

JOSEPH

ROSS

SANTOSH

ALI

REDMON

GIRSHICK

DIVVALA

FARHADI

Dog



“YOU ONLY LOOK ONCE”  
REAL-TIME  
DETECTION

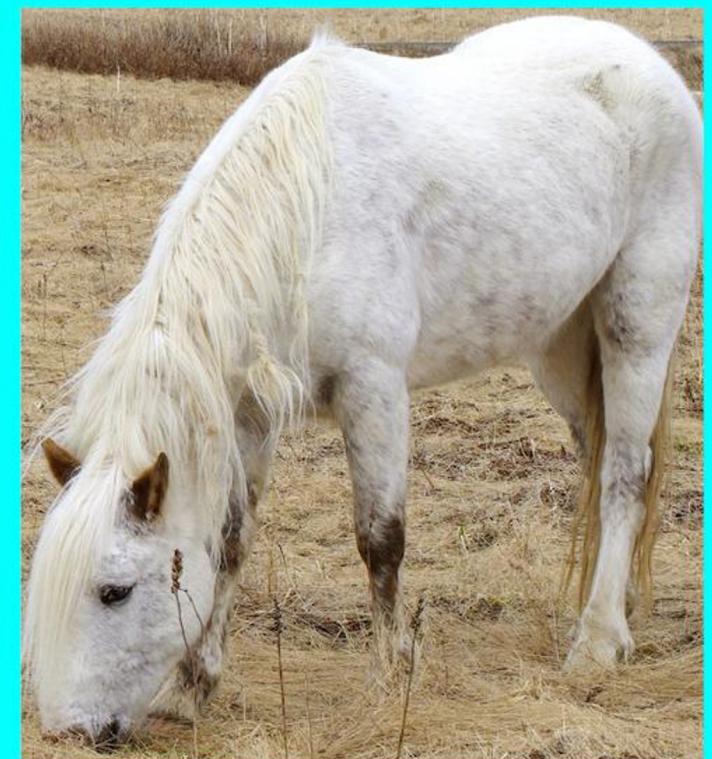
Dog



Person



Horse



# Accurate object detection is slow!

	<b>Pascal 2007 mAP</b>	<b>Speed</b>	
DPM v5	33.7	.07 FPS	14 s/img

# Accurate object detection is slow!

	<b>Pascal 2007 mAP</b>	<b>Speed</b>	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img

# Accurate object detection is slow!

	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img



$\frac{1}{3}$  Mile, 1760 feet



# Accurate object detection is slow!

	<b>Pascal 2007 mAP</b>	<b>Speed</b>	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img



176 feet



# Accurate object detection is slow!

	<b>Pascal 2007 mAP</b>	<b>Speed</b>	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img



# Accurate object detection is slow!

	<b>Pascal 2007 mAP</b>	<b>Speed</b>	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	63.4	45 FPS	22 ms/img



2 feet  
→

A green arrow points horizontally to the right, indicating a distance of two feet between the front of the car and the text "2 feet".

# Accurate object detection is slow!

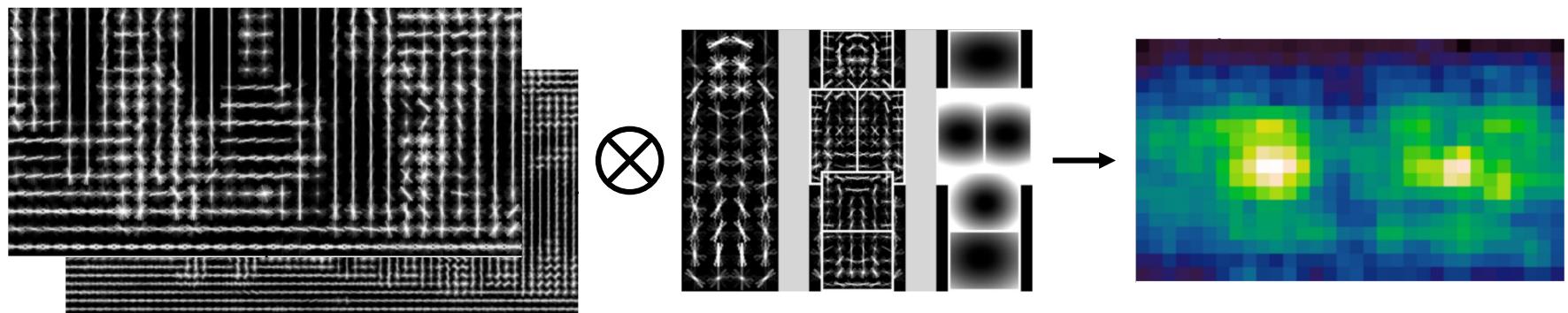
	<b>Pascal 2007 mAP</b>	<b>Speed</b>	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	<del>63.4</del> 69.0	45 FPS	22 ms/img



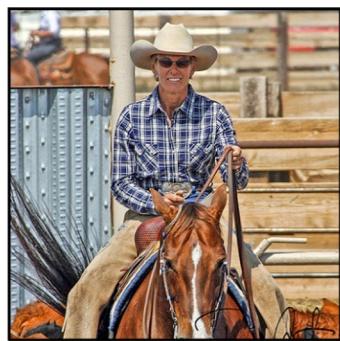
2 feet  
→

A green arrow points horizontally to the right, indicating a distance of two feet between the front of the car and the text "2 feet".

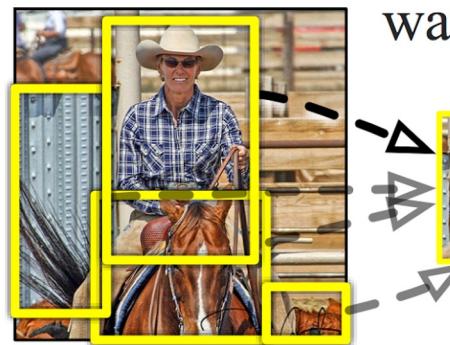
## DPM: *Deformable Part Models*



## R-CNN: *Regions with CNN features*

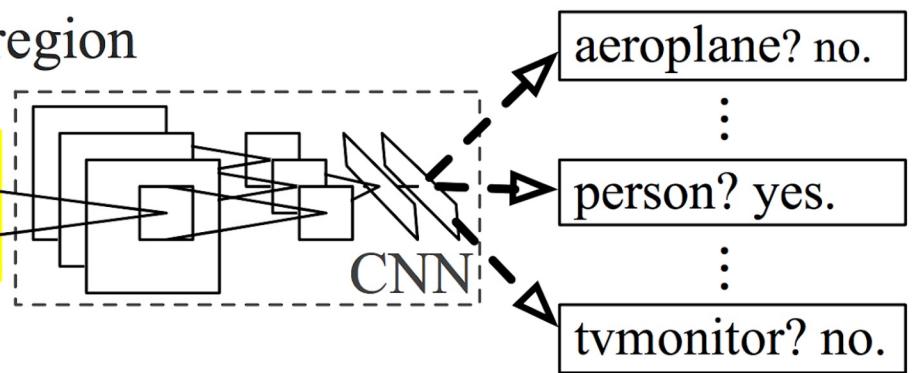


1. Input image



2. Extract region proposals (~2k)

warped region

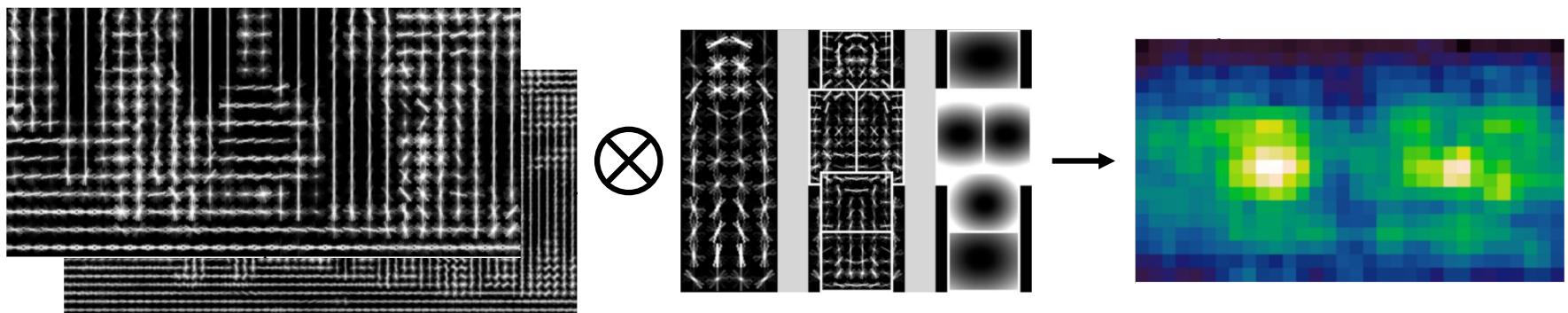


3. Compute CNN features

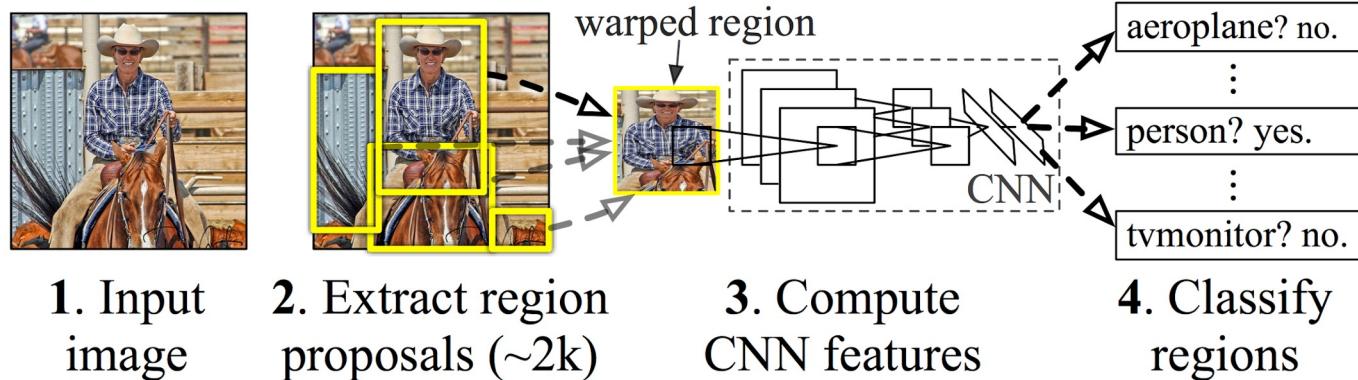
4. Classify regions

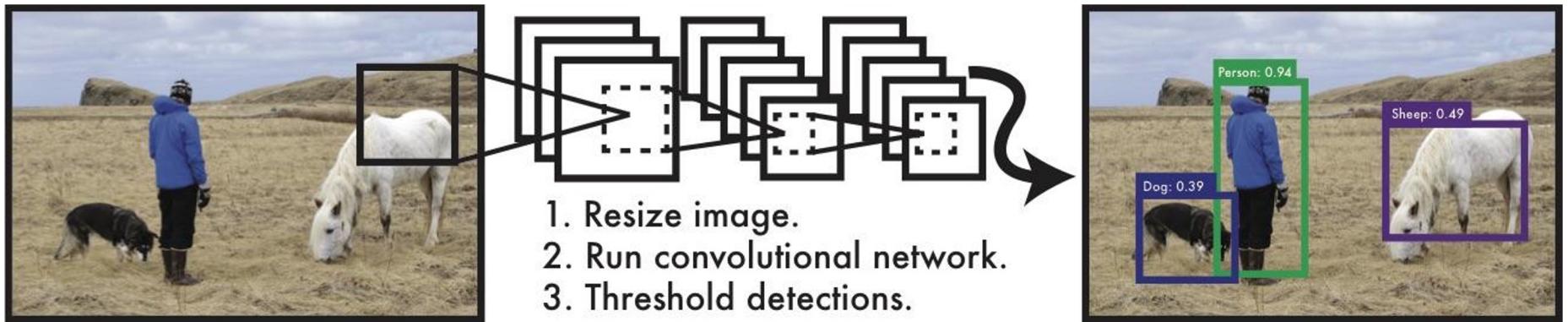
# Sliding window, DPM, R-CNN all train region-based classifiers to perform detection

## DPM: *Deformable Part Models*

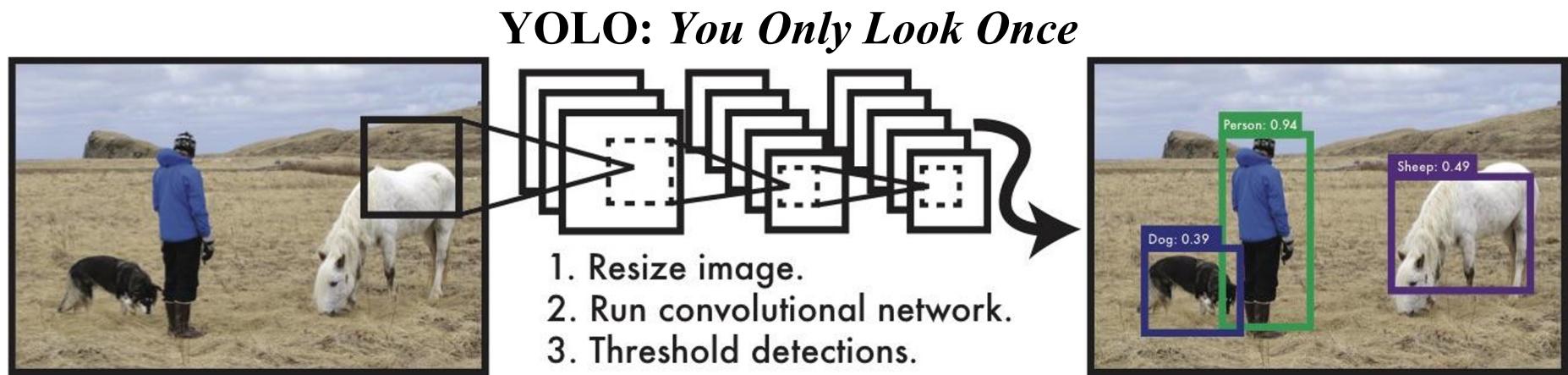


## R-CNN: *Regions with CNN features*



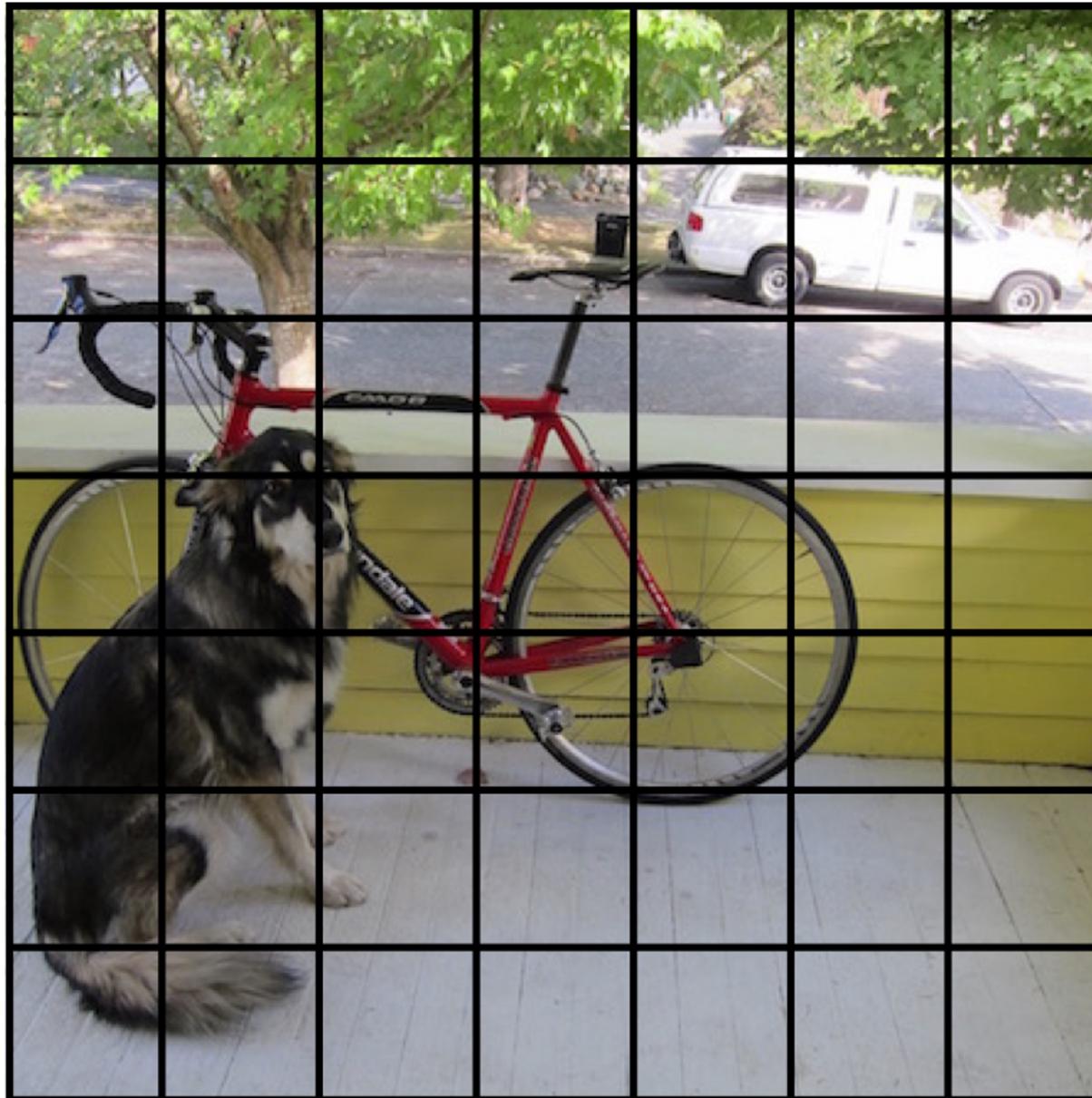


With YOLO, you only look once at an image to perform detection

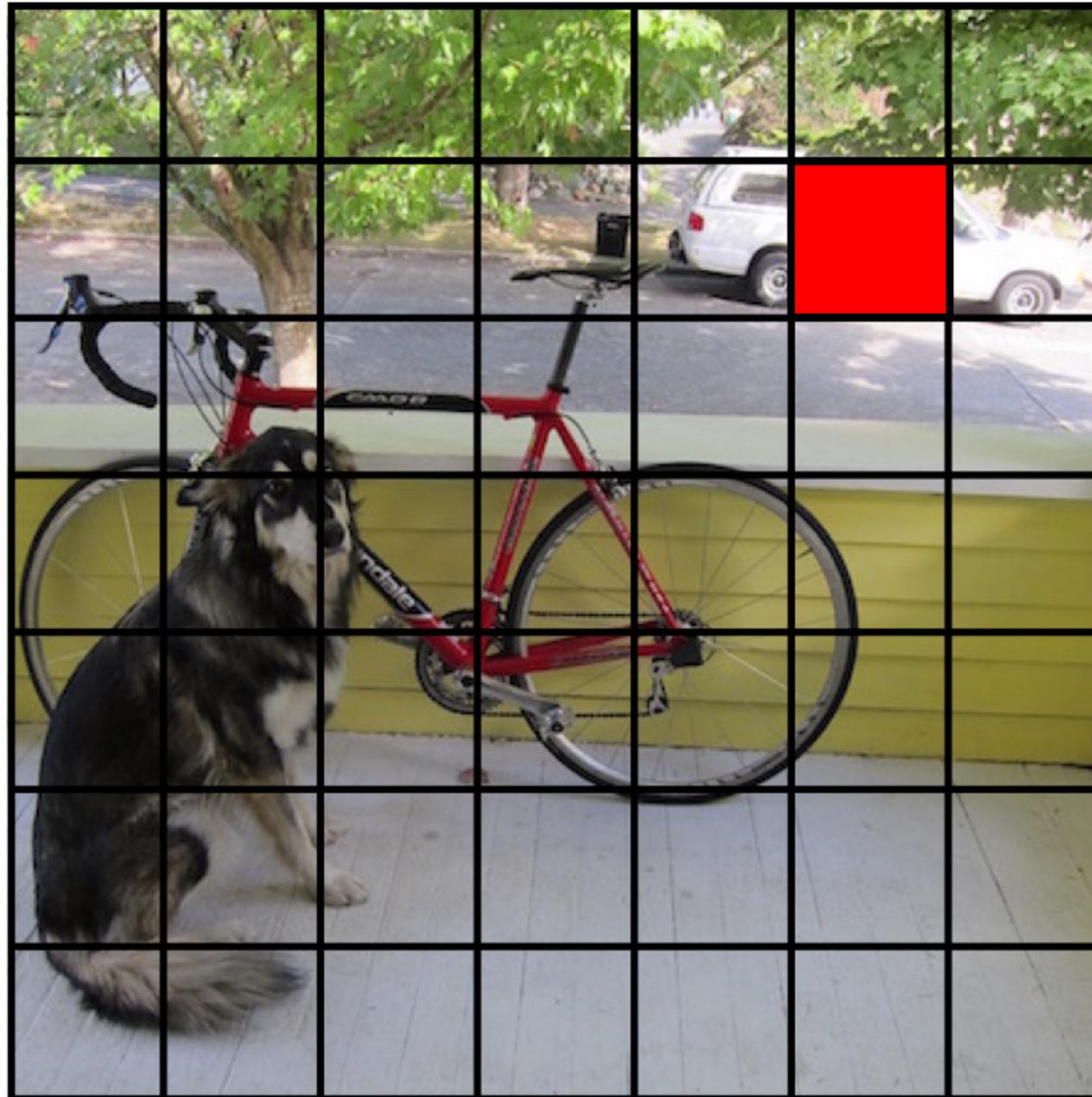




We split the image into a grid



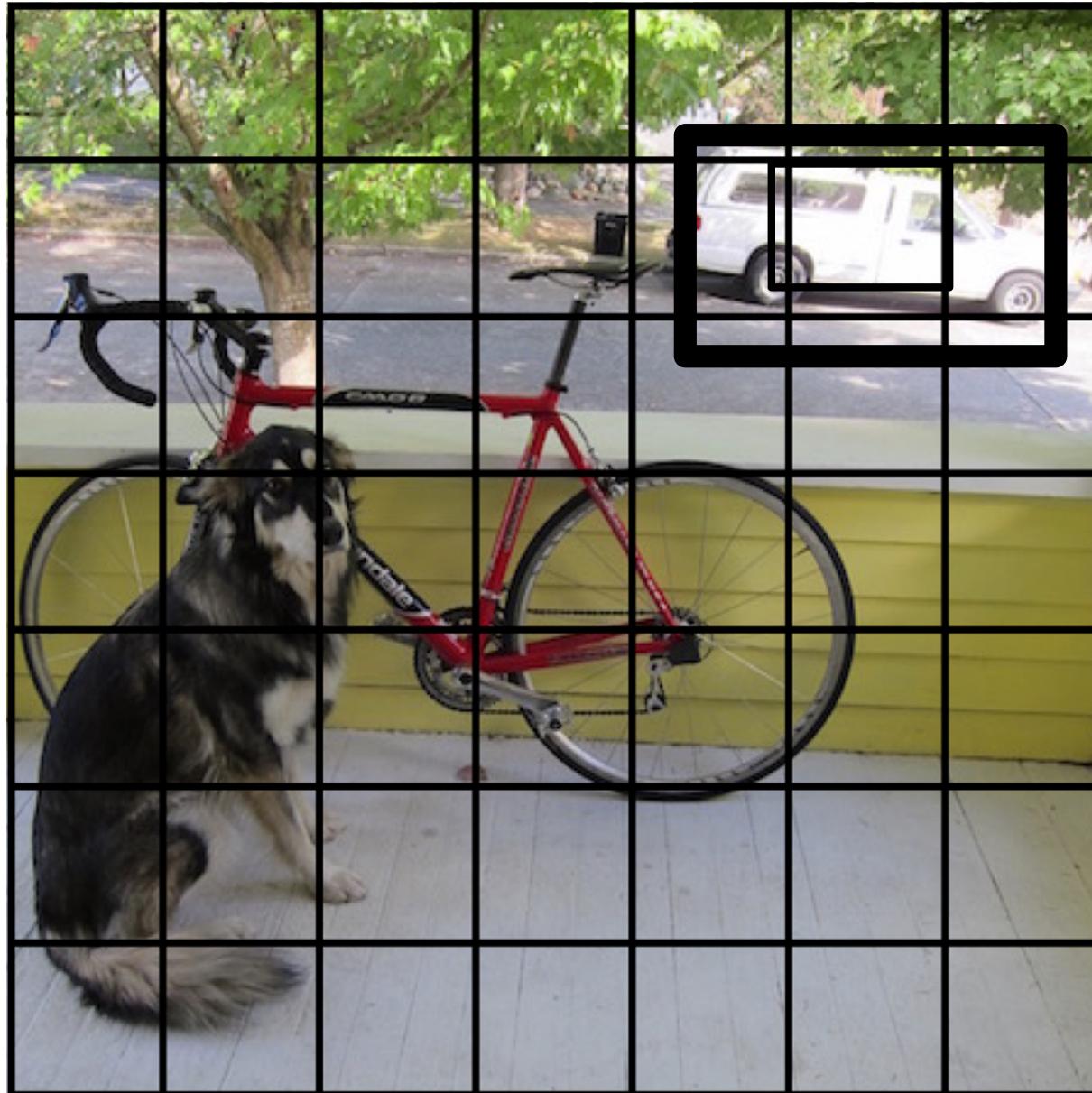
Each cell predicts boxes and confidences:  $P(\text{Object})$



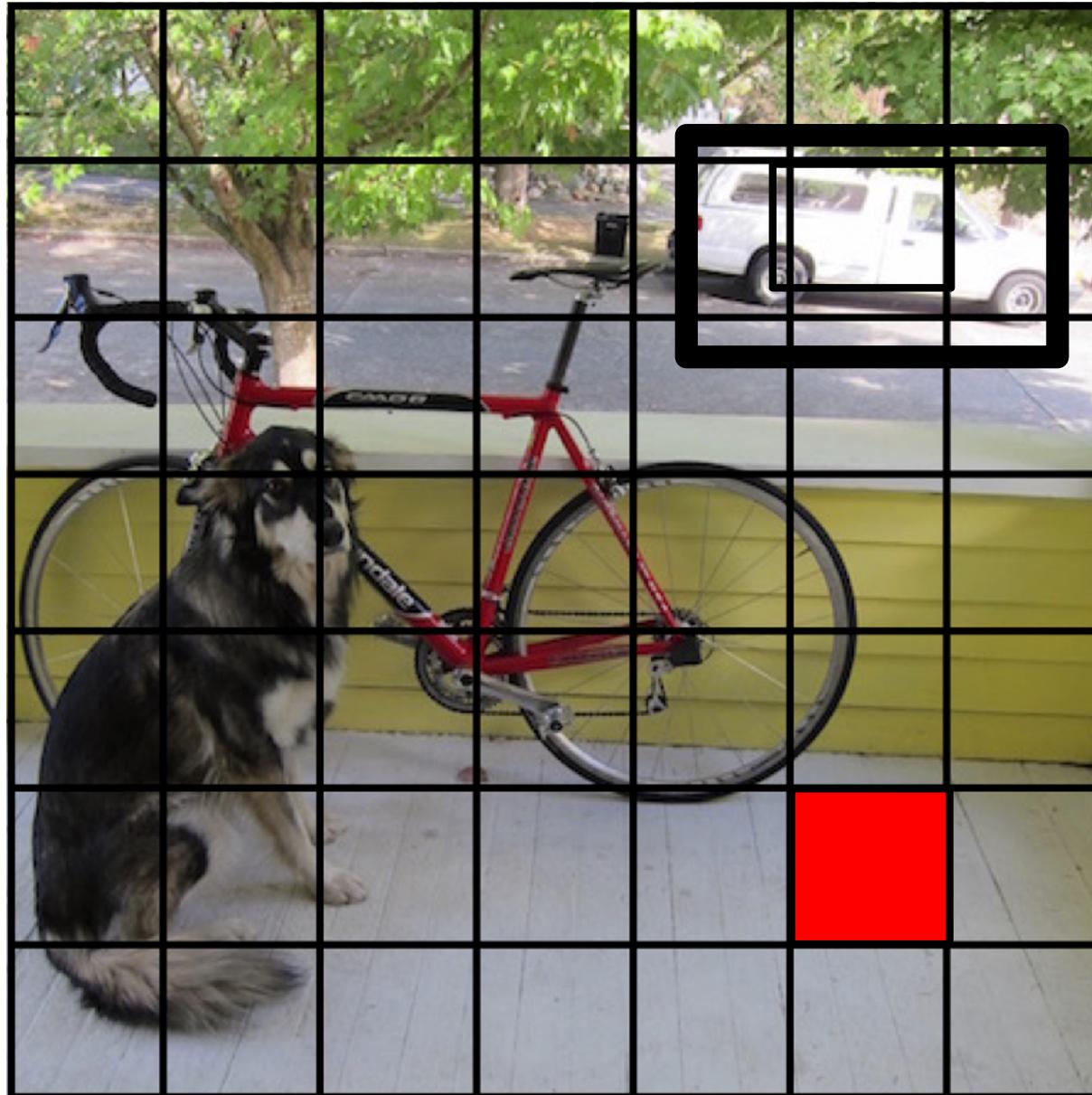
Each cell predicts boxes and confidences:  $P(\text{Object})$



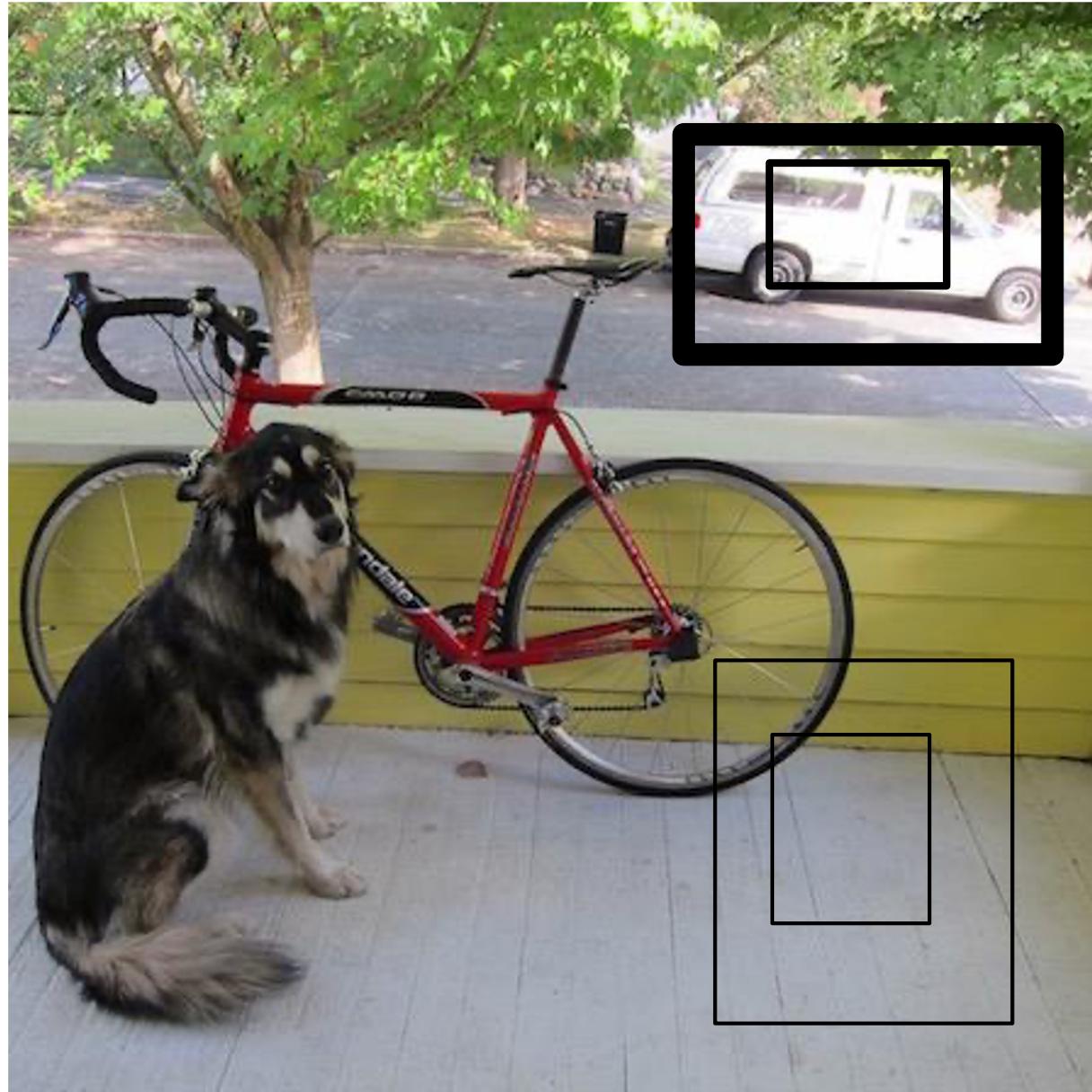
Each cell predicts boxes and confidences:  $P(\text{Object})$



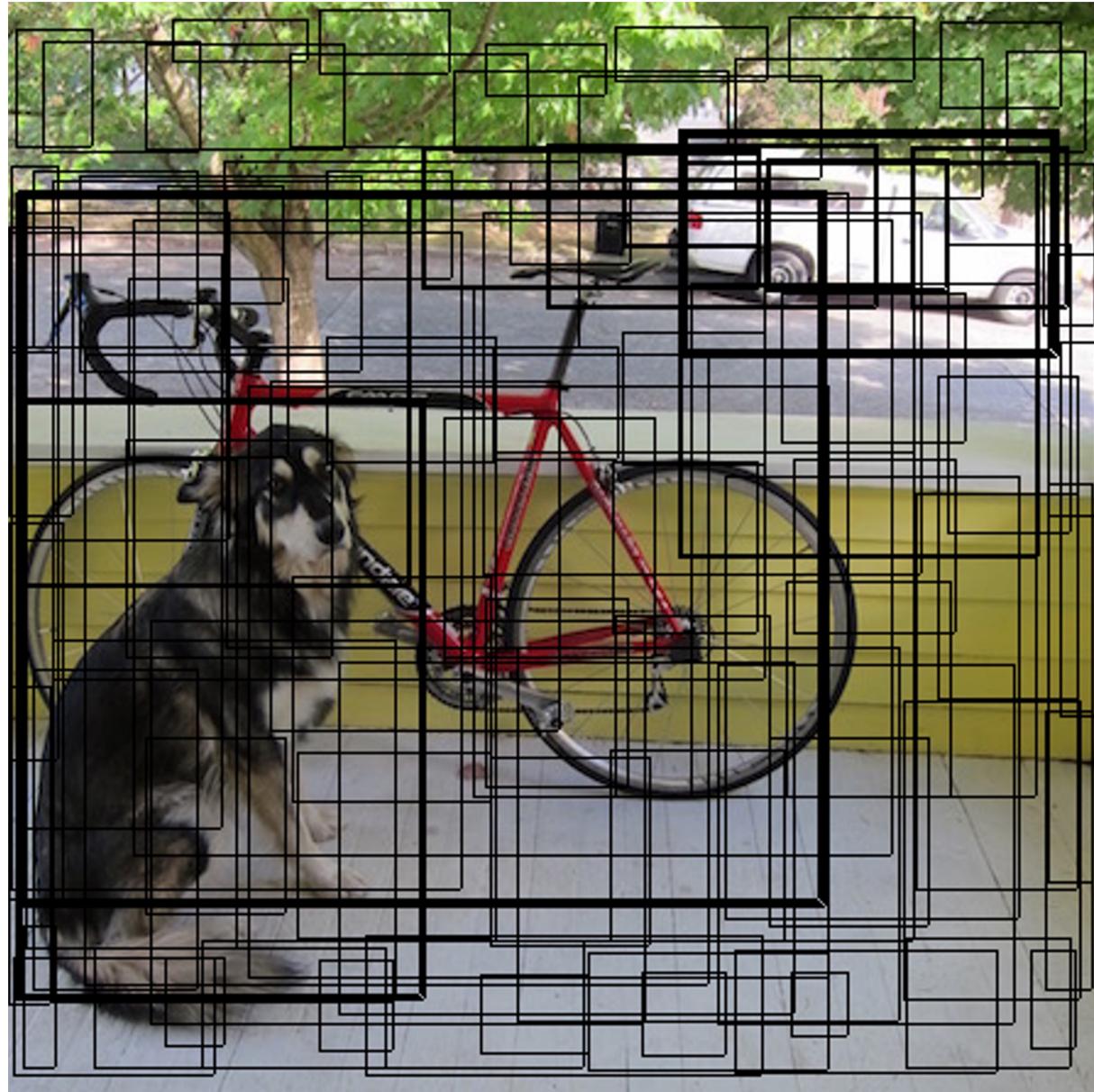
Each cell predicts boxes and confidences:  $P(\text{Object})$



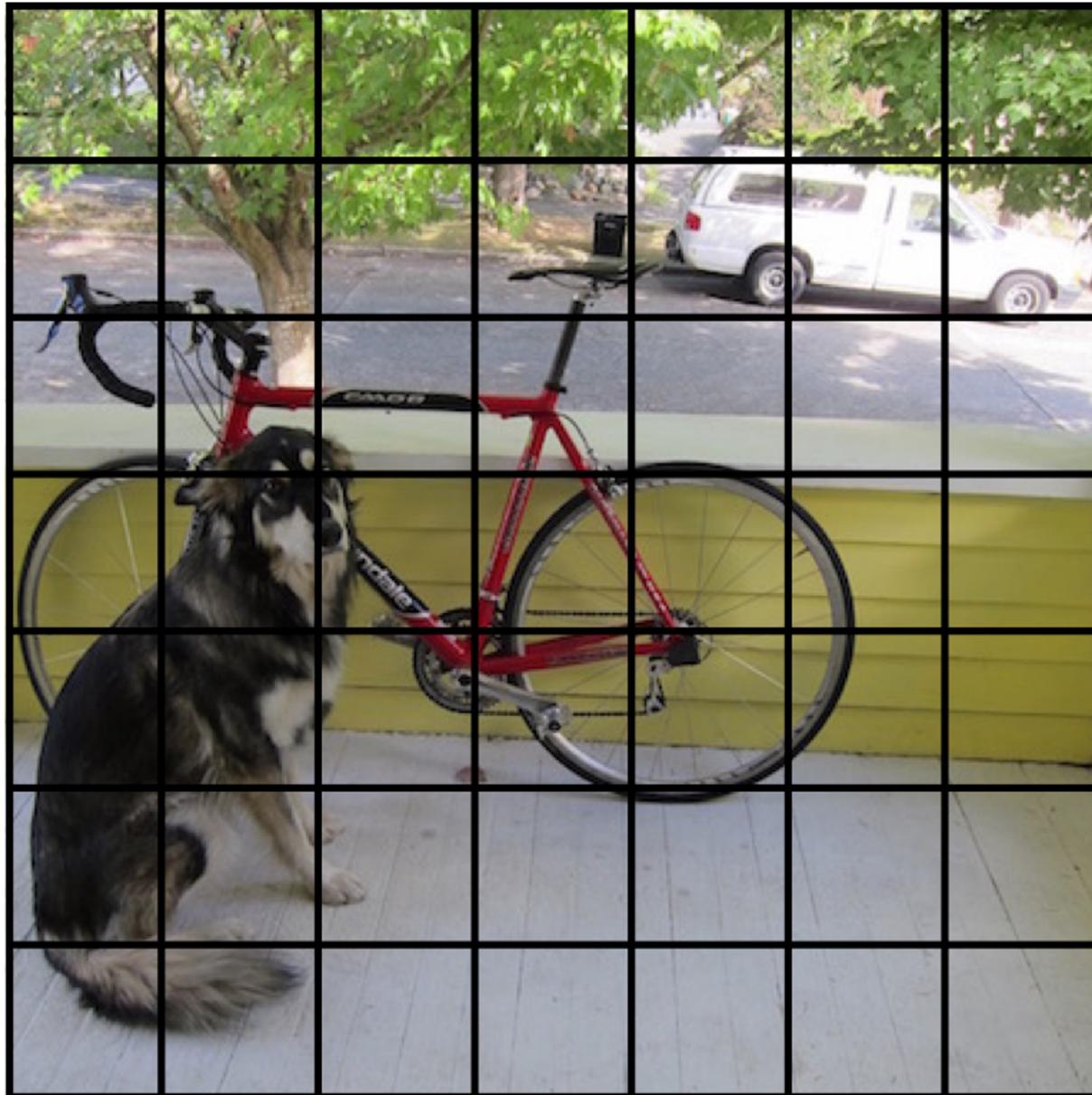
Each cell predicts boxes and confidences:  $P(\text{Object})$



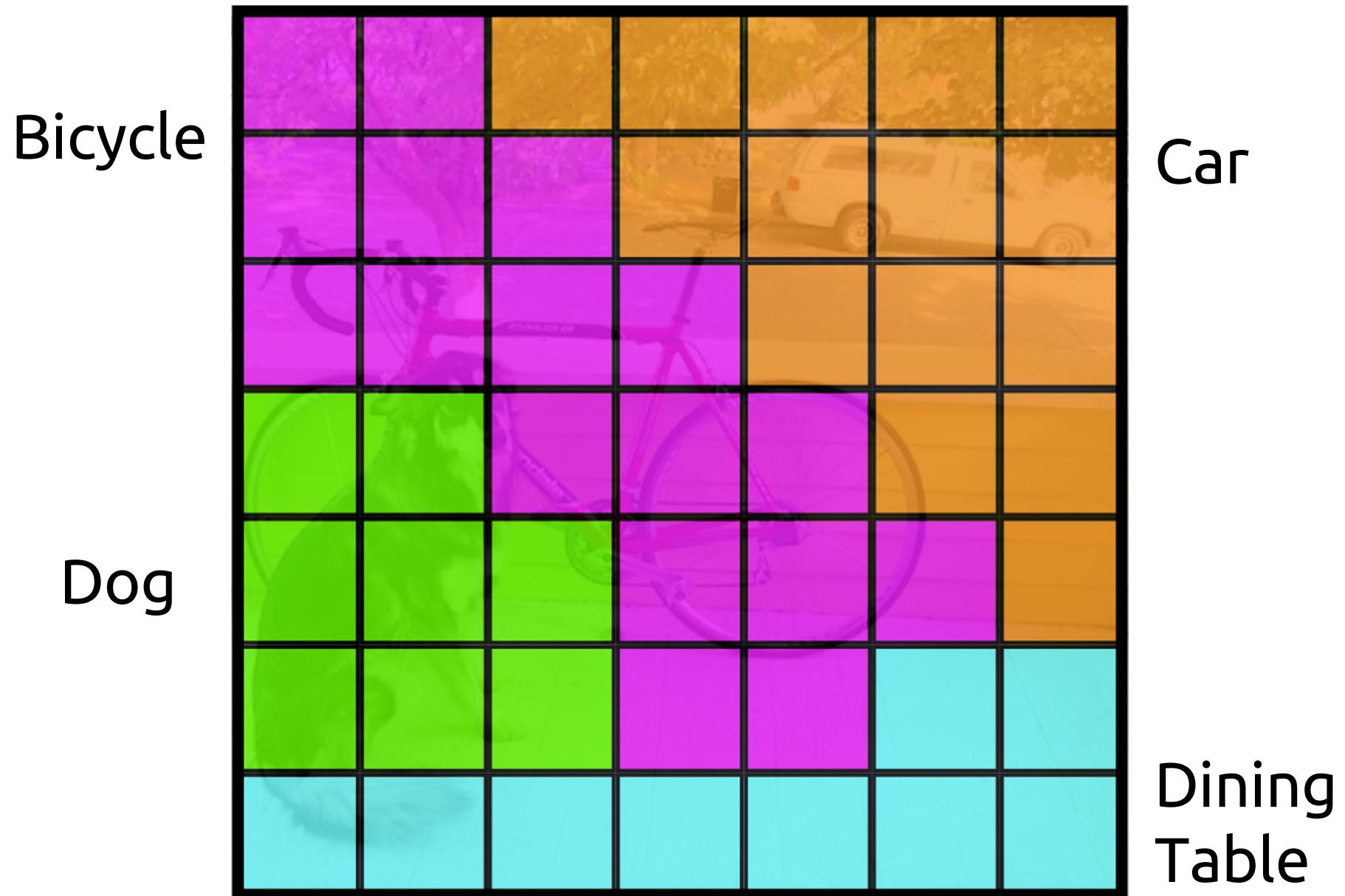
Each cell predicts boxes and confidences:  $P(\text{Object})$



Each cell also predicts a class probability.



Each cell also predicts a class probability.



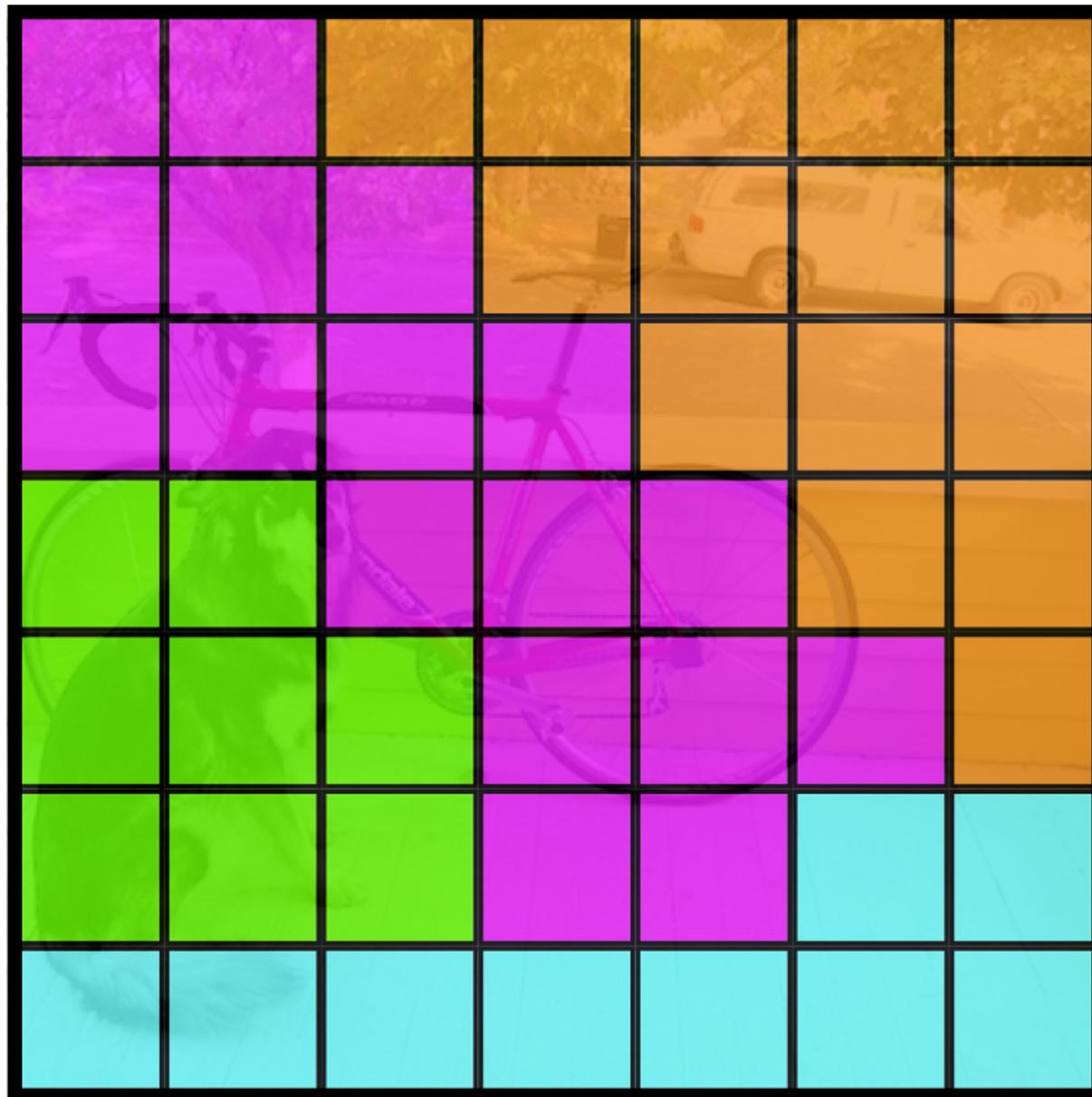
Conditioned on object:  $P(\text{Car} \mid \text{Object})$

Bicycle

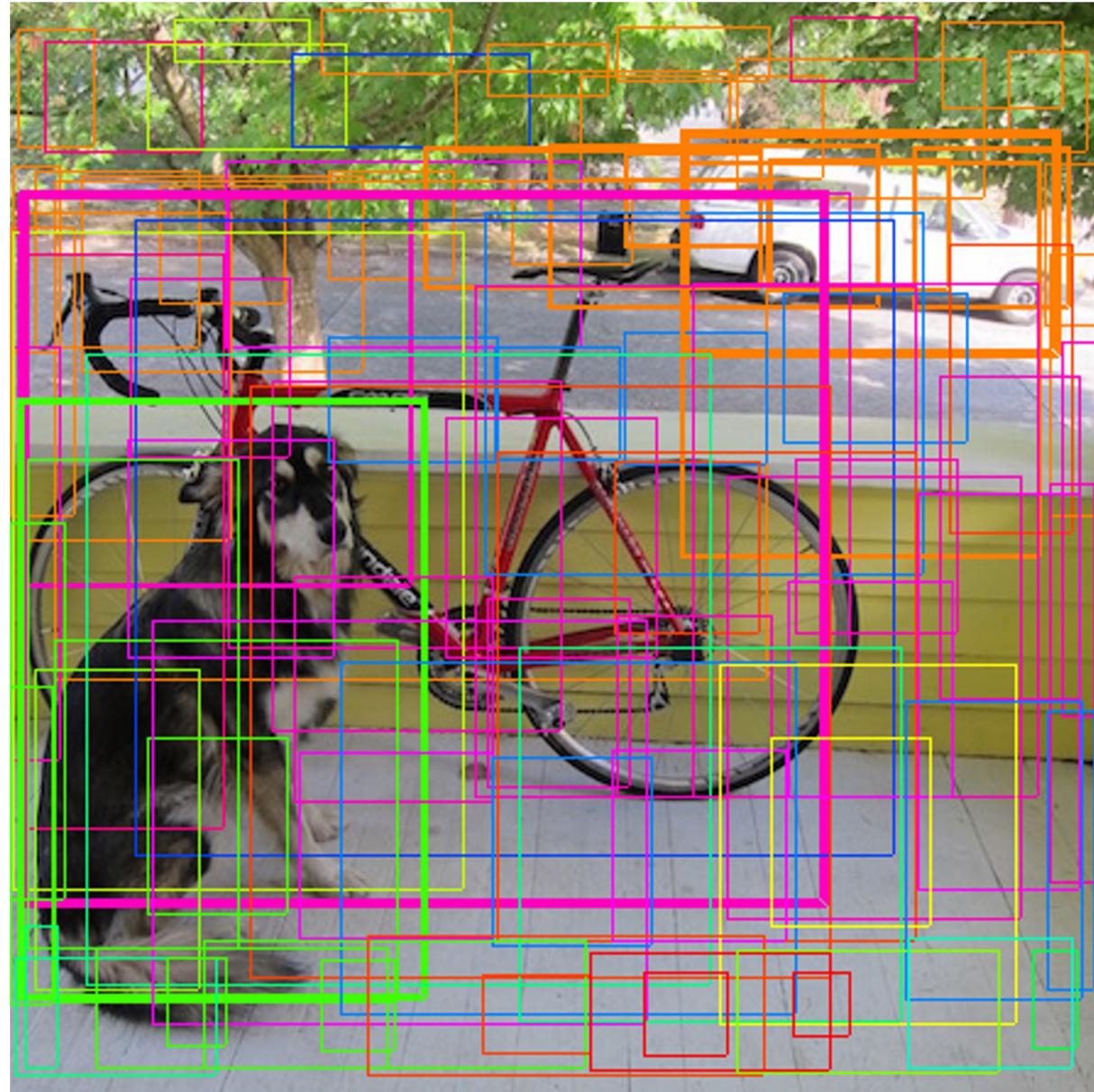
Car

Dog

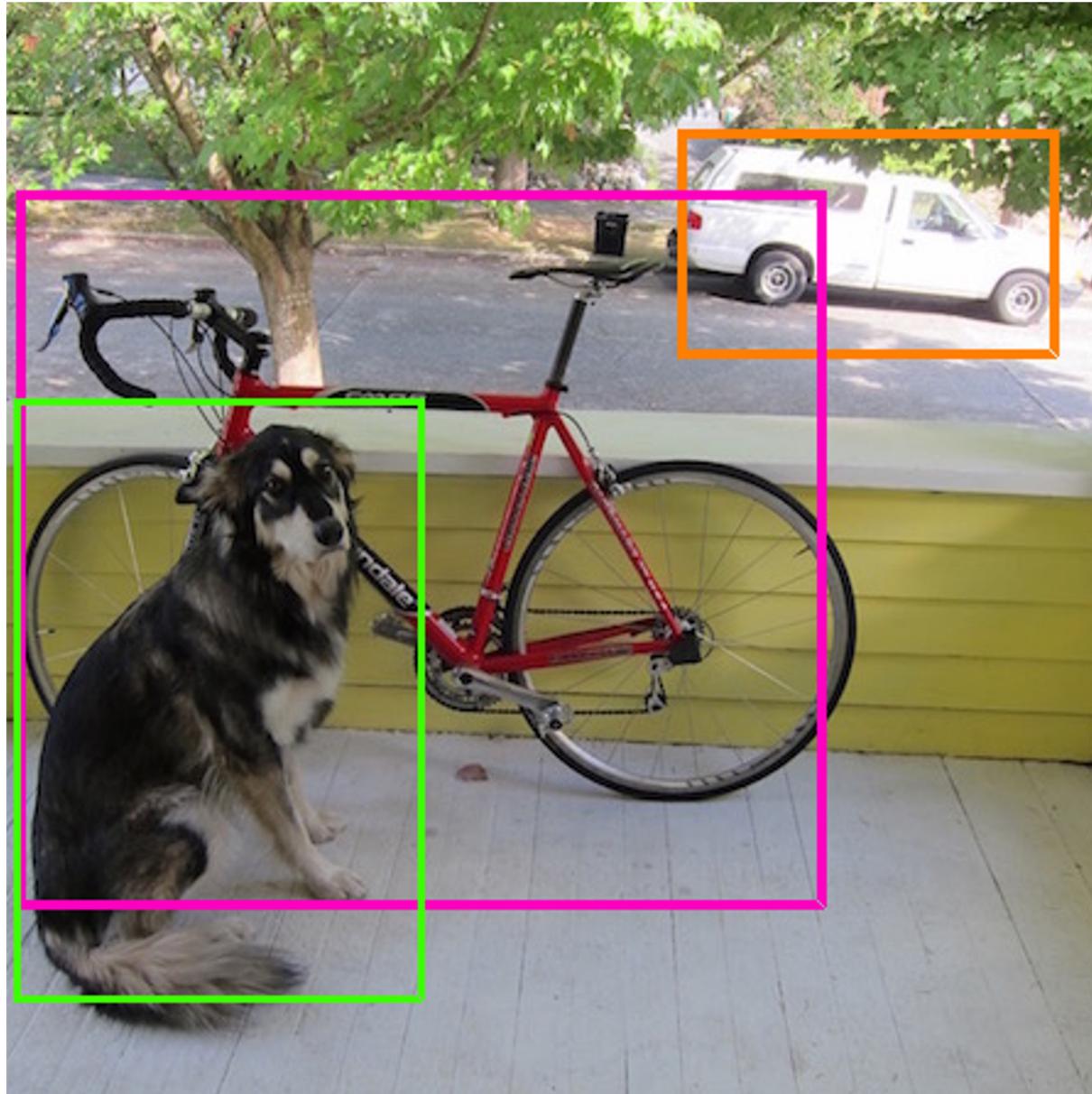
Dining  
Table



Then we combine the box and class predictions.



Finally we do NMS and threshold detections



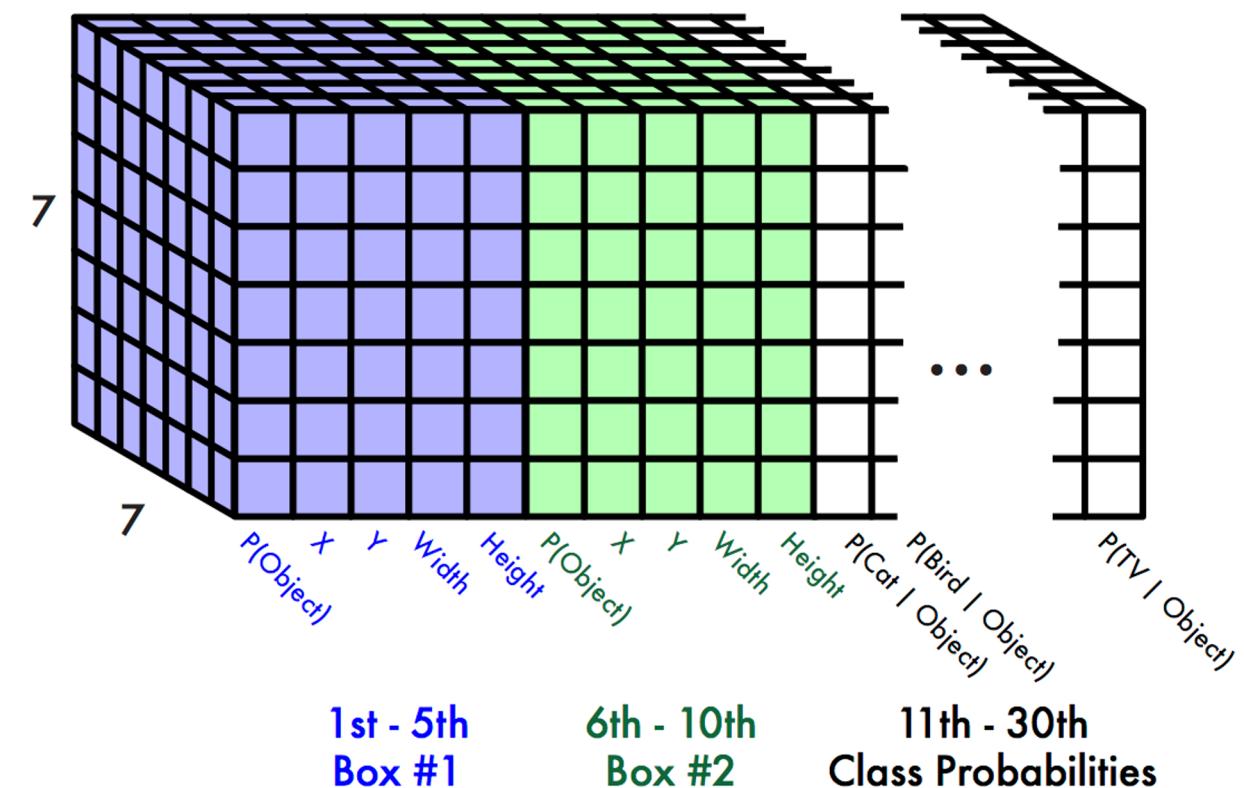
# This parameterization fixes the output size

Each cell predicts:

- For each bounding box:
  - 4 coordinates ( $x, y, w, h$ )
  - 1 confidence value
- Some number of class probabilities

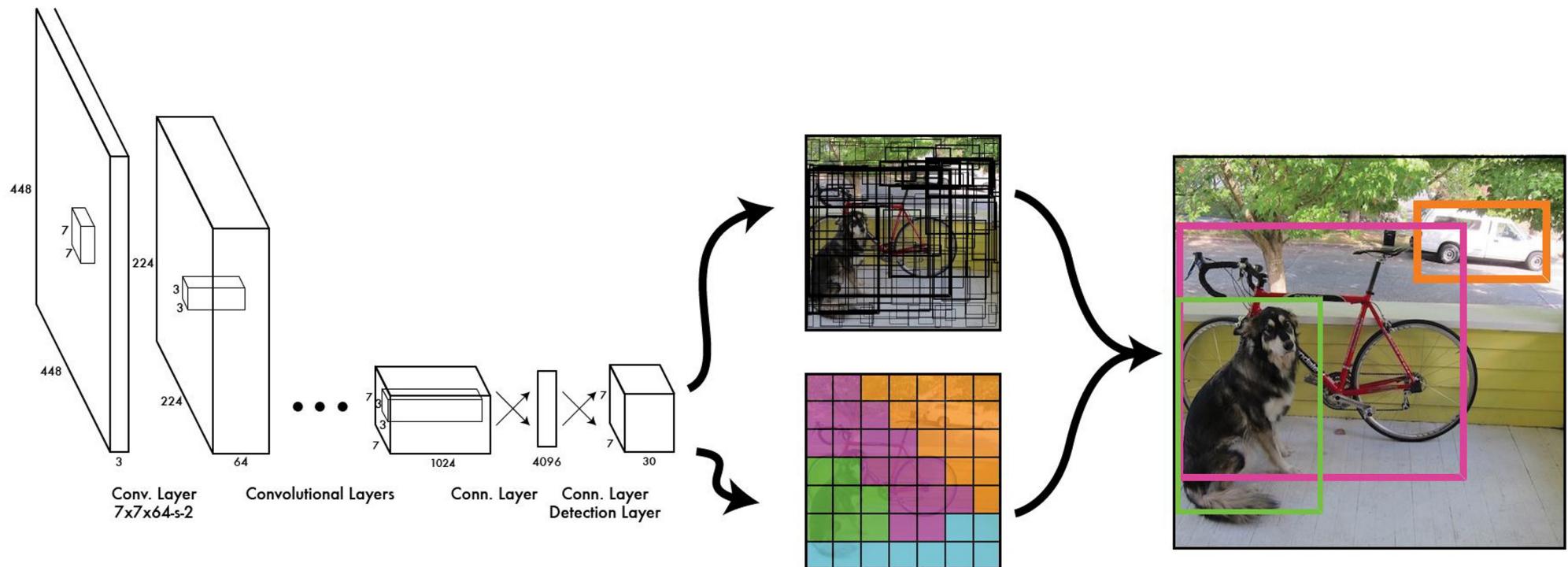
For Pascal VOC:

- $7 \times 7$  grid
- 2 bounding boxes / cell
- 20 classes

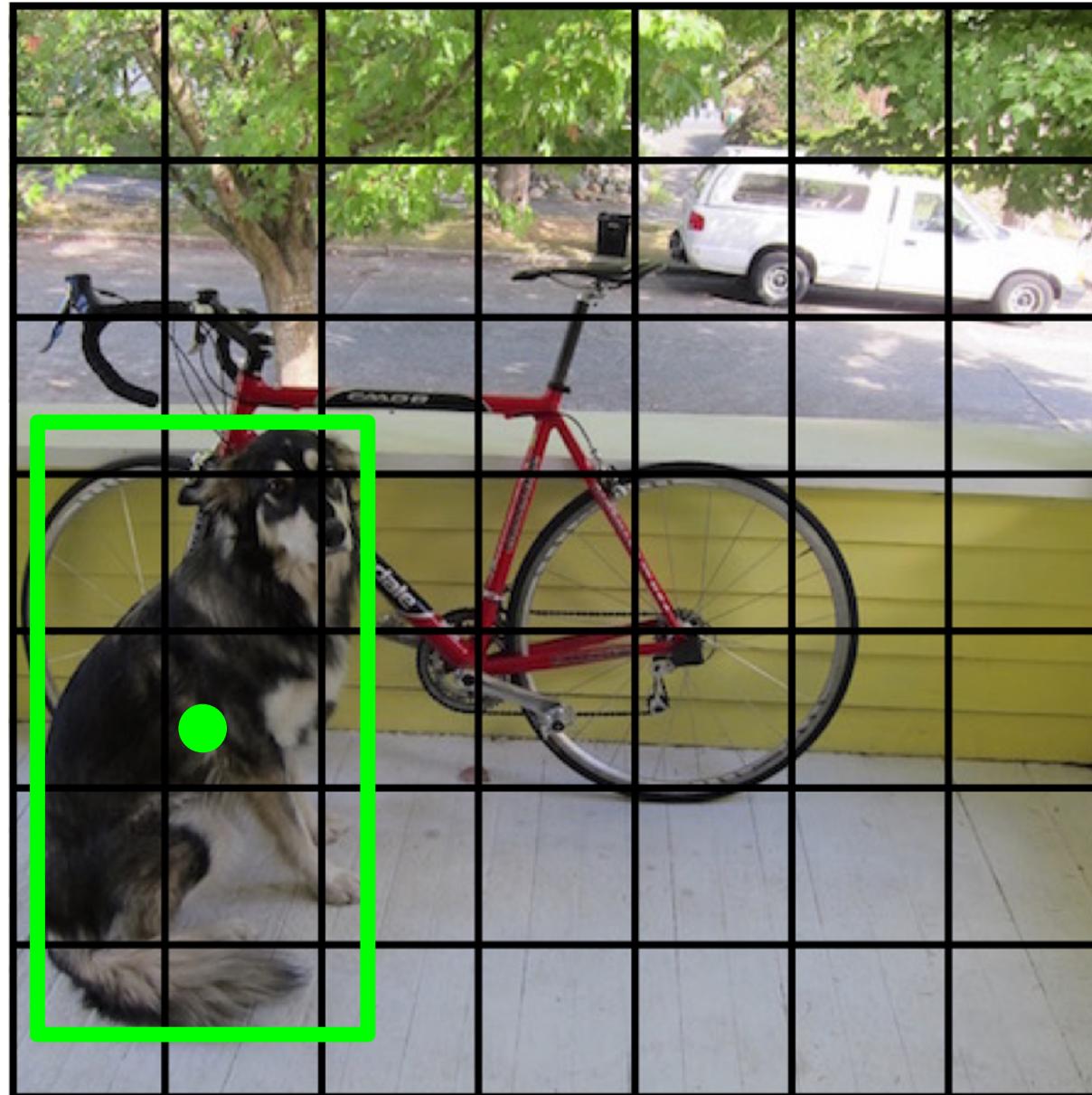


$$7 \times 7 \times (2 \times 5 + 20) = 7 \times 7 \times 30 \text{ tensor} = \mathbf{1470 \text{ outputs}}$$

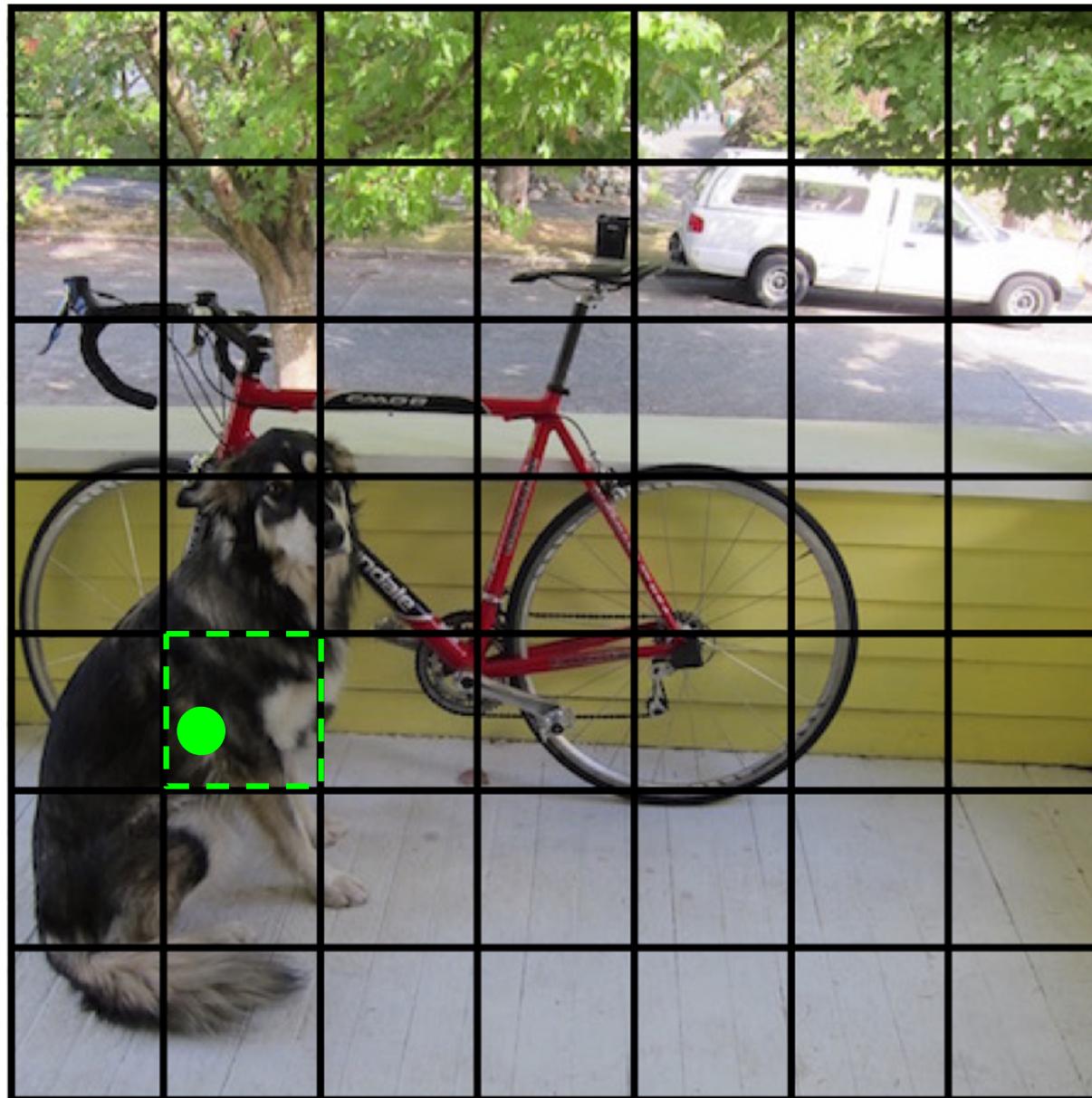
Thus we can train one neural network to be a whole detection pipeline



During training, match example to the right cell



During training, match example to the right cell



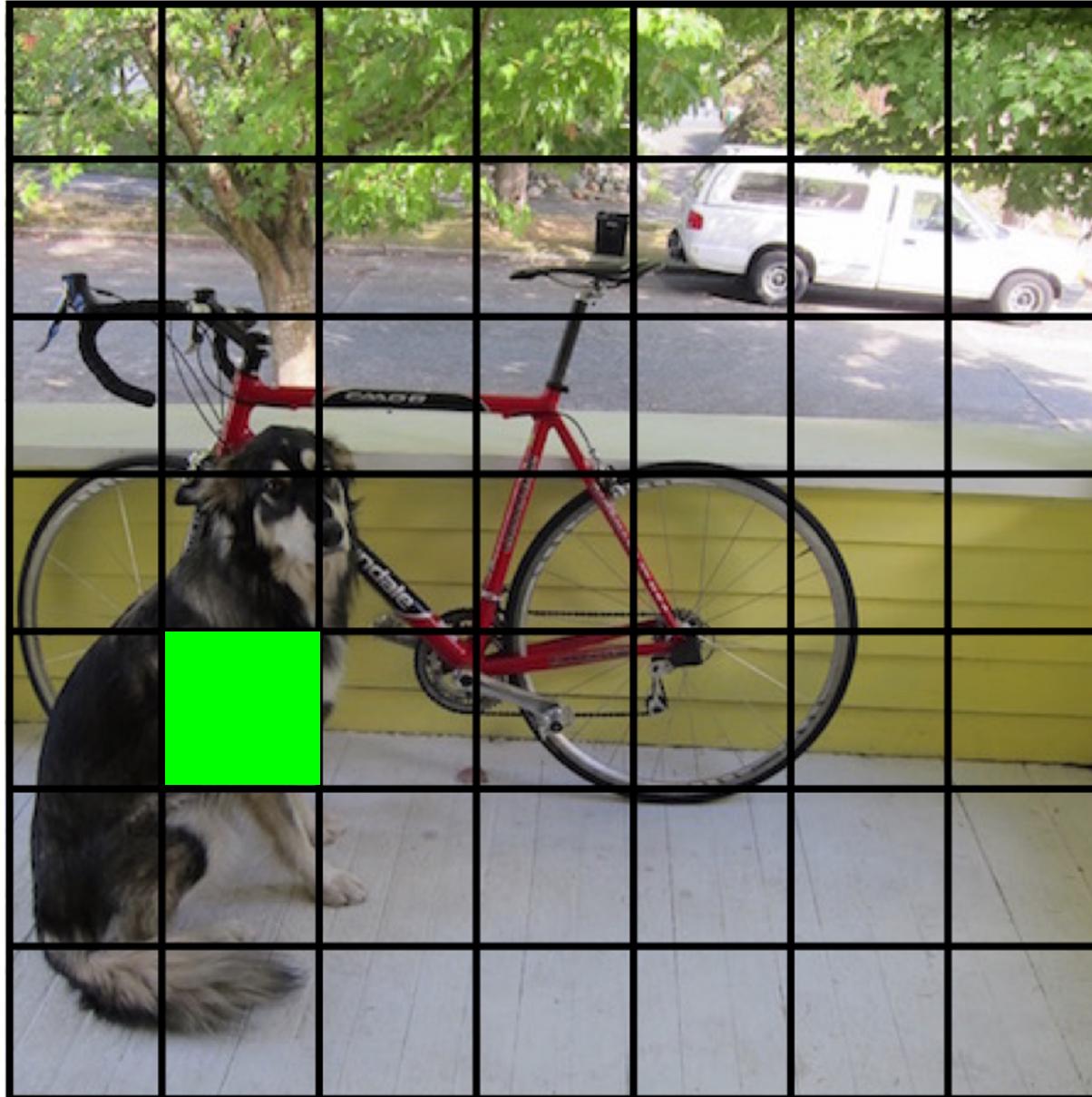
# Adjust that cell's class prediction

**Dog = 1**

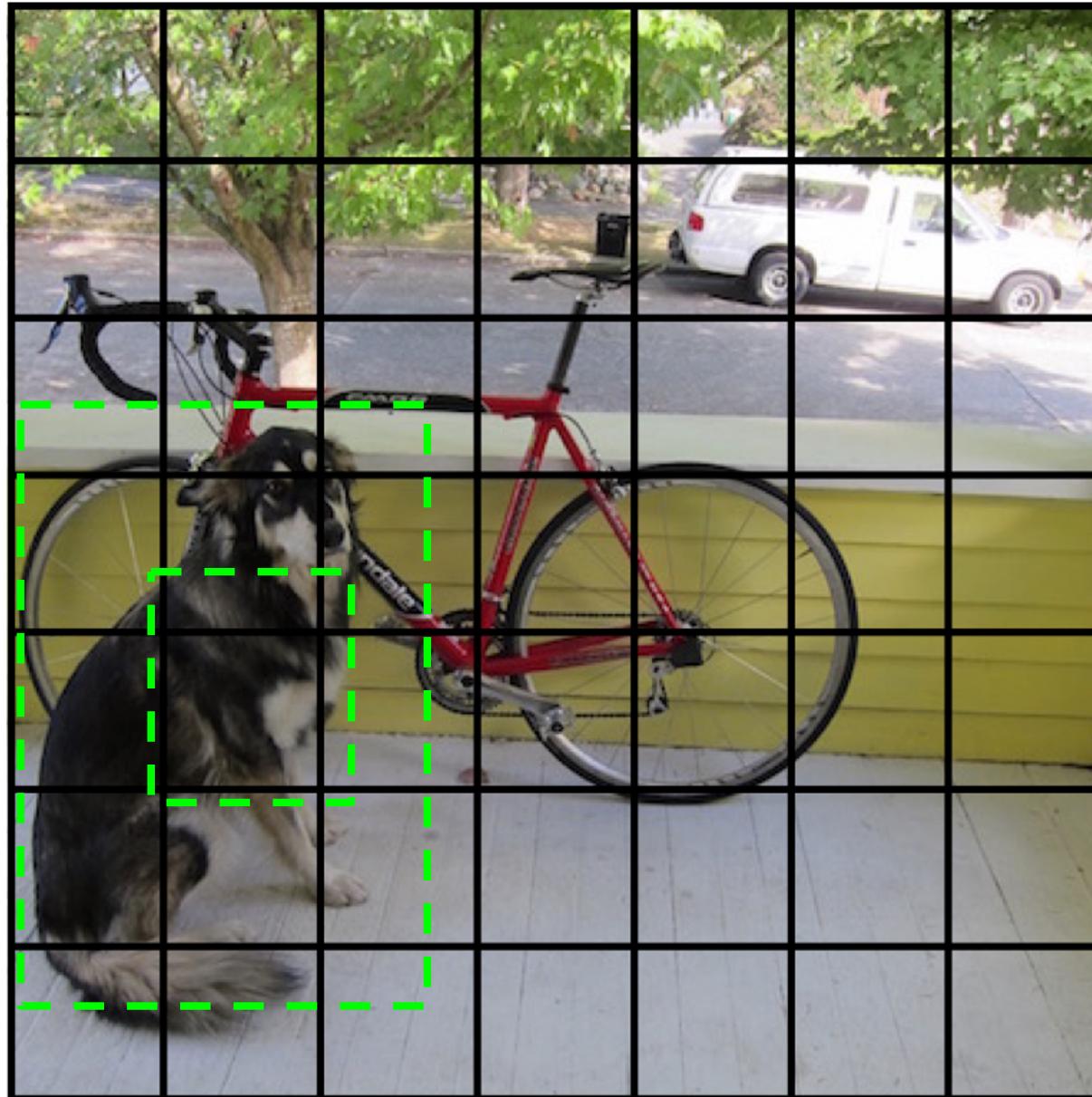
Cat = 0

Bike = 0

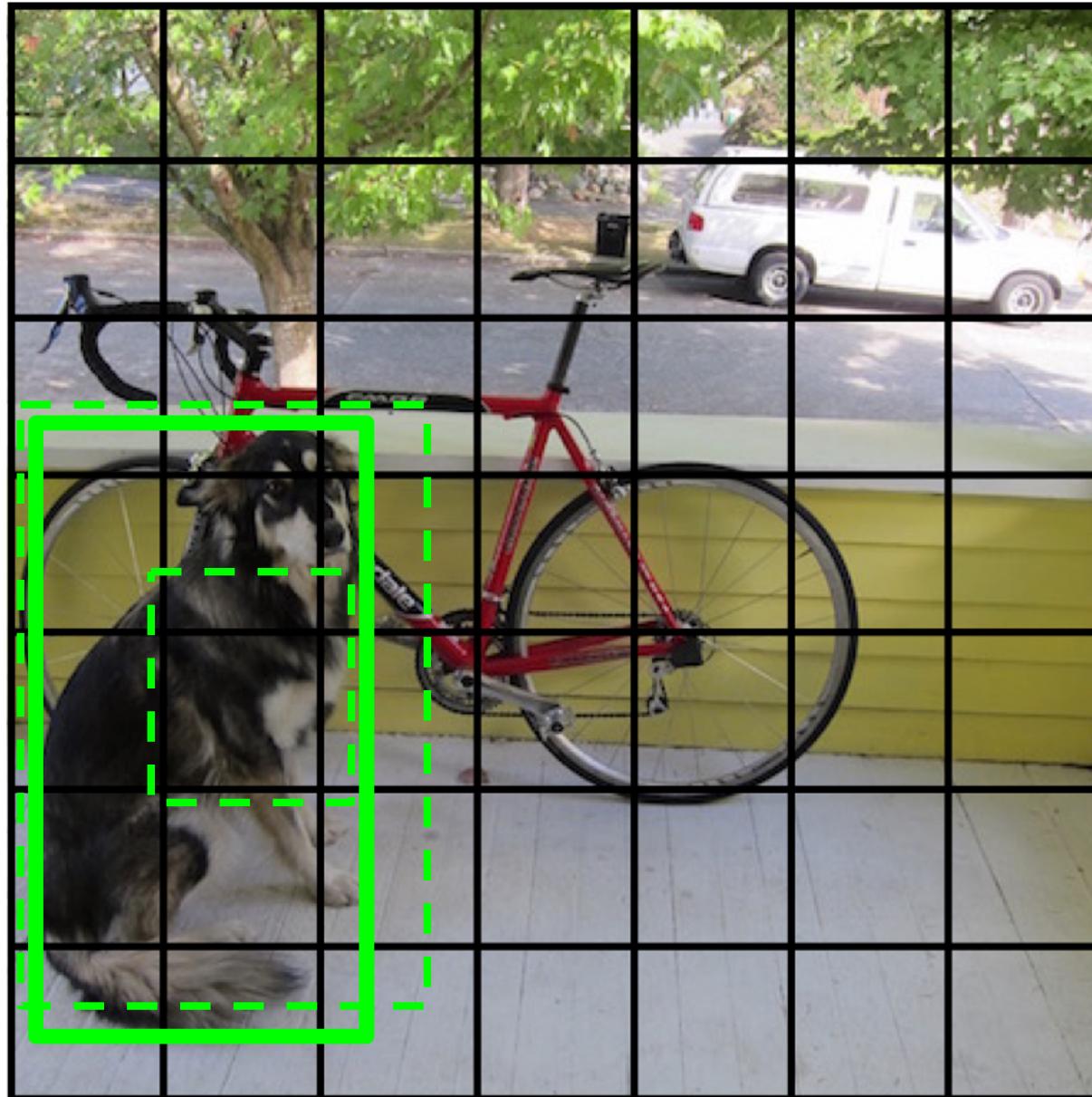
...



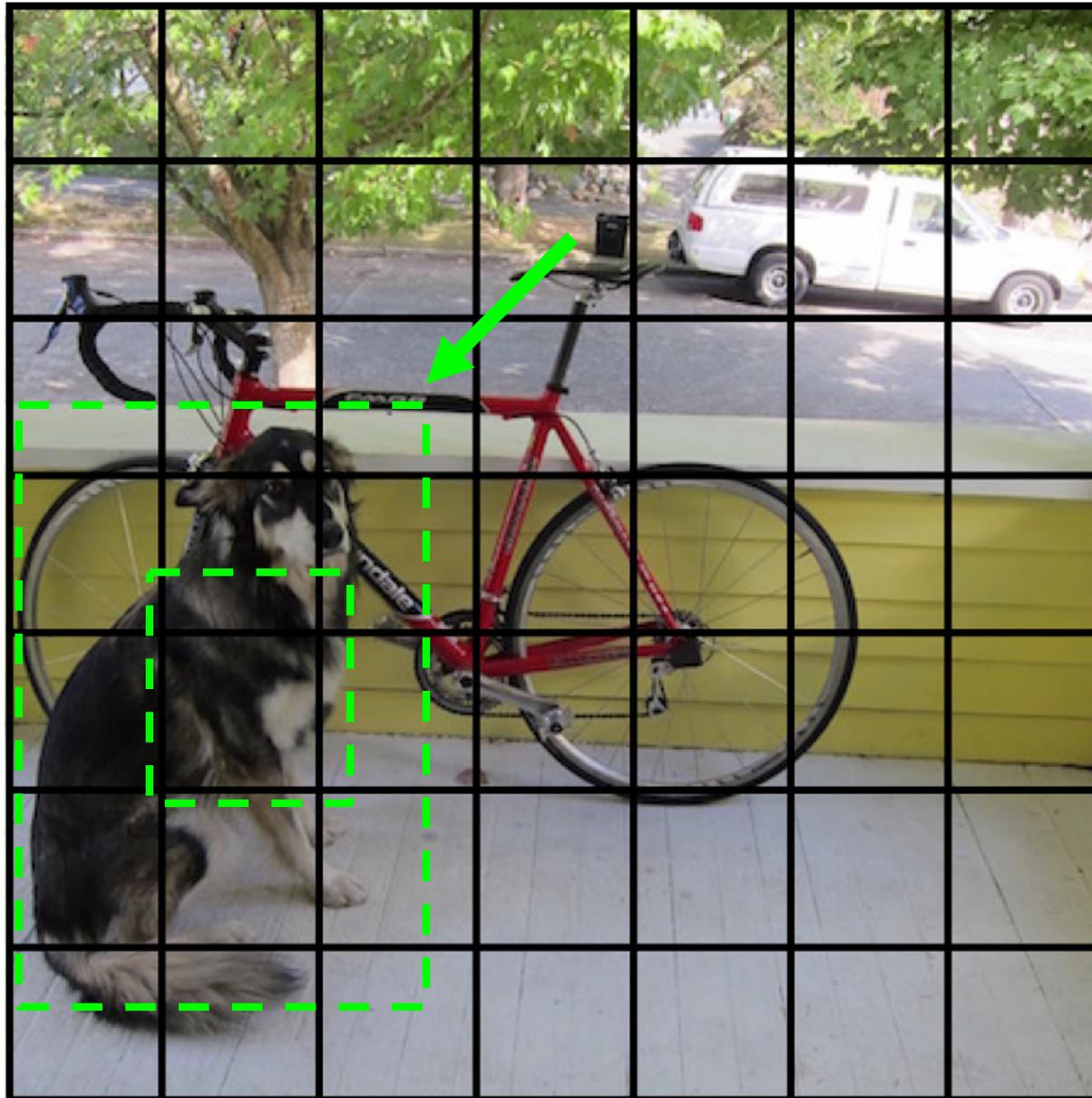
Look at that cell's predicted boxes



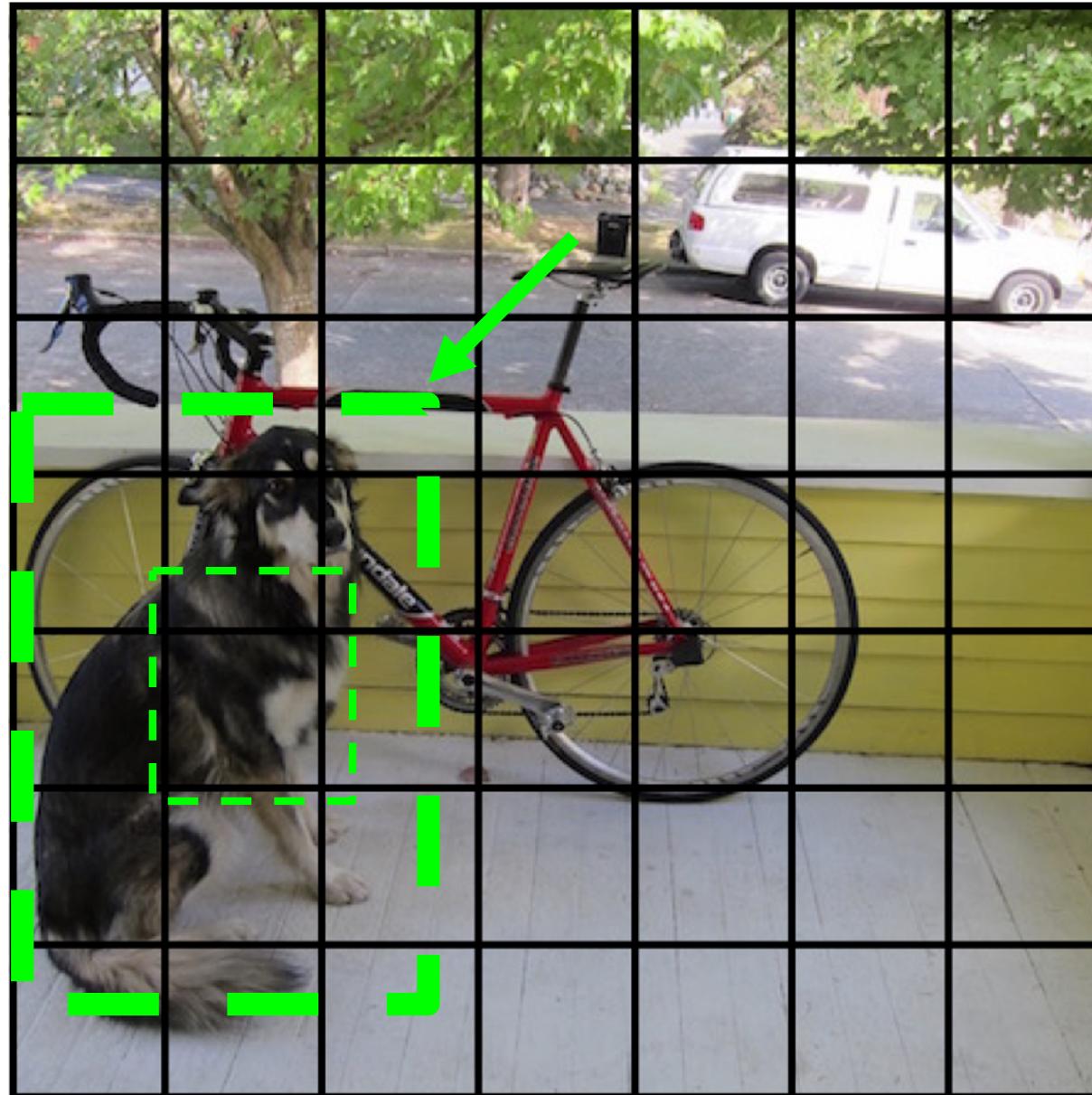
Find the best one, adjust it, increase the confidence



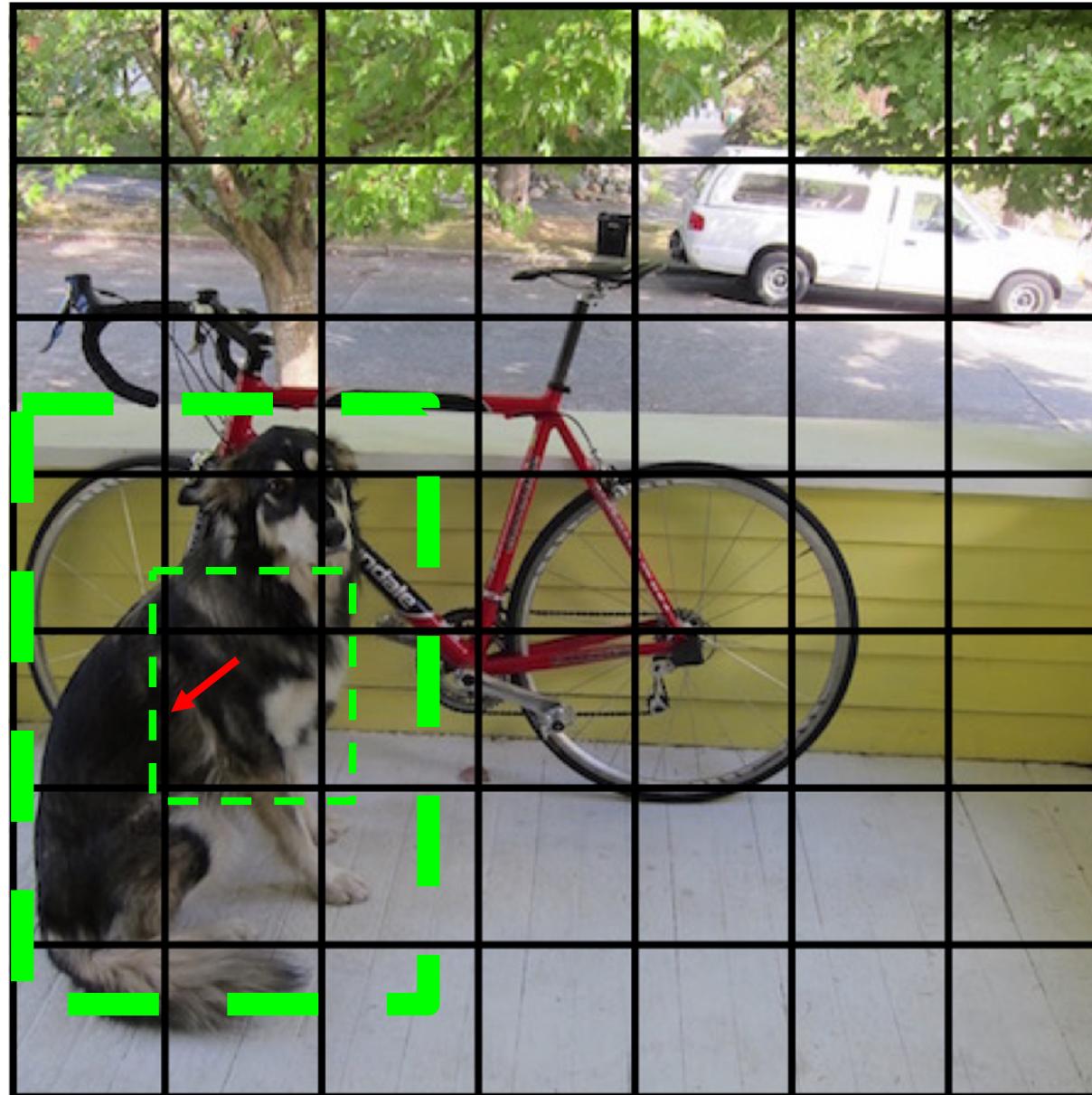
Find the best one, adjust it, increase the confidence



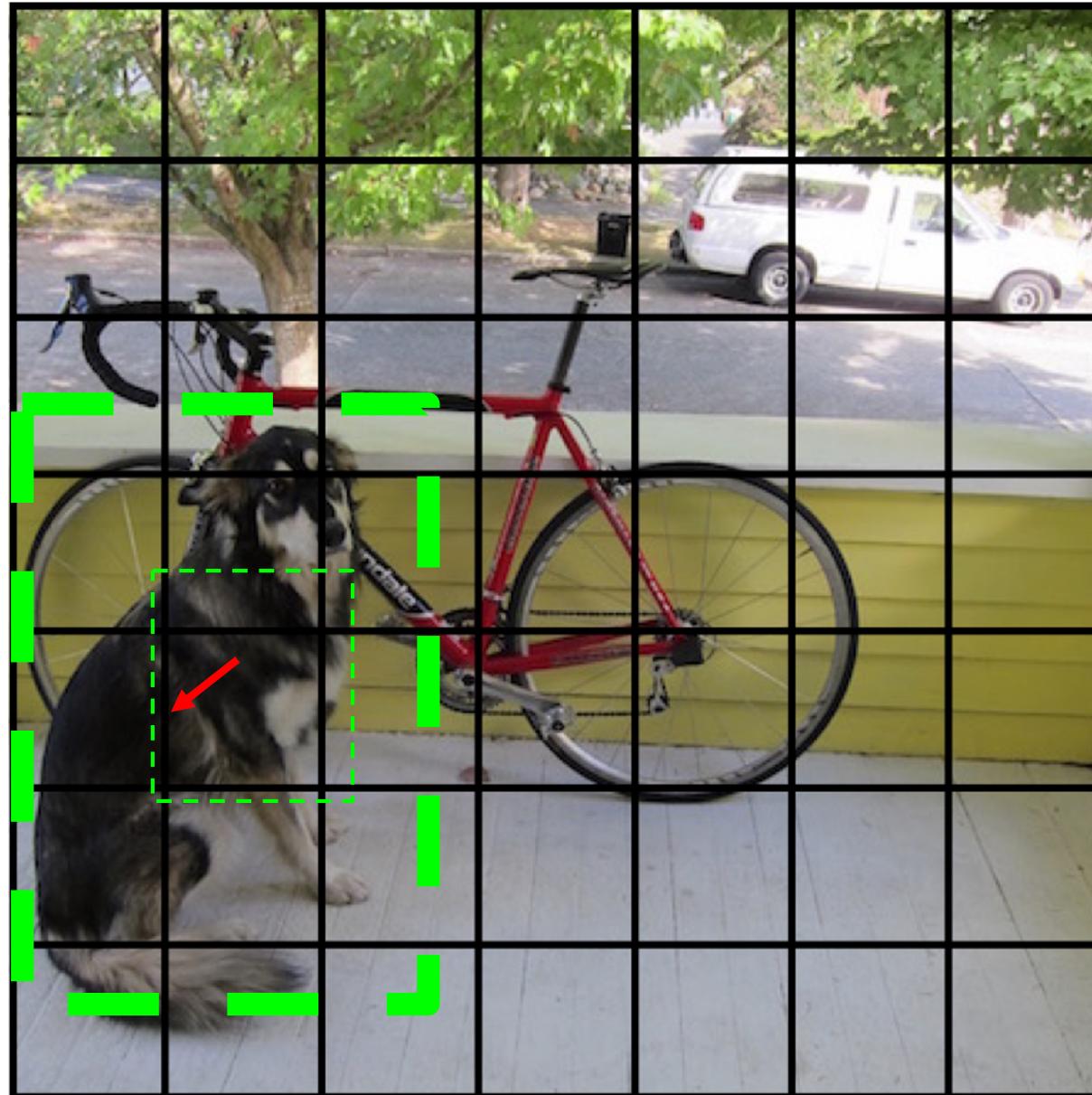
Find the best one, adjust it, increase the confidence



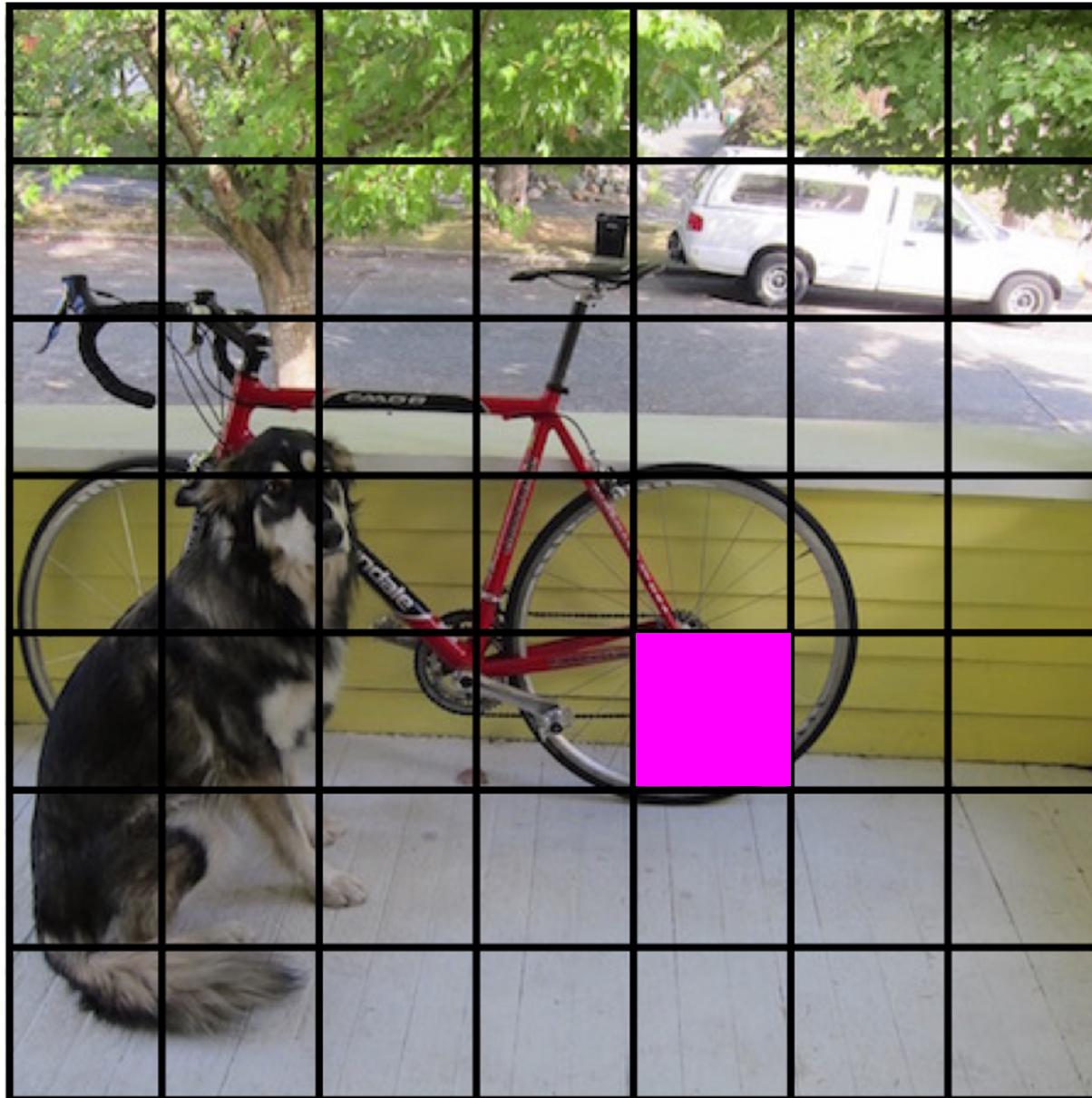
# Decrease the confidence of other boxes



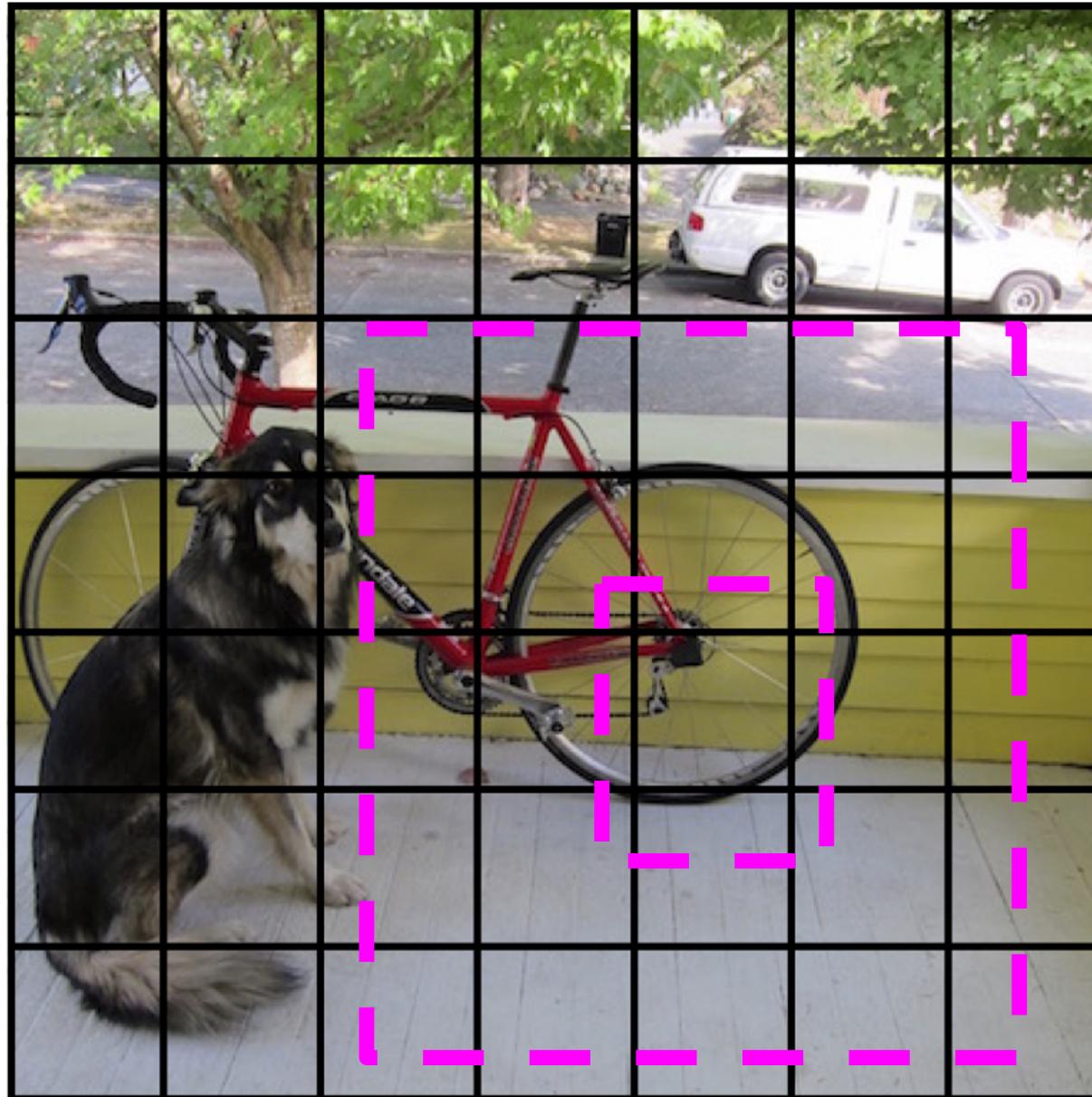
# Decrease the confidence of other boxes



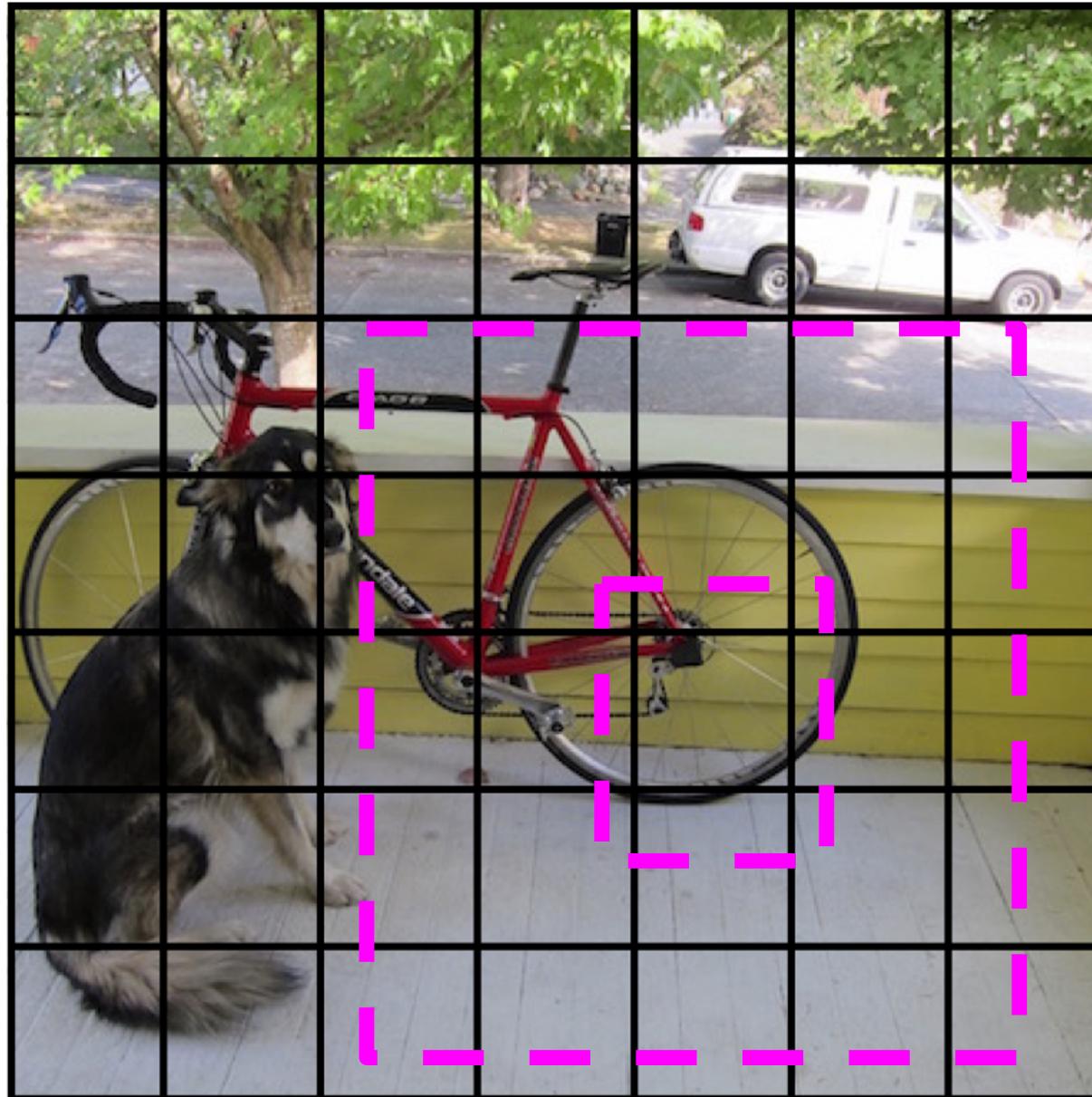
Some cells don't have any ground truth detections!



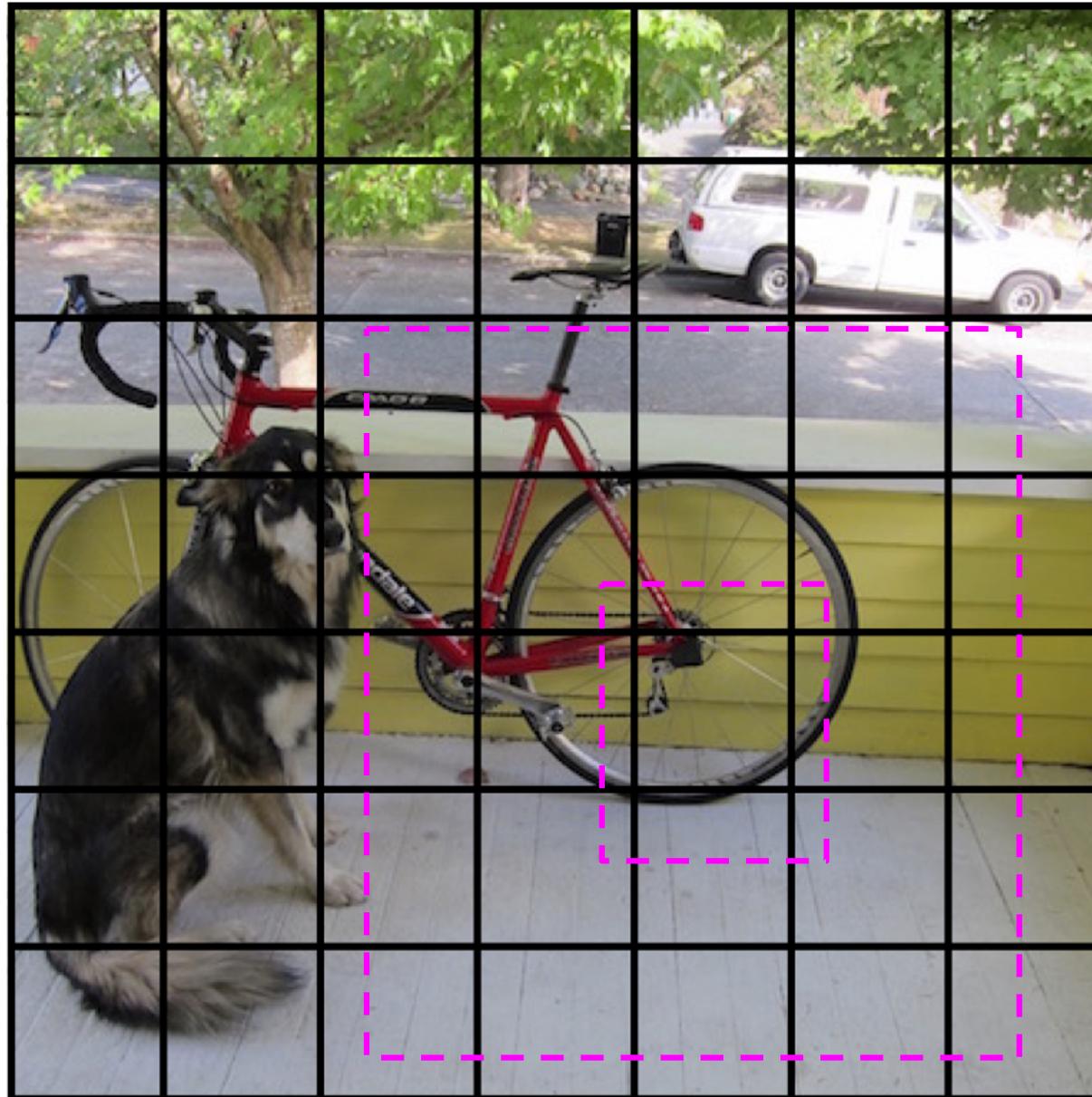
Some cells don't have any ground truth detections!



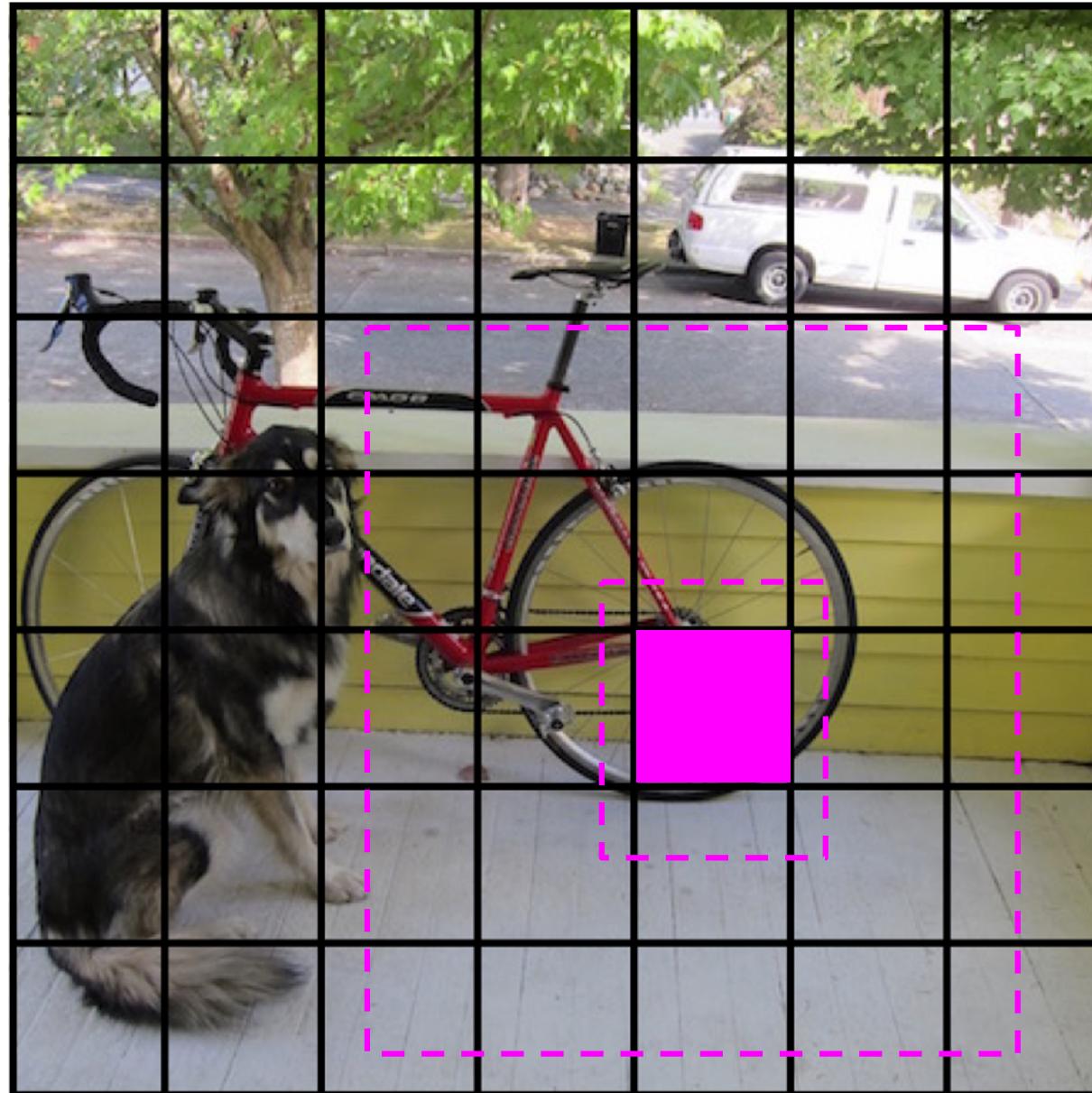
Decrease the confidence of these boxes



Decrease the confidence of these boxes

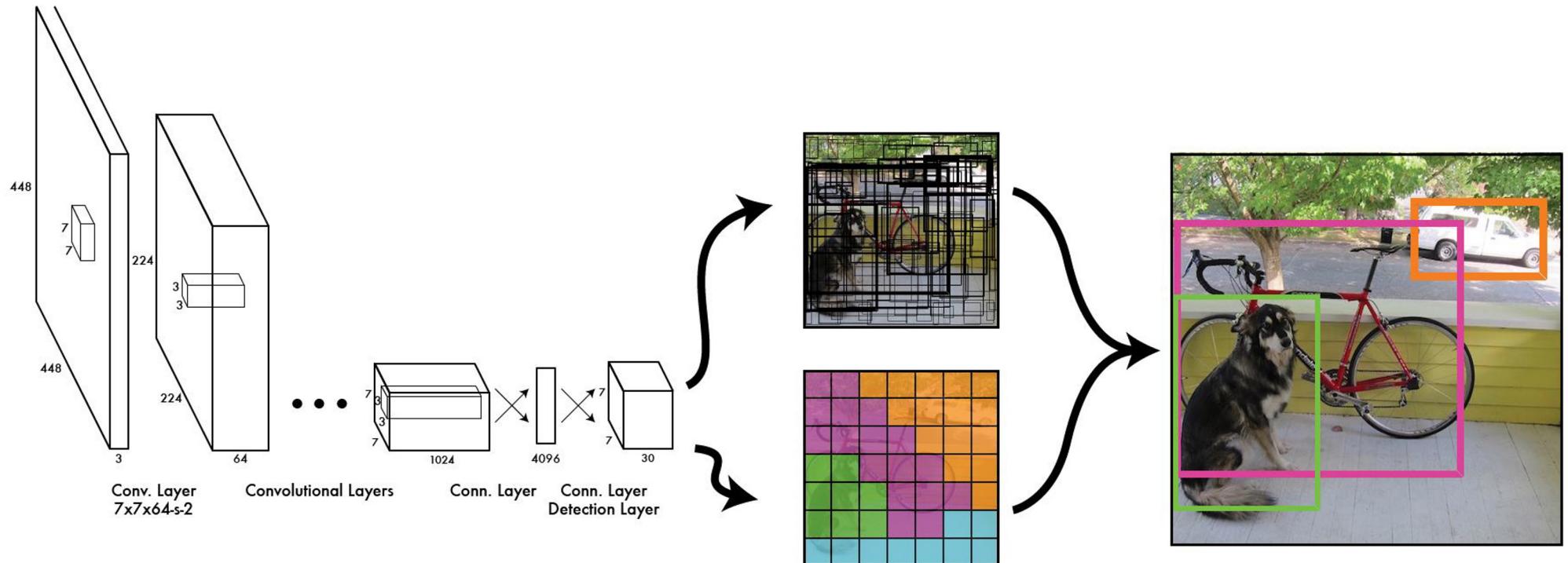


Don't adjust the class probabilities or coordinates

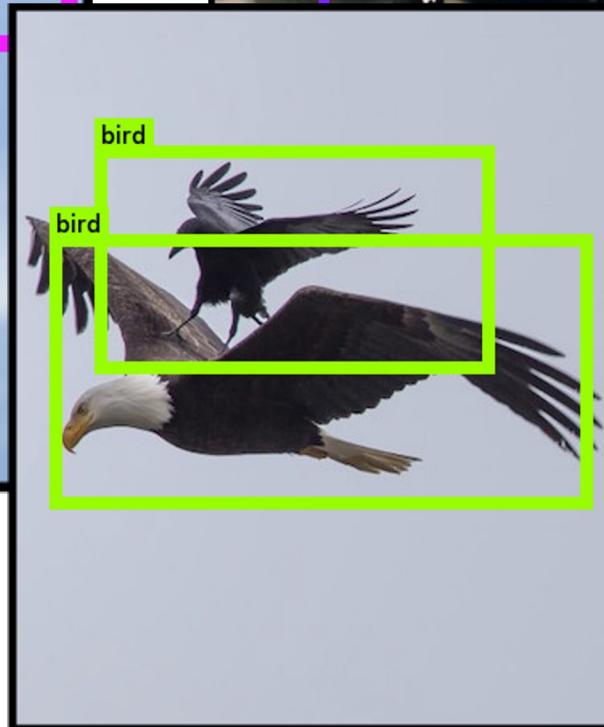
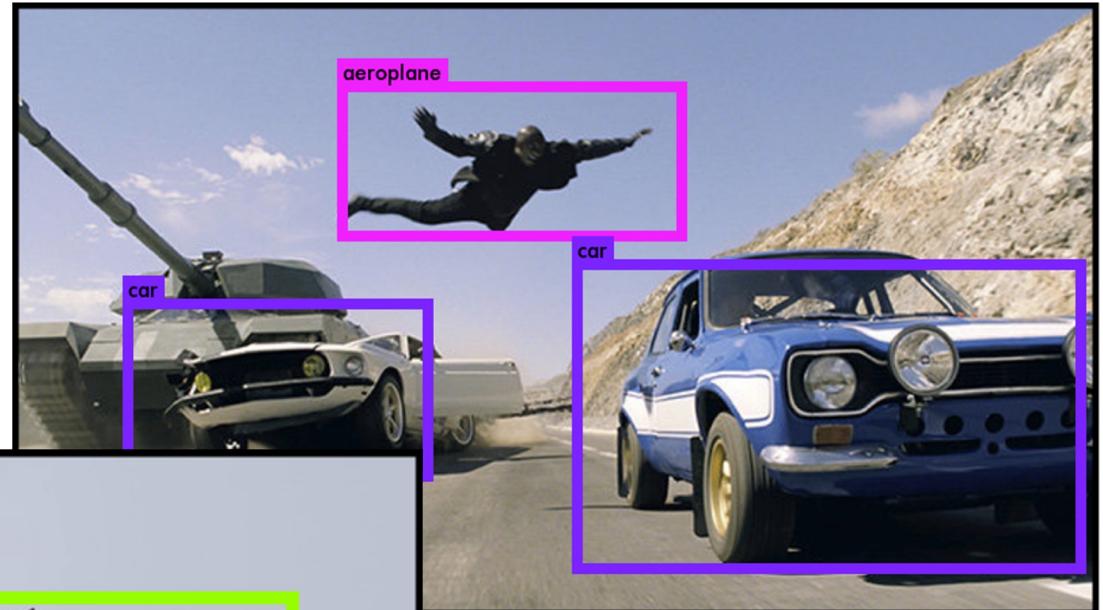
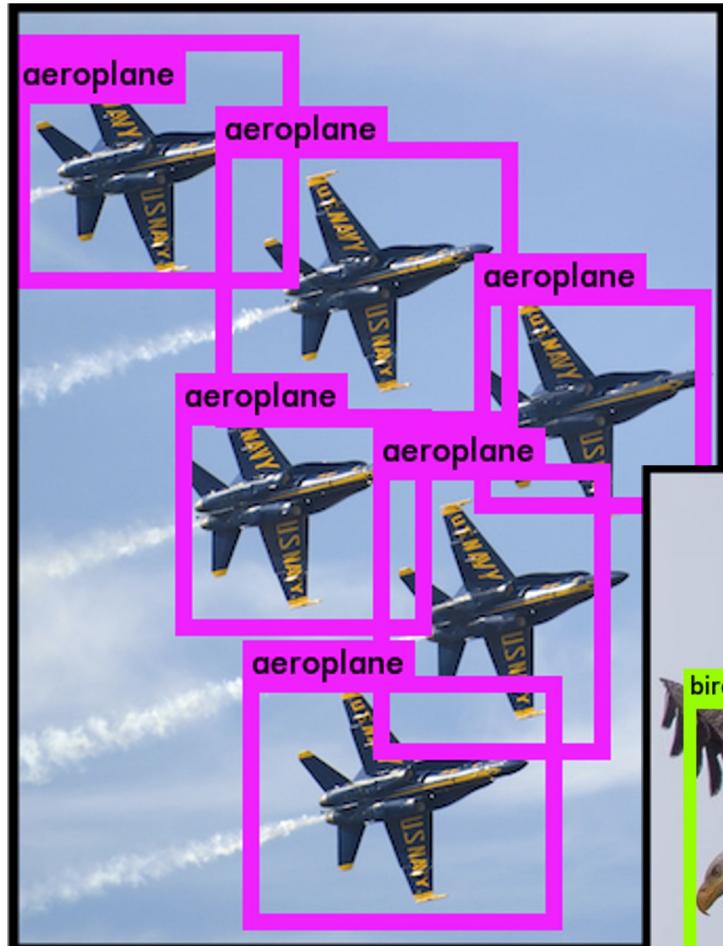


# We train with standard tricks:

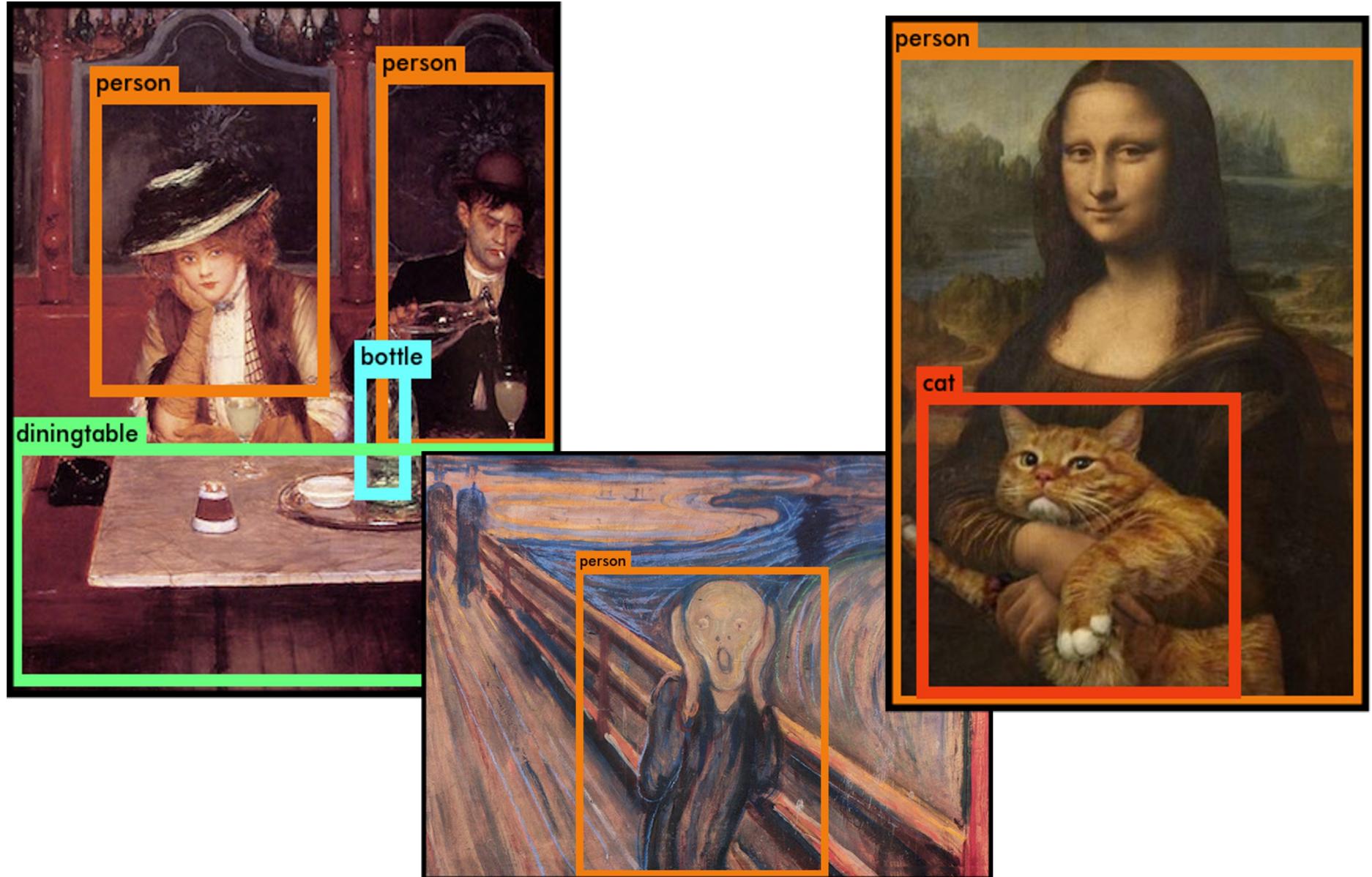
- Pretraining on Imagenet
- SGD with decreasing learning rate
- Extensive data augmentation
- For details, see the paper



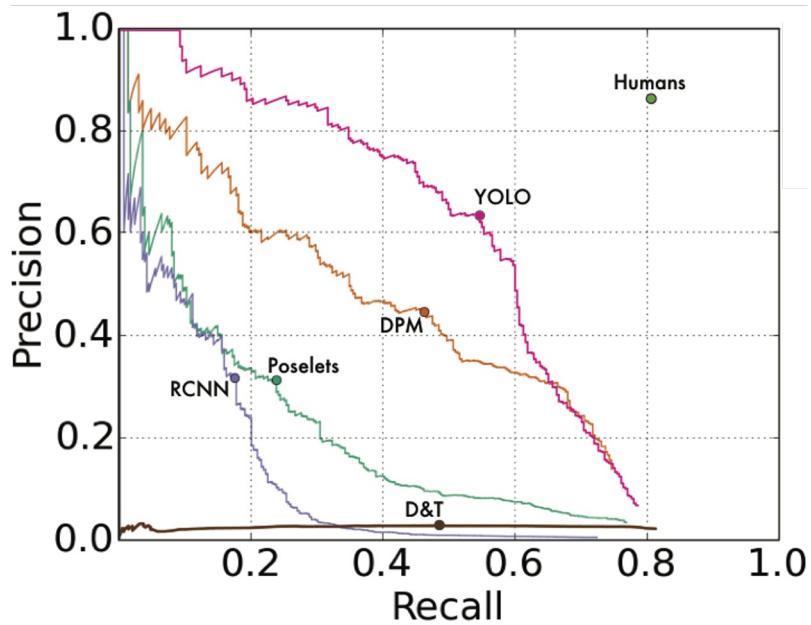
# YOLO works across a variety of natural images



It also generalizes well to new domains (like art)



# YOLO outperforms methods like DPM and R-CNN when generalizing to person detection in artwork



	VOC 2007	Picasso		People-Art
	AP	AP	Best $F_1$	AP
<b>YOLO</b>	<b>59.2</b>	<b>53.3</b>	<b>0.590</b>	<b>45</b>
R-CNN	54.2	10.4	0.226	26
DPM	43.2	37.8	0.458	32

S. Ginosar, D. Haas, T. Brown, and J. Malik. Detecting people in cubist art. In Computer Vision-ECCV 2014 Workshops, pages 101–116. Springer, 2014.

H. Cai, Q. Wu, T. Corradi, and P. Hall. The cross-depiction problem: Computer vision algorithms for recognising objects in artwork and in photographs.

Code available! [pjreddie.com/yolo](http://pjreddie.com/yolo)



	Pascal 2007 mAP	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	<del>63.4</del> 69.0	45 FPS	22 ms/img

