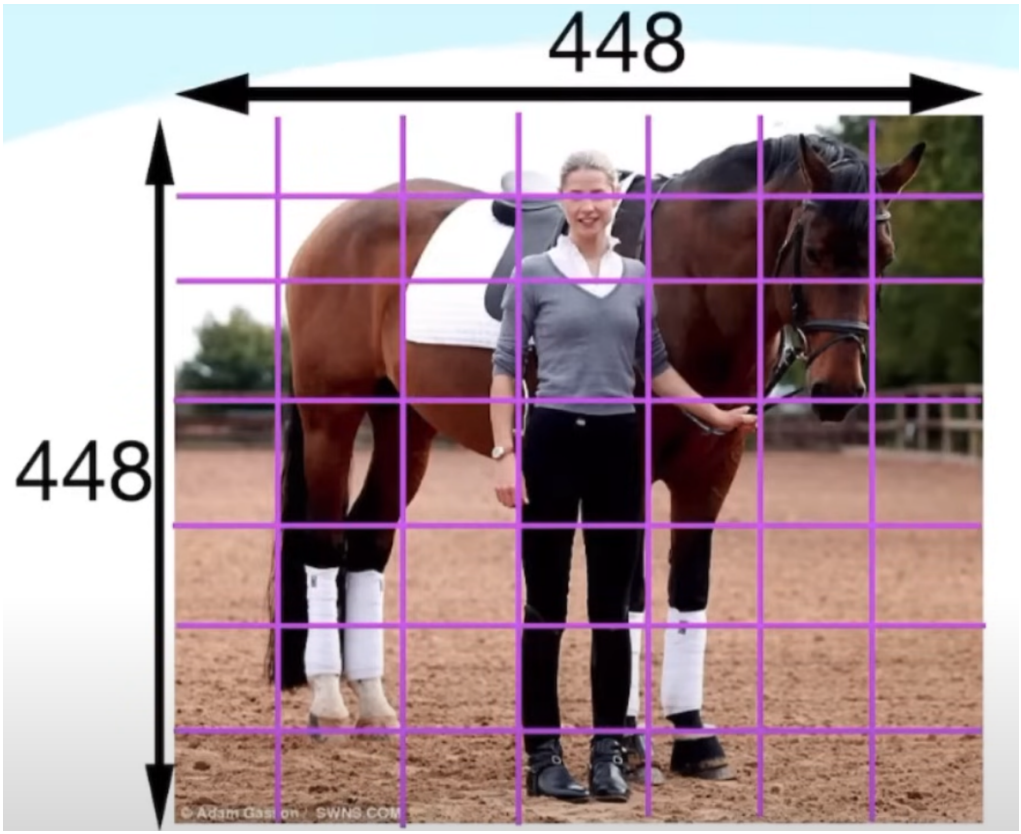
Steps in YOLO

- Take input image
- Resize to 448×448
- Divide into $S \times S$ Grid cells
- S=7 in paper
- Each cell is responsible for predicting one object
- the size of each cell is 64×64



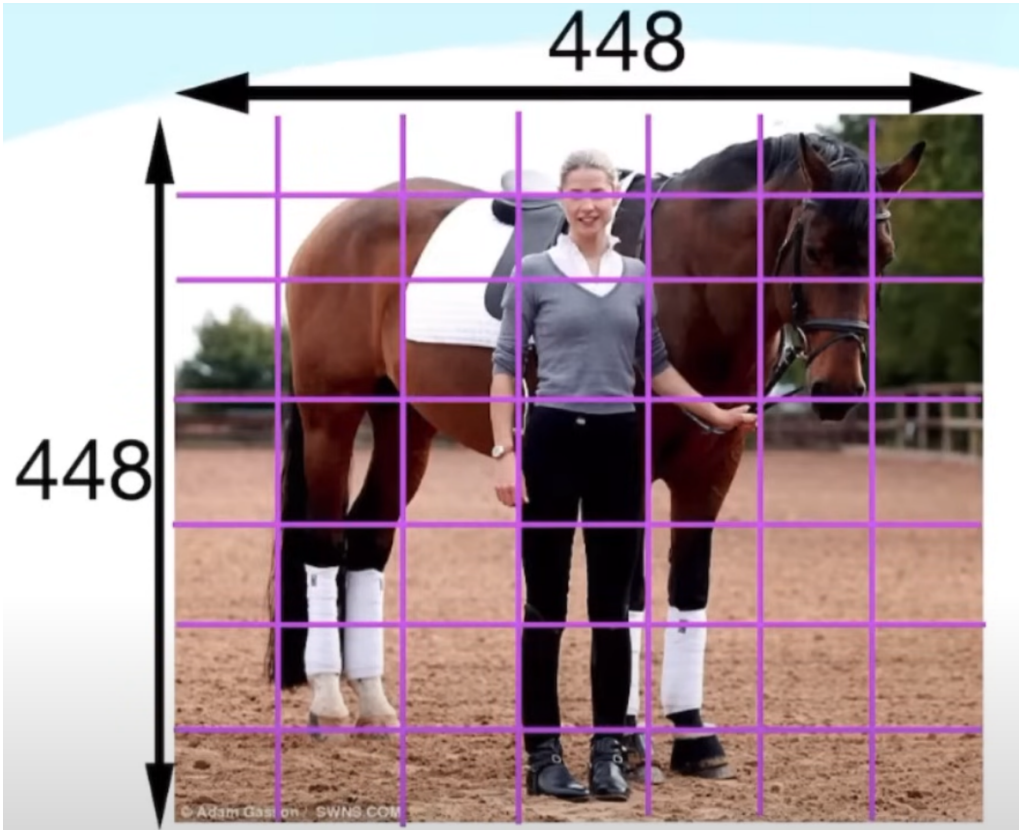
Steps in YOLO

- Take input image
- Resize to 448×448
- Divide into $S \times S$ Grid cells
- S=7 in paper
- Each cell is responsible for predicting one object
- the size of each cell is 64×64
- Which cell is responsible ?



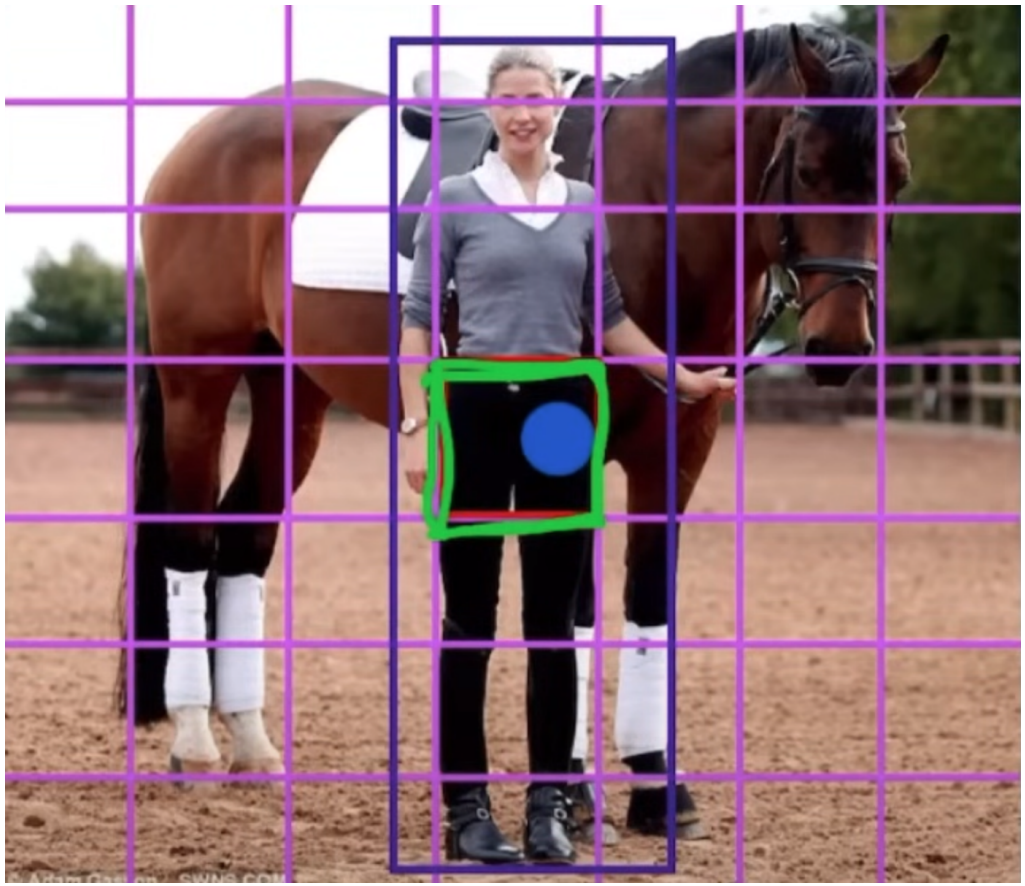
Steps in YOLO

- Take input image
- Resize to 448×448
- Divide into $S \times S$ Grid cells
- S=7 in paper
- Each cell is responsible for predicting one object
- the size of each cell is 64×64
- Which cell is responsible ?
- Where Center of objects falls into



Bounding Boxes

- Box-x,y,w,h
- How are these encoded?
- Relative to Grid cell that the object center falls into
- $(200, 311, 142, 250)$



Bounding Boxes

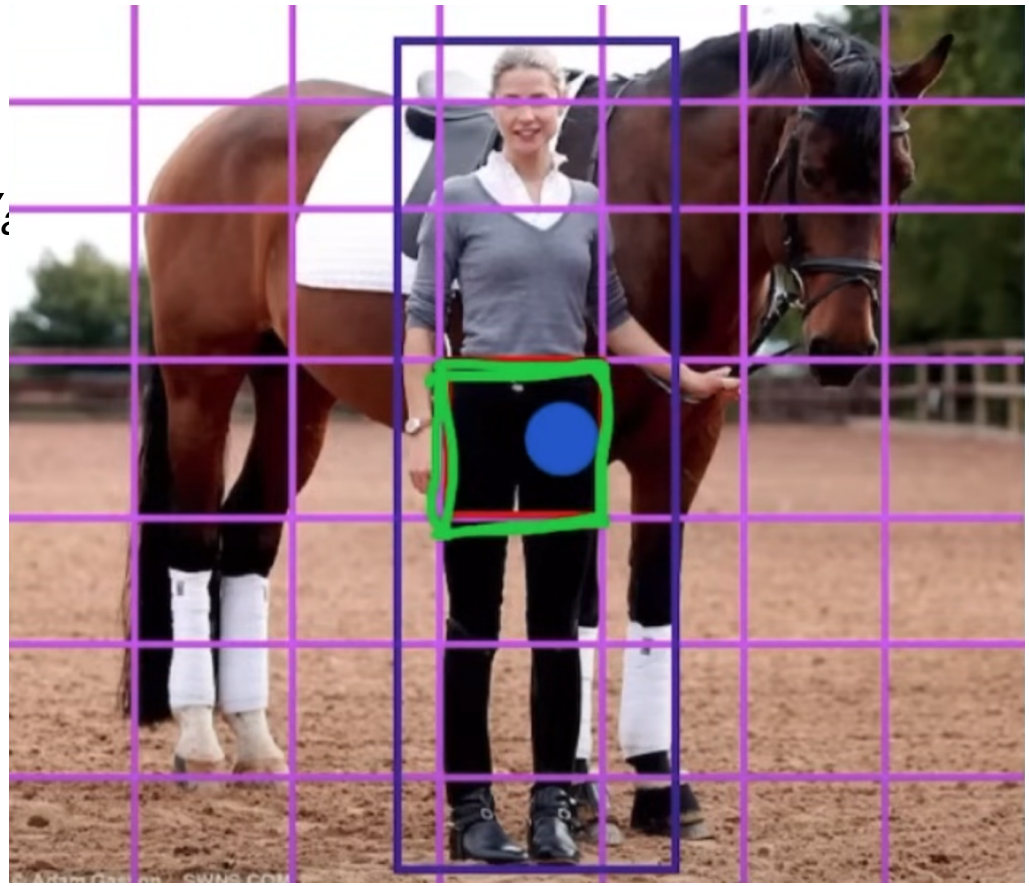
- Center point (x,y):Relative to anchor that (x,y) falls into.

$$\delta x = (x - x_a)/64 \delta y = (y - y_a$$
(1)

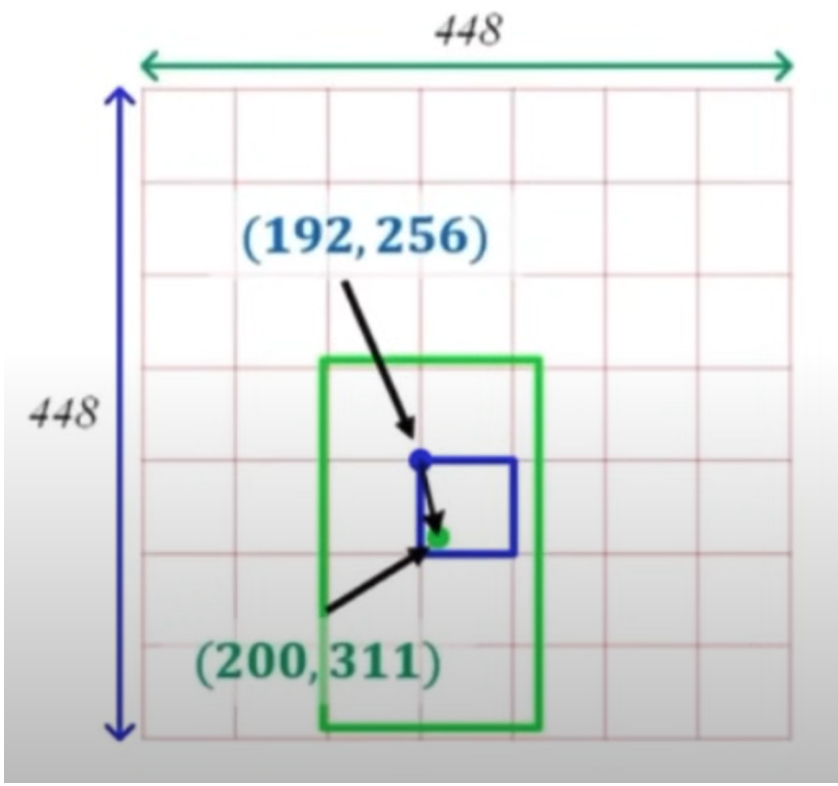
where x_a, y_a : the coordinates of left-top point

- Width/height (w,h):relative to the whole image

$$\delta w = w/448 \delta h = h/448$$
(2)



Example Calculation-Targets



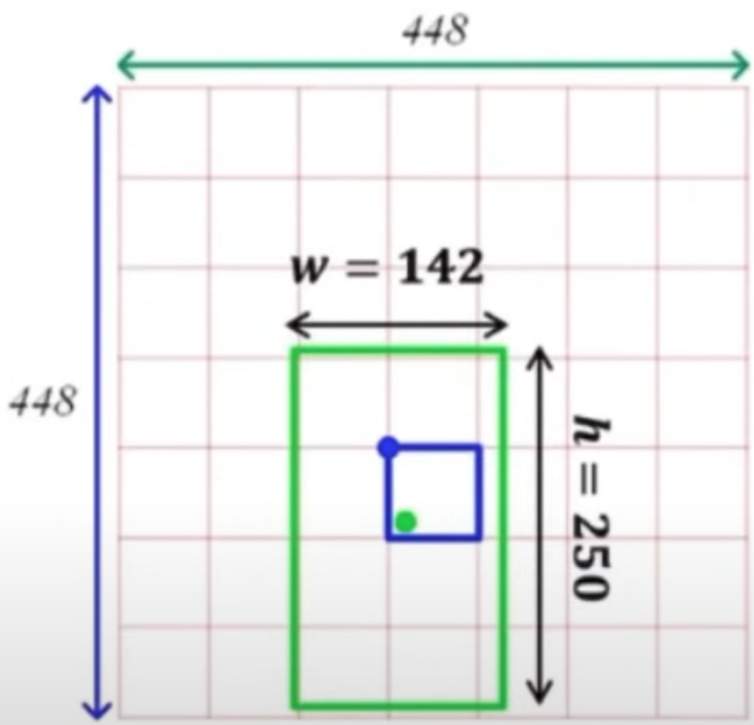
$$\Delta x = \frac{200 - 192}{64} \approx 0.13$$

$$\Delta y = \frac{311 - 256}{64} \approx 0.87$$

$$\Delta w = w/448$$

$$\Delta h = h/448$$

Example Calculation-Targets



$$\Delta x = \frac{200 - 192}{64} \approx 0.13$$

$$\Delta y = \frac{311 - 256}{64} \approx 0.87$$

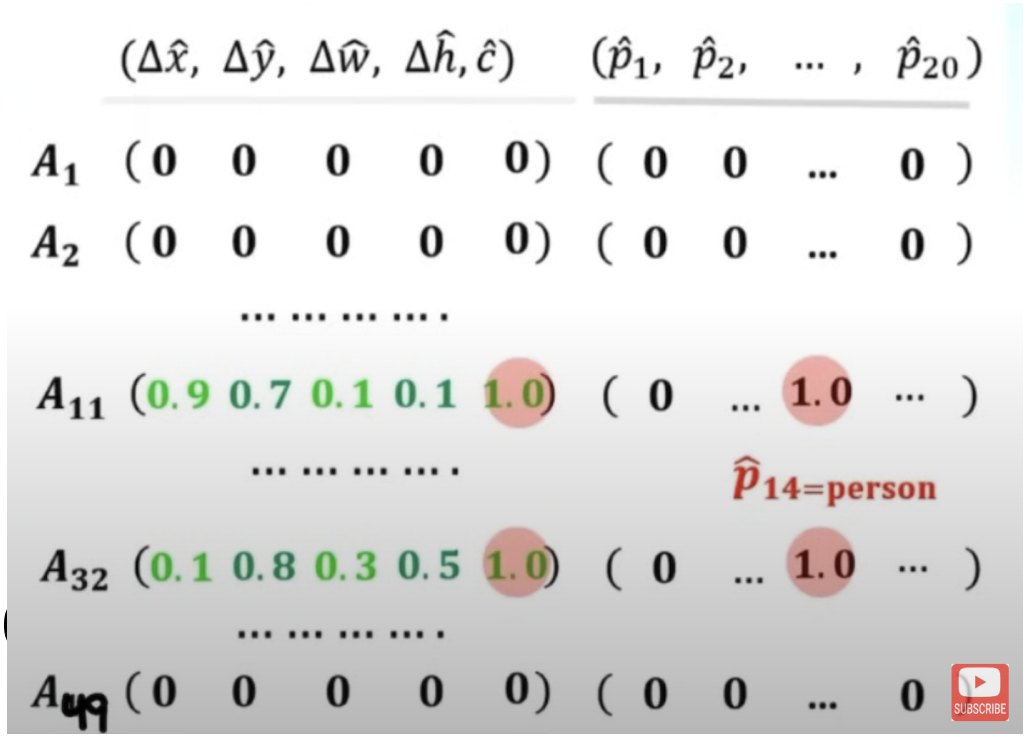
$$\Delta w = \frac{142}{448} \approx 0.31$$

$$\Delta h = \frac{250}{448} \approx 0.56$$

$$(200, 311, 142, 250) \Rightarrow (0.13,\ 0.87,\ 0.31,\ 0.56)$$

Label Encoding

- For every grid cell (Anchor box), we need to create targets/labels
- No object-All Zeros
- Object-Relative Values w.r.t. grid
- Classes-one-hot encoding
- Ex:(x,y,w,h,c,)=(0.9,0.7,0.1,(



  - Classes=(1.0,0,0,. . . ,0)-20 values

Prediction Vector

- Each grid cell (we consider that as anchor here) predicts:
  - 2 Bounding boxes (B=2)
  - For each bounding box:

$$(\Delta x_i, \Delta y_i, \Delta w_i, \Delta h_i, c_i)_{i=1}^{B} \tag{3}$$

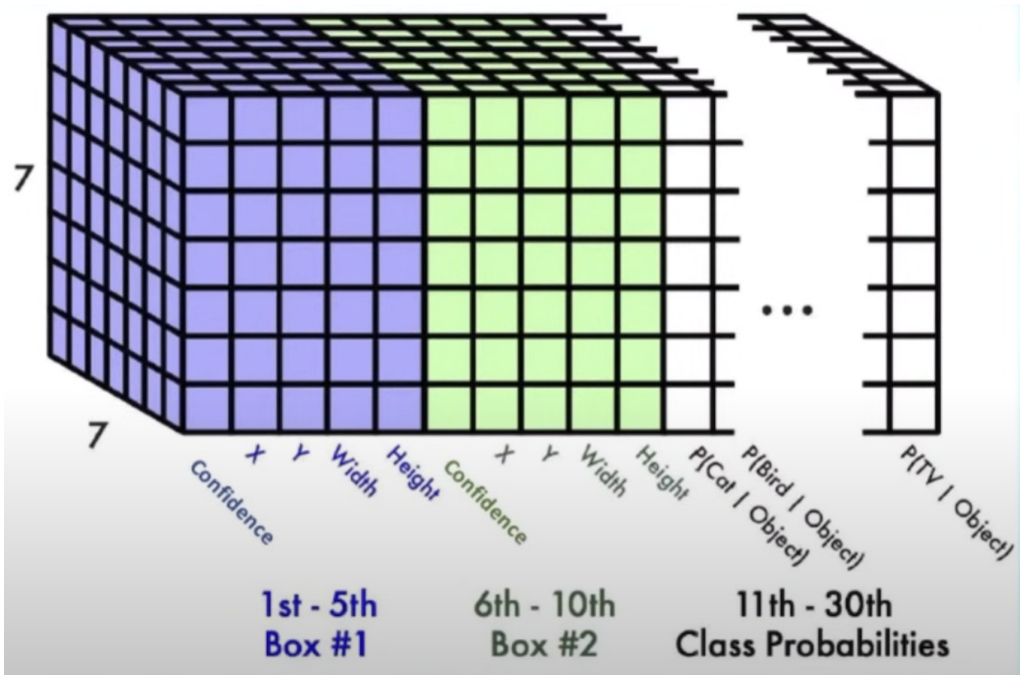
  - Conditional class probabilities (n=20)
    -

$$(P_1, P_2, \ldots, P_20) \tag{4}$$

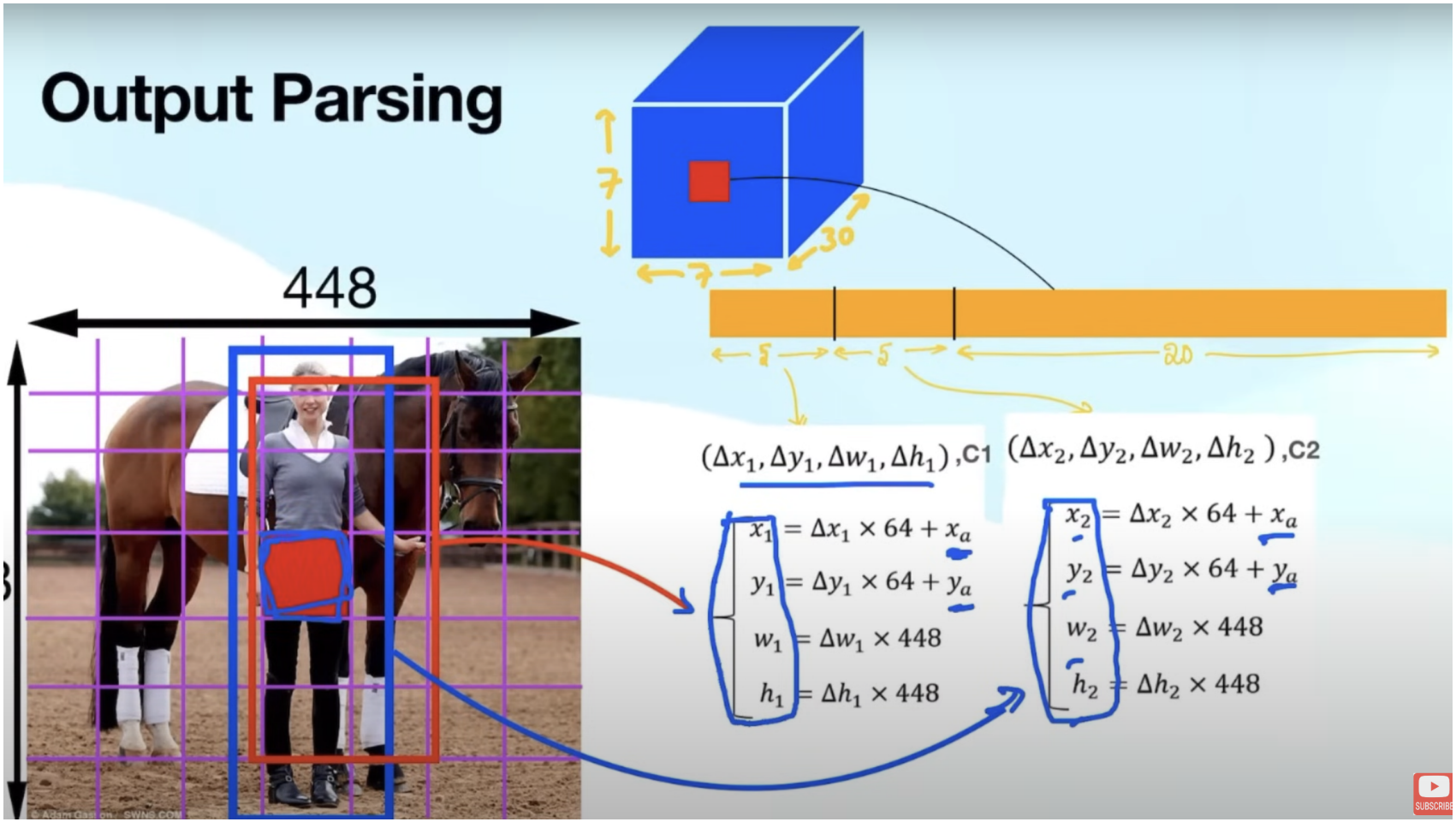
  - Number of parameters per grid cell: (2×5+20=30)

Prediction Vector

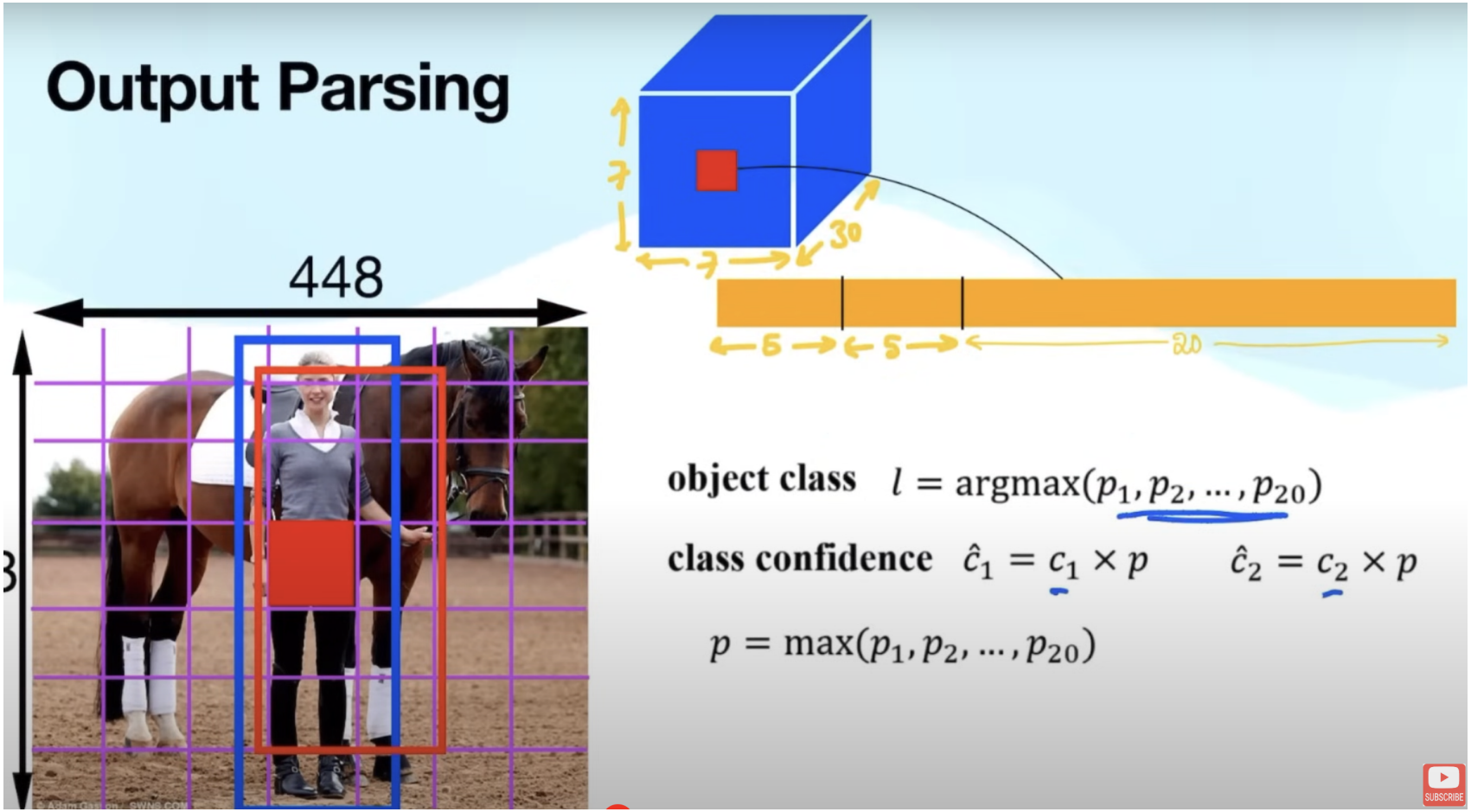
- Output layer-$7 \times 7 \times 30$



Output Parsing

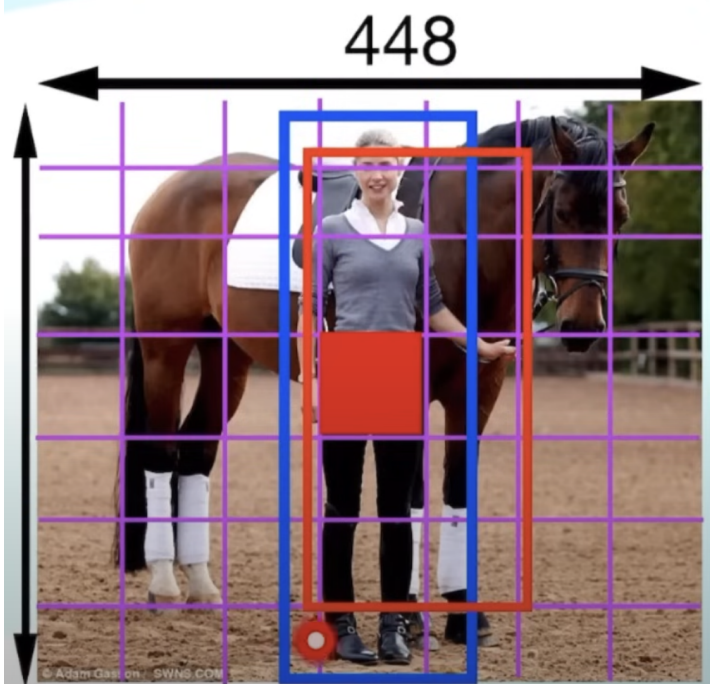


Output Parsing



Output Parsing

Post processing



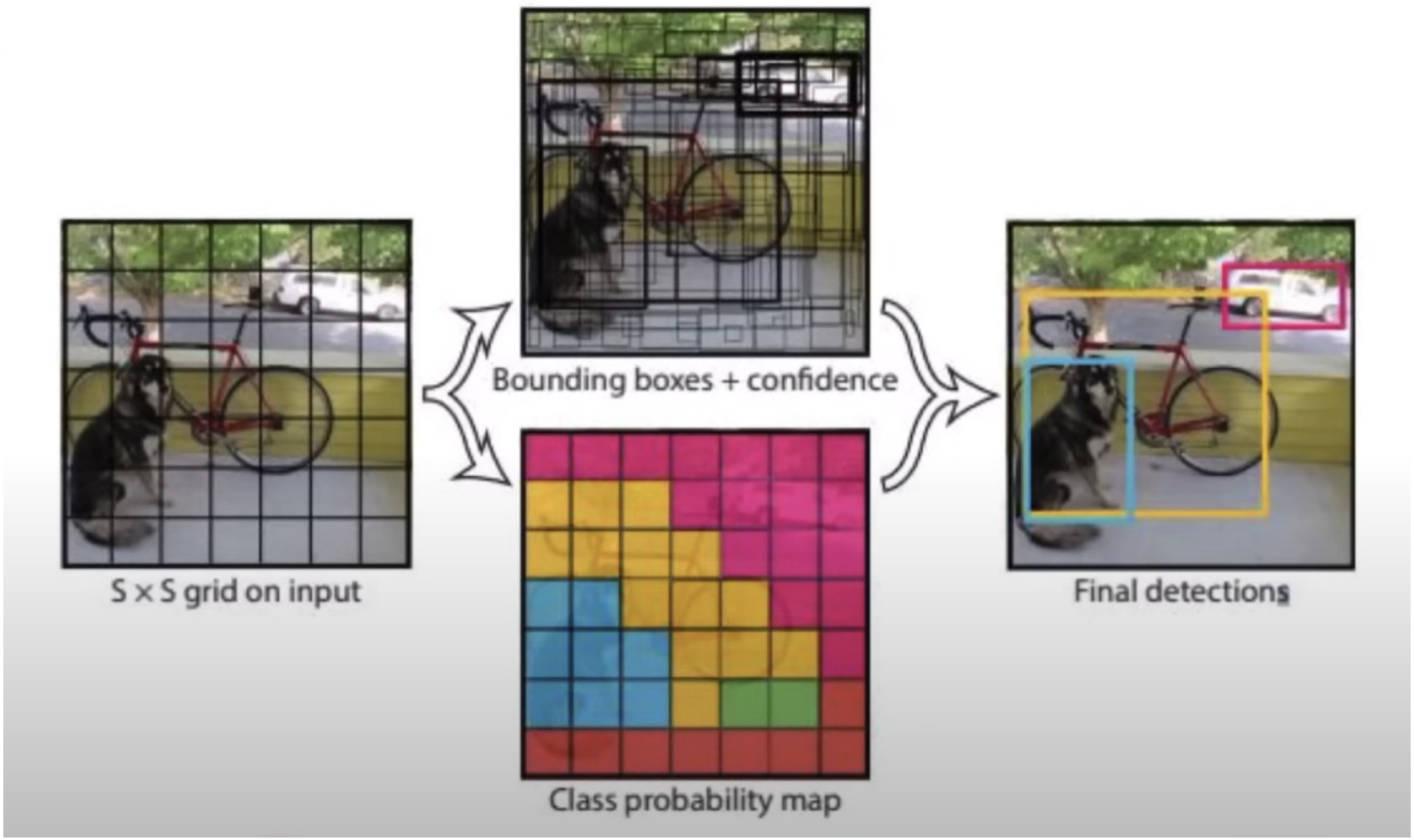
object class $l = \text{argmax}(p_1, p_2, \ldots, p_{20})$

class confidence $\hat{c}_1 = c_1 \times p \qquad \hat{c}_2 = c_2 \times p$

$p = \max(p_1, p_2, \ldots, p_{20})$

* Consider the box with highest confidence score per each grid

Overview So Far



Architecture

- Inspired by GoogleNet Model
- Network:
  - 24 Convolution all layers
  - 2 Fully connected layers

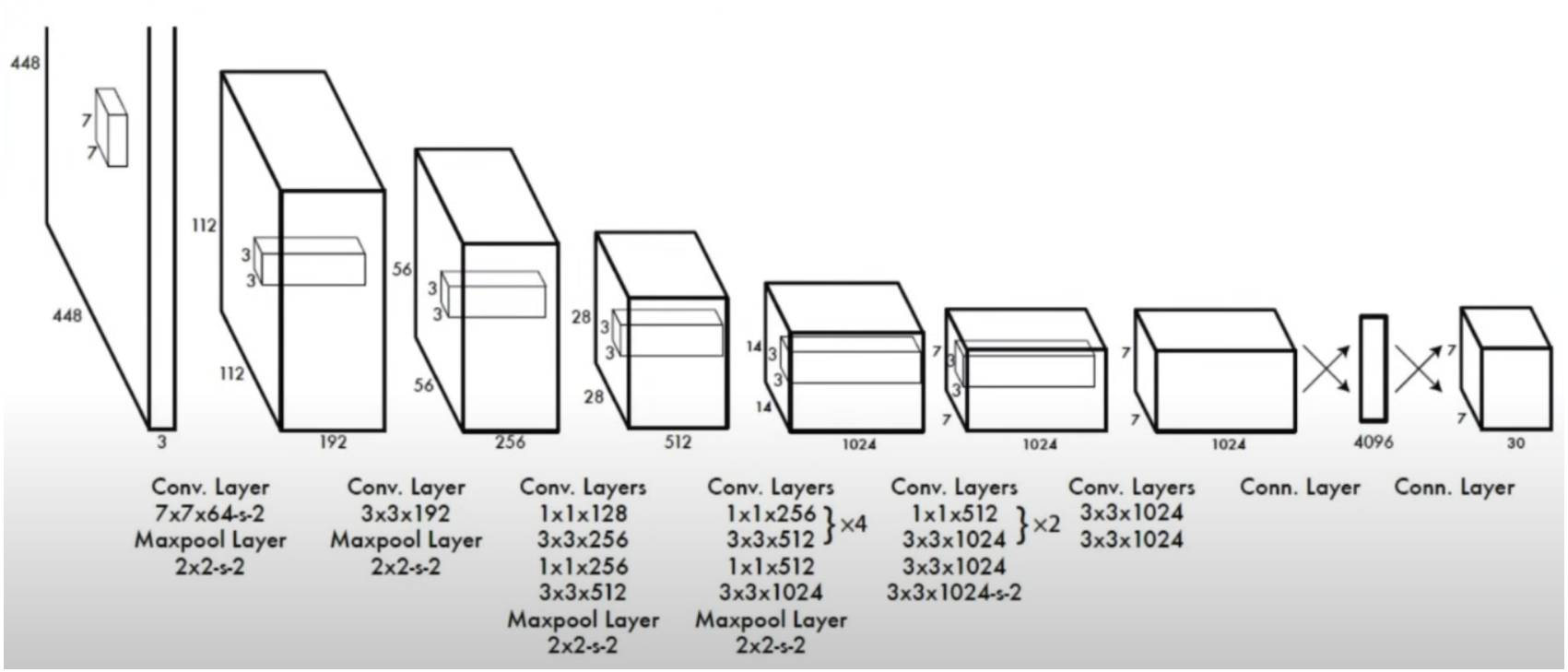
	Type	Size	Filters	Stride	Output	
	Conv.	7 x 7 x 3	64	2	224 x 224 x 64	
	max pool	2 x 2			112 x 112 x 64	
	Conv.	3 x 3 x 64	192	1	112 x 112 x 192	
	max pool	2 x 2			56 x 56 x 192	
	Conv.	1 x 1 x 192	128	1	56 x 56 x 128	6
	Conv.	3 x 3 x 128	256	1	56 x 56 x 256	
	Conv.	1 x 1 x 256	256	1	56 x 56 x 256	
	Conv.	3 x 3 x 256	512	1	56 x 56 x 512	
	max pool	2 x 2			28 x 28 x 512	
4 x	Conv.	1 x 1 x 512	256	1	28 x 28 x 256	$4 \times 2 = 8$
	Conv.	3 x 3 x 256	512	1	28 x 28 x 512	
	Conv.	1 x 1 x 512	512	1	28 x 28 x 512	2
	Conv.	3 x 3 x 512	1024	1	28 x 28 x 1024	
	max pool	2 x 2			14 x 14 x 1024	
2 x	Conv.	1 x 1 x 1024	512	1	14 x 14 x 512	$2 \times 2 = 4$
	Conv.	3 x 3 x 512	1024	1	14 x 14 x 1024	
	Conv.	3 x 3 x 1024	1024	1	14 x 14 x 1024	4
	Conv.	3 x 3 x 1024	1024	2	7 x 7 x 1024	
	Conv.	3 x 3 x 1024	1024	1	7 x 7 x 1024	
	Conv.	3 x 3 x 1024	1024	1	7 x 7 x 1024	

$6 + 8 + 2 + 4 + 4 = 24$

Architecture

- Flatten the last conv map $7 \times 7 \times 1024$ to 50176 feature vector
- Pass through 2 fully connected layers
- Output-1470 feature vector
- Reshape 1470 vector to $7 \times 7 \times 30$ feature map

Architecture

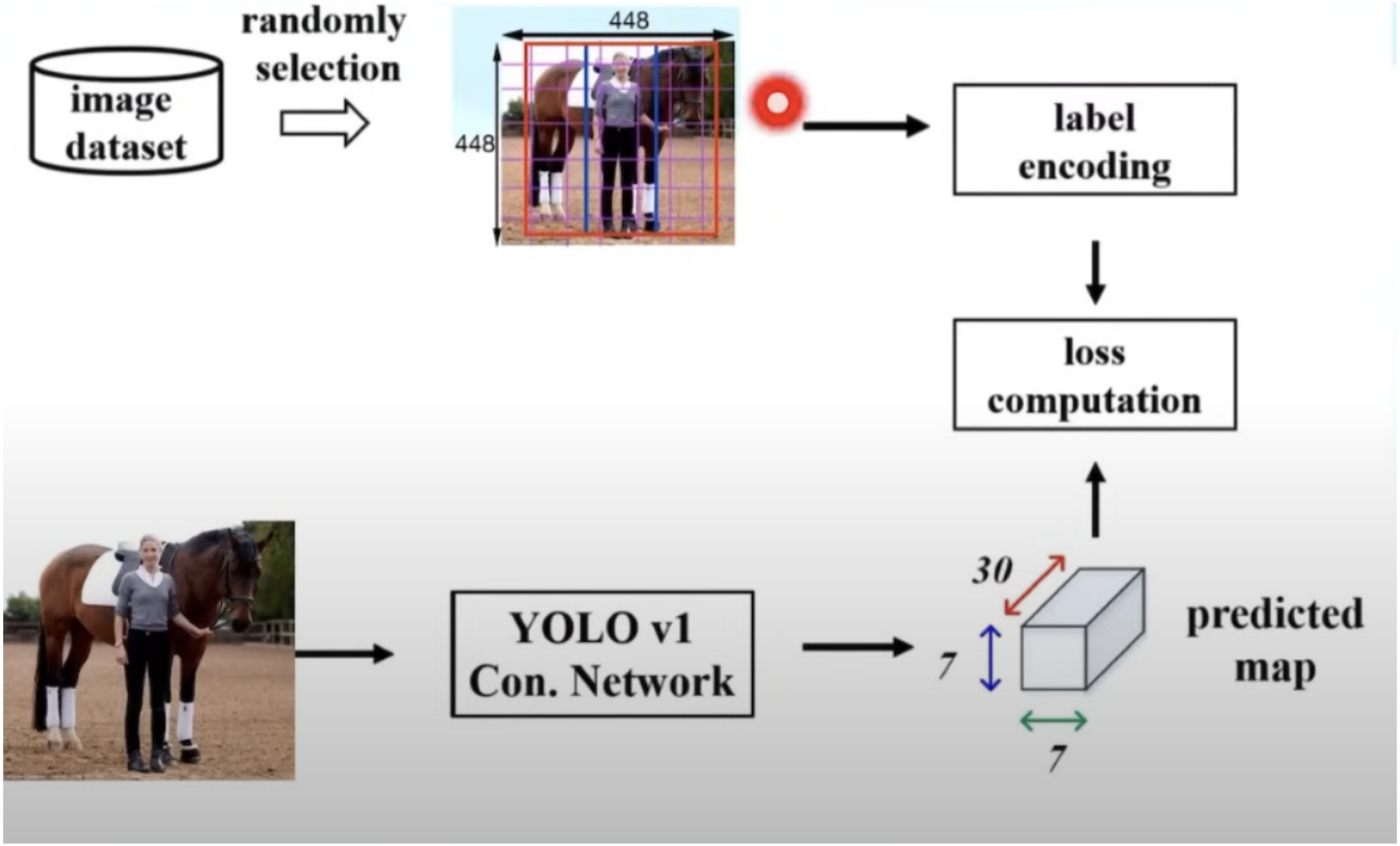


Training Process

- Dataset: Pascal VOC-20 classes
- Network pretrained on Imagenet at 224×224
- Actual training on 448×448 on VOC dataset

Training Process

Dataset: Pascal VOC-20 classes



Loss Function

- Loss L is the sum of losses over all grid cells S
- Put more importance on grid cells thtat contain objects
- Decrease the importance of grid cells having no objects
- Ex: 2 objects cells, 47 no -object cells

$$L = \sum_{i=1}^{S} {}^{2} L_i \tag{5}$$

Loss Function

- Loss L is the sum of losses over all grid cells S
- Put more importance on grid cells thtat contain objects
- Decrease the importance of grid cells having no objects

$$L = \sum_{i=1}^{S} {}^{2}1_{i}^{obj} L_i, obj + \lambda_{no_obj} \sum_{i=1}^{S} {}^{2}1_{i}^{no-obj} L_i, no - obj \tag{6}$$

Loss for Object cells

- Loss=Objectness loss+classification loss+Box Regression Loss
- Put more weightage on box parameters

$$L_{i,obj} = L^{box}_{i,obj} + L^{conf}_{i,obj} + L^{cls}_{i,obj} \tag{7}$$

Bounding box loss

- Sum of squared errors on predicted box parameters and ground truth labels

$$L_{i,obj}^{box} = \left(\Delta x_i^* - \Delta \hat{x}_i\right)^2 + \left(\Delta y_i^* - \Delta \hat{y}_i\right)^2 + \left(\sqrt{\Delta w_i^*} - \sqrt{\Delta \hat{w}_i}\right)^2 + \left(\sqrt{\Delta h_i^*} - \sqrt{\Delta h_i}\right)^2$$

- $(\Delta \hat{x}_i, \Delta \hat{y}_i, \Delta \hat{w}_i, \hat{h}_i)$: ground-truth box
- $(\Delta x_i^*, \Delta y_i^*, \Delta w_i^*, \Delta h_i^*)$: **responsible** predicted box that has the largest IoU with ground-truth box

References

The End

Questions? Comments?