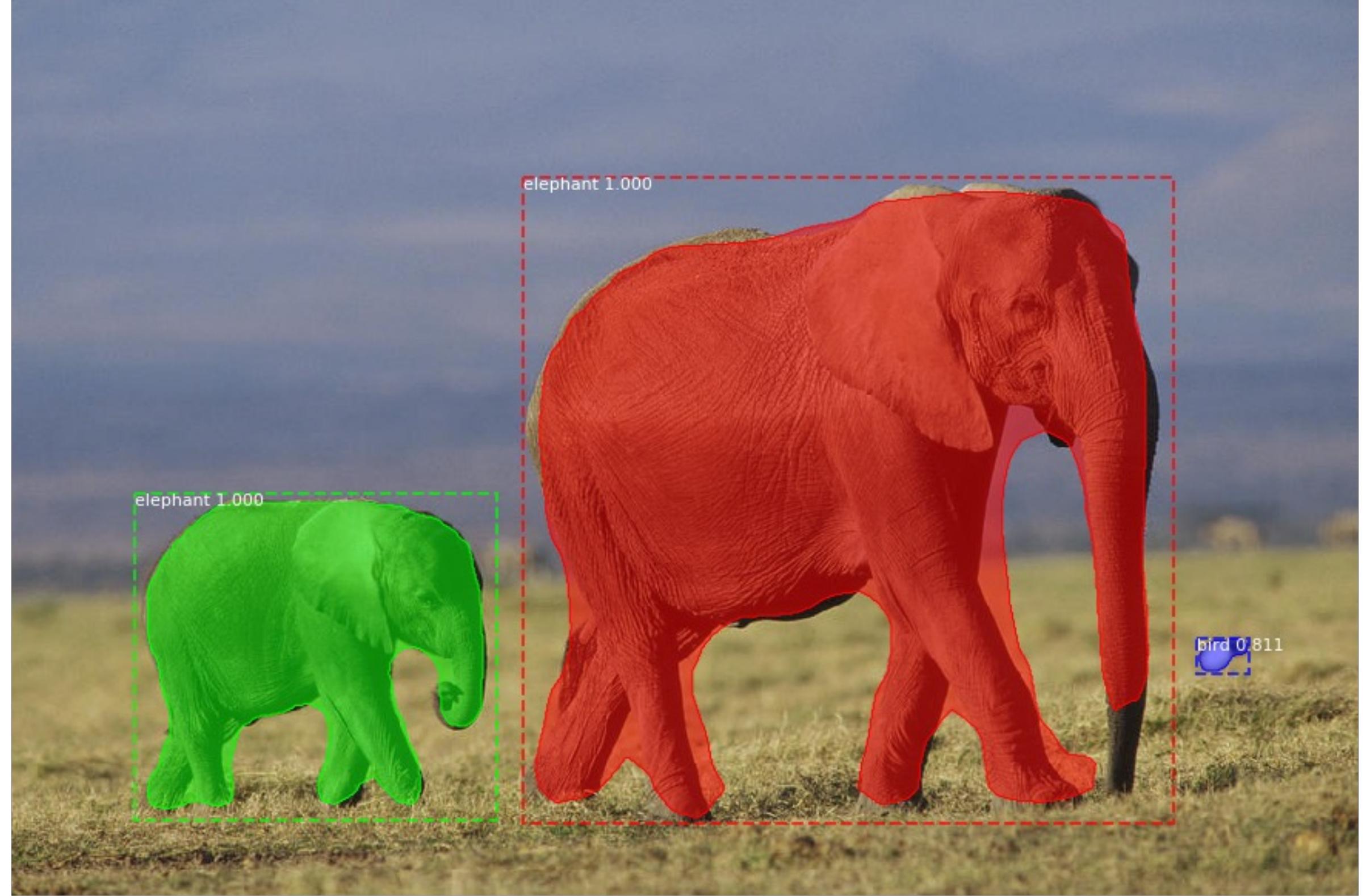


Convolutional Neural Networks for Object Detection



10. November 2023

Computer Vision Tasks

Classification



CAT

No spatial extent

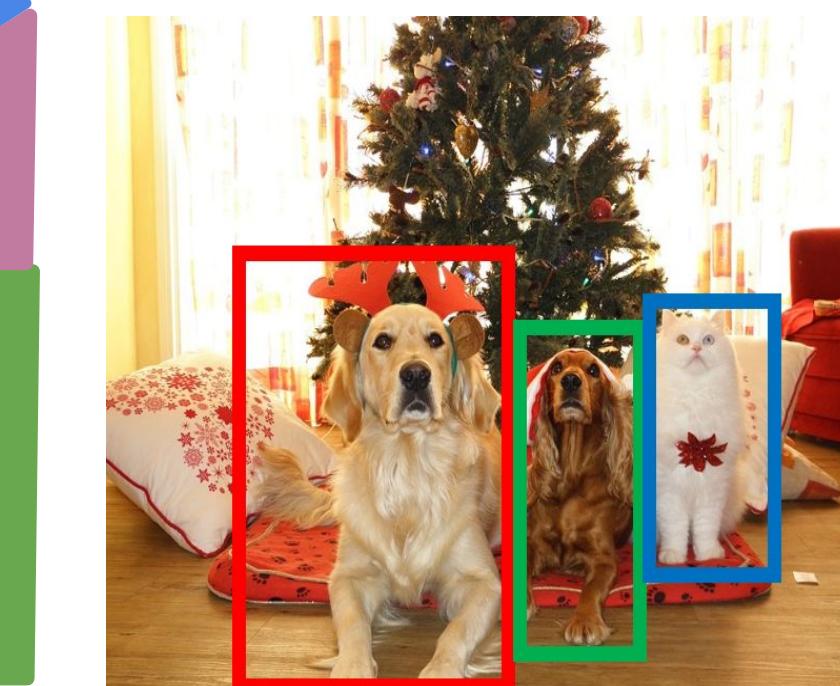
Semantic Segmentation



GRASS, CAT,
TREE, SKY

No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation



DOG, DOG, CAT

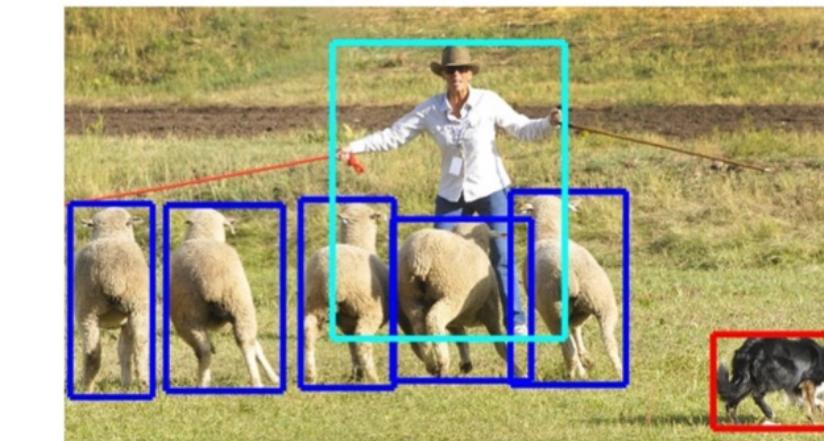
Multiple Object

This image is CC0 public domain

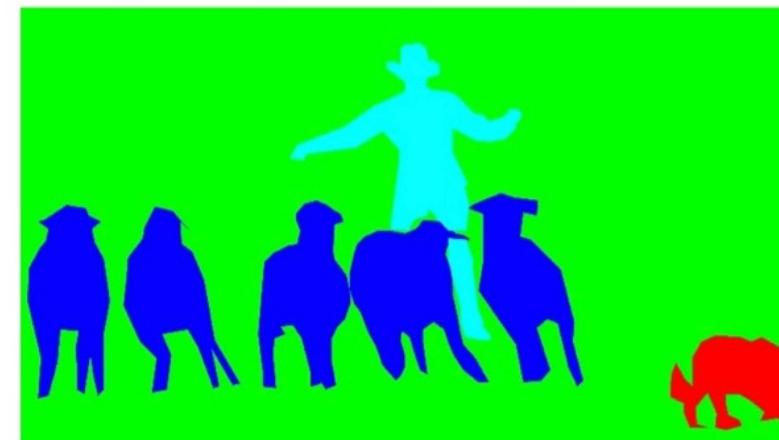
Datasets : Coco (Common Object Context)



(a) Image classification



(b) Object localization



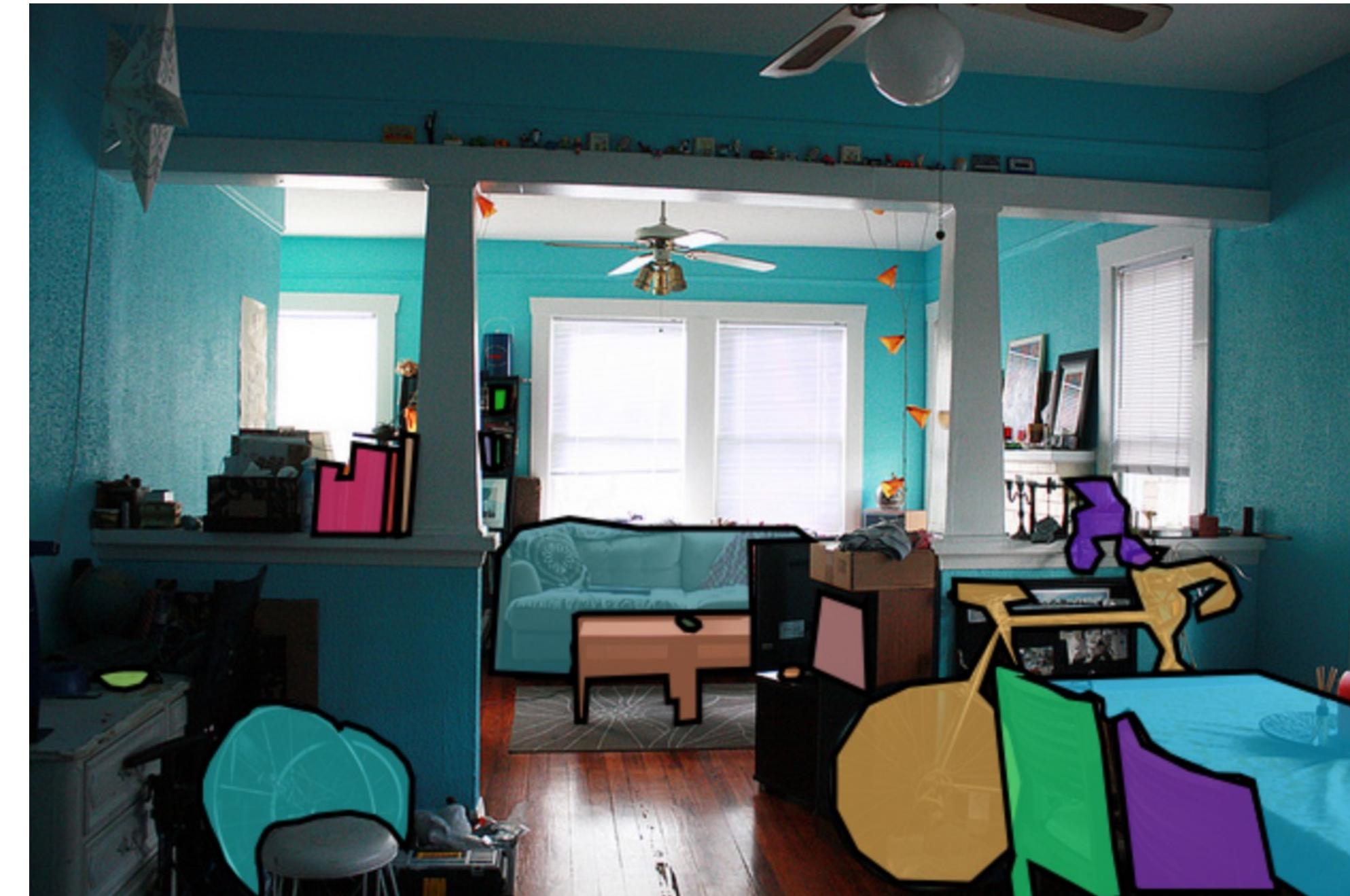
(c) Semantic segmentation



(d) This work

- Large Dataset for Object Detection, Image Segmentation, and Captioning
- 330'000 images
- 1.5 million objects
- 80 categories of objects

Datasets : Coco (Common Object Context)



* Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco: Common objects in context." In *European conference on computer vision*, pp. 740-755. Springer, Cham, 2014.

Datasets : Coco (Common Object Context)



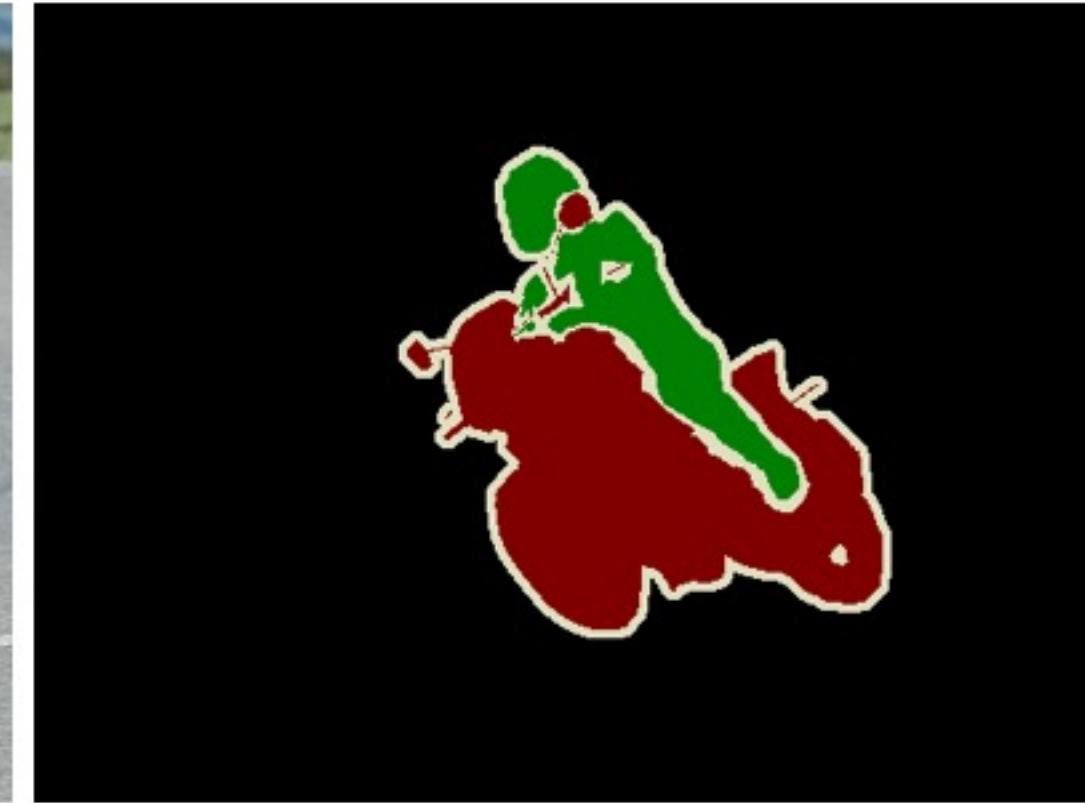
* Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft coco: Common objects in context." In *European conference on computer vision*, pp. 740-755. Springer, Cham, 2014.

Datasets : PASCAL Visual Object Classes (PASCAL VOC)*

Image

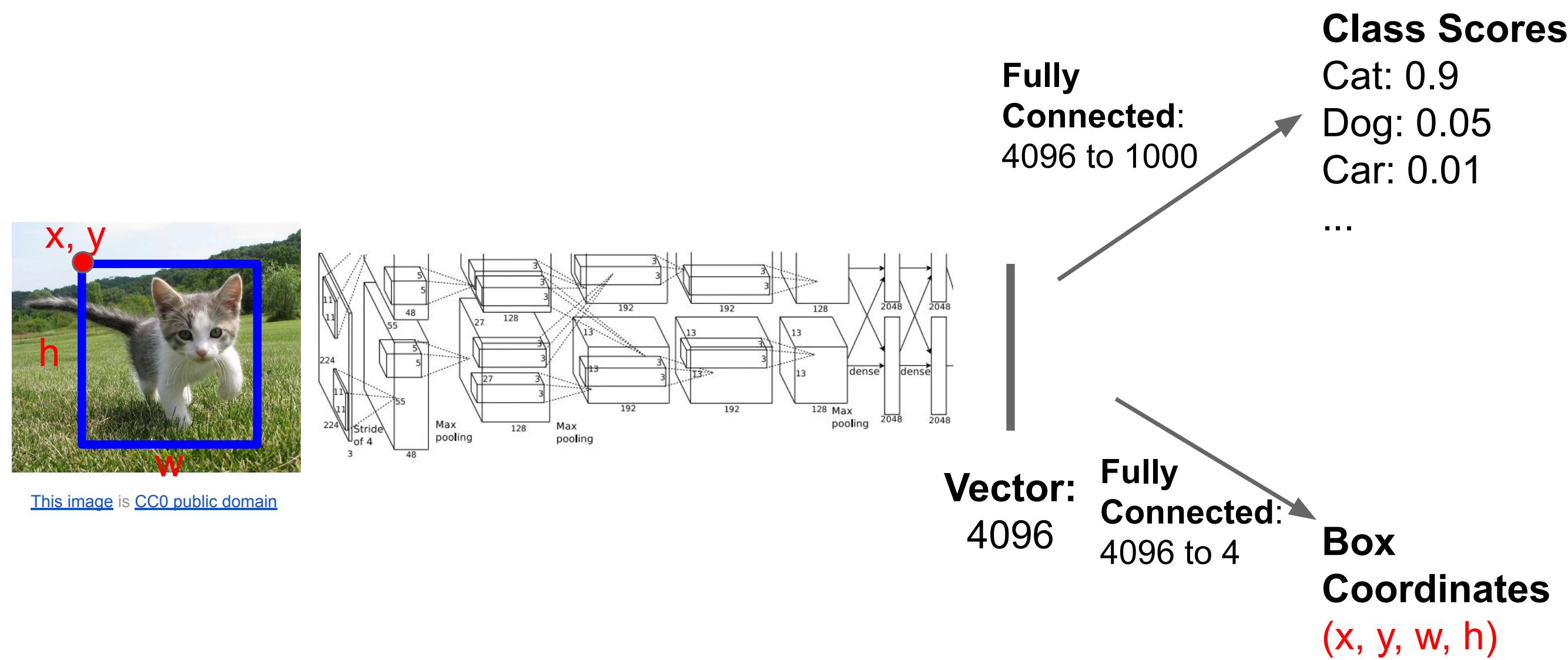


Objects

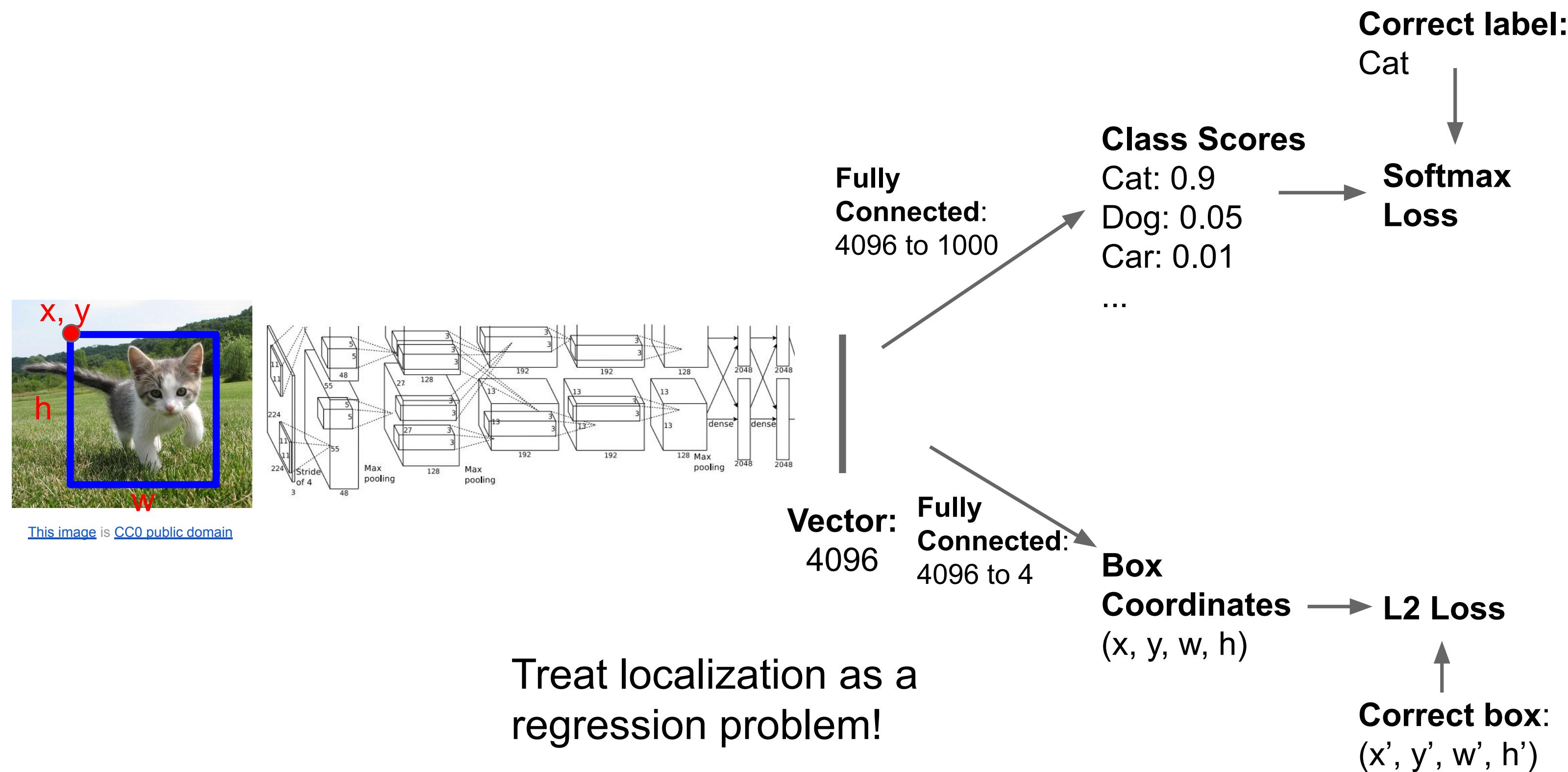


- 20 classes
- The train/val data has 11'530 images containing 27'450 ROI annotated objects and 6'929 segmentations.

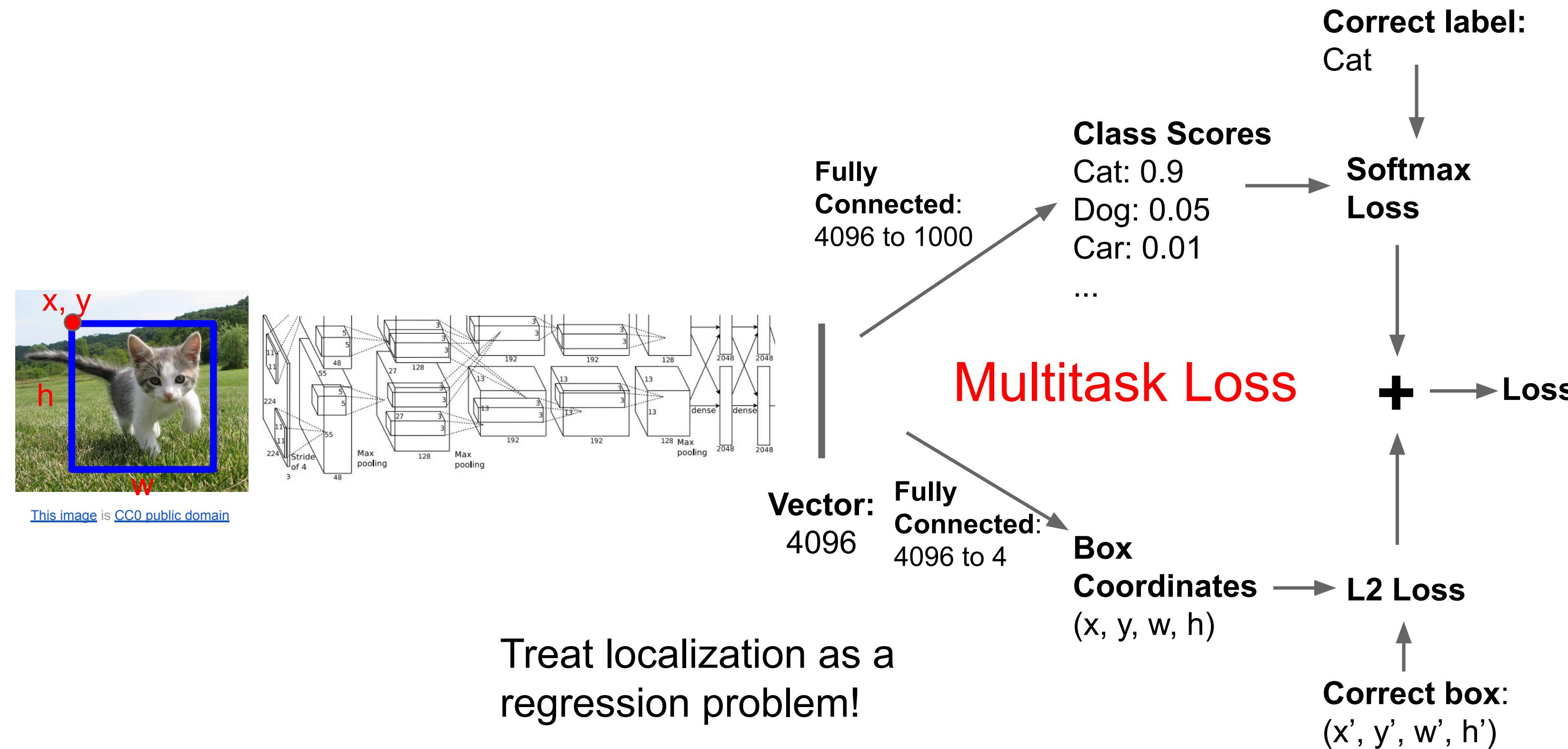
Object Detection Single Object : Classification + Localization



Object Detection Single Object : Classification + Localization



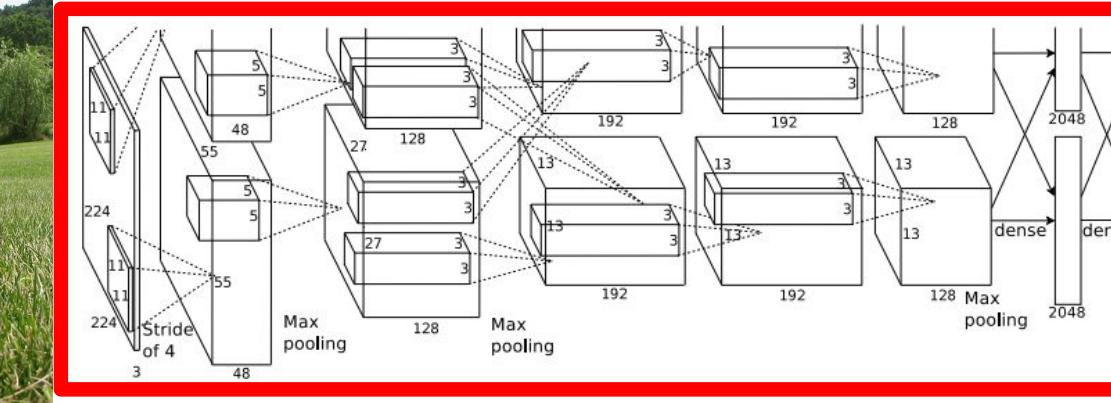
Object Detection Single Object : Classification + Localization



Object Detection Single Object : Classification + Localization



This image is CC0 public domain



Often pretrained on ImageNet
(Transfer learning)

Treat localization as a
regression problem!

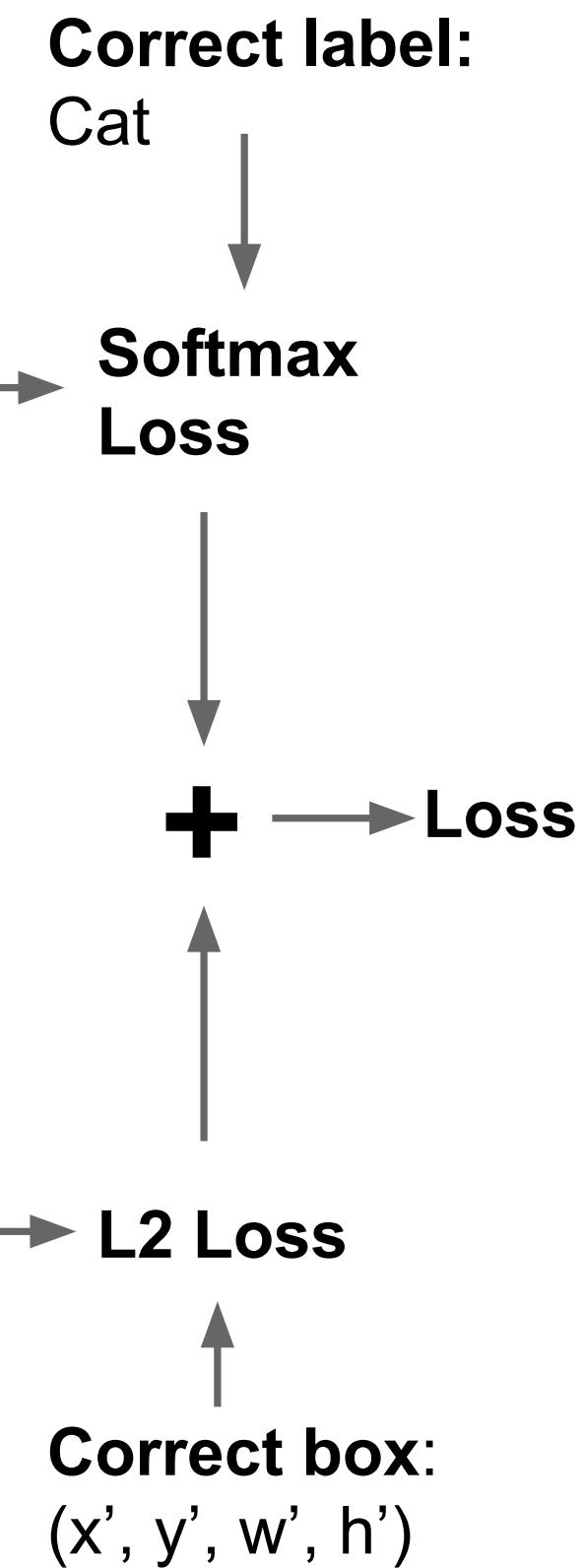
Fully Connected:
4096 to 1000

Vector:
4096

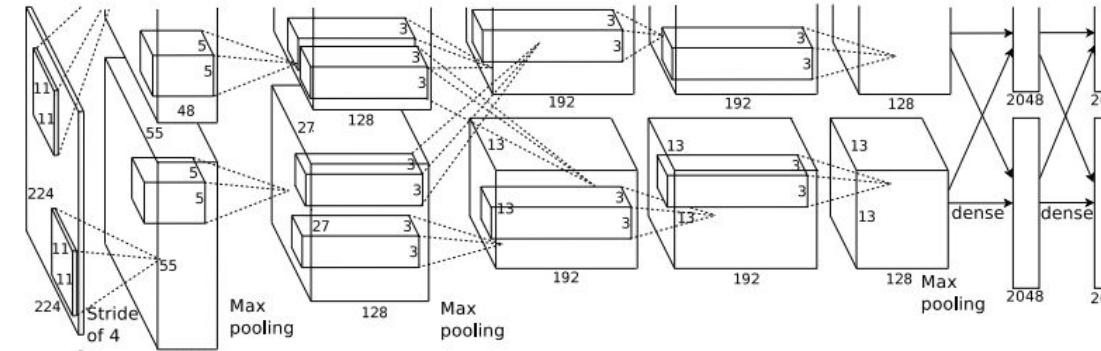
Fully Connected:
4096 to 4

Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

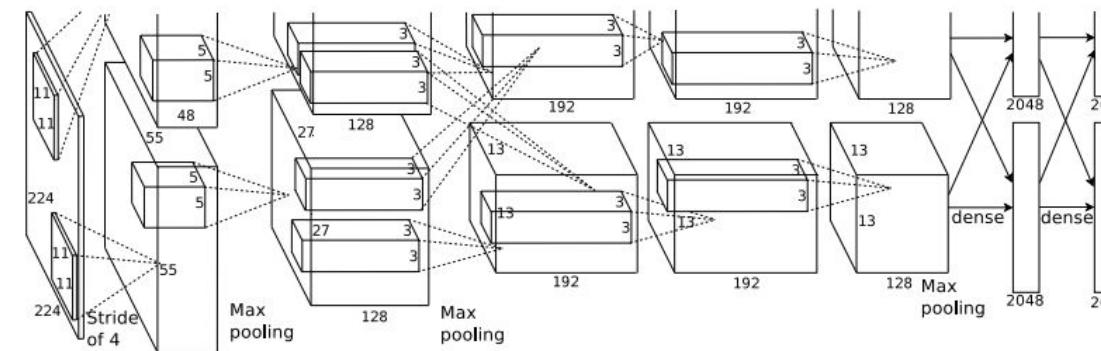
Box Coordinates → L2 Loss
 (x, y, w, h)



Object Detection : Multiple Objects



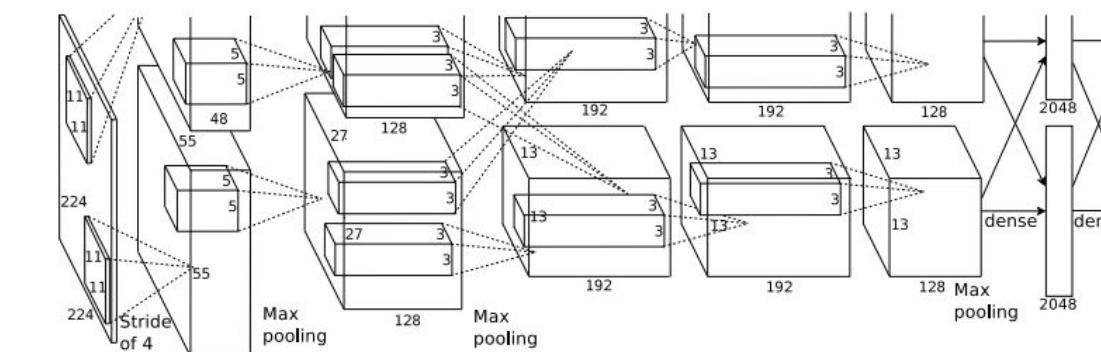
CAT: (x, y, w, h)



DOG: (x, y, w, h)

DOG: (x, y, w, h)

CAT: (x, y, w, h)

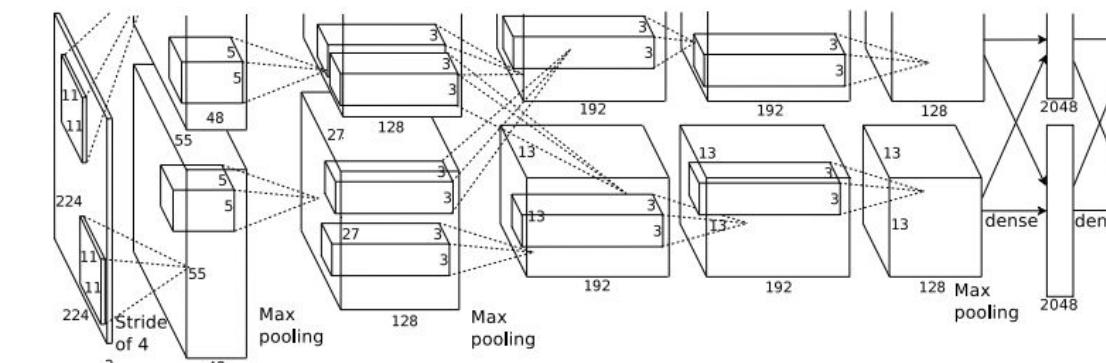
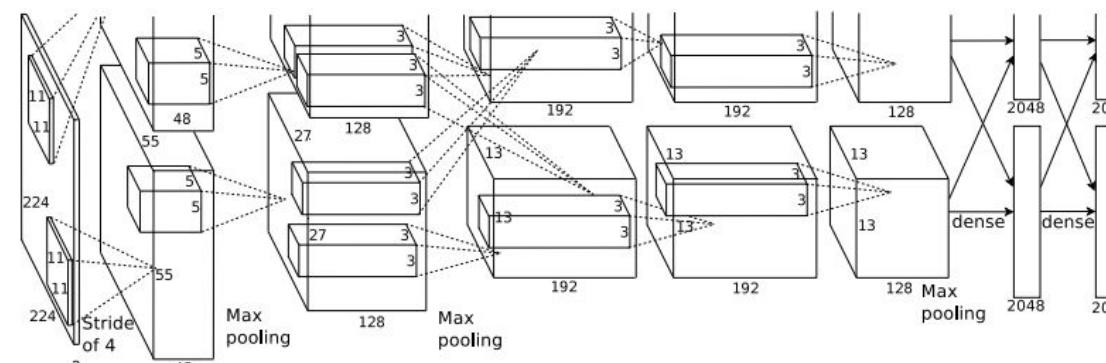
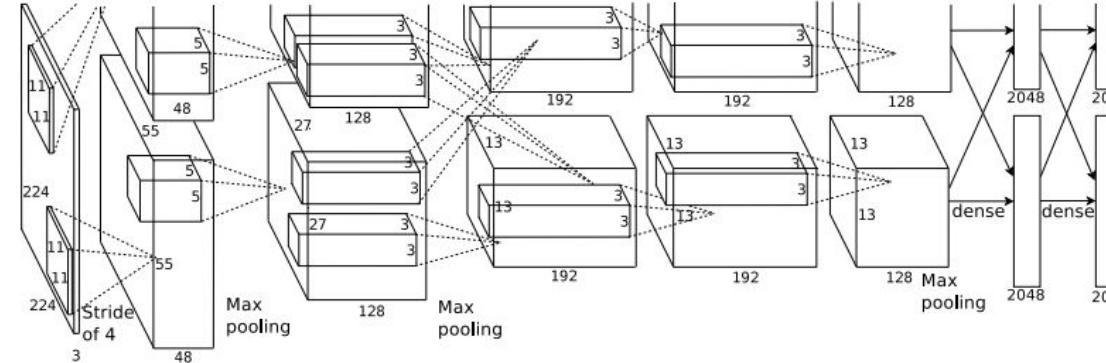


DUCK: (x, y, w, h)

DUCK: (x, y, w, h)

....

Object Detection : Multiple Objects



Each image needs a
different number of outputs!

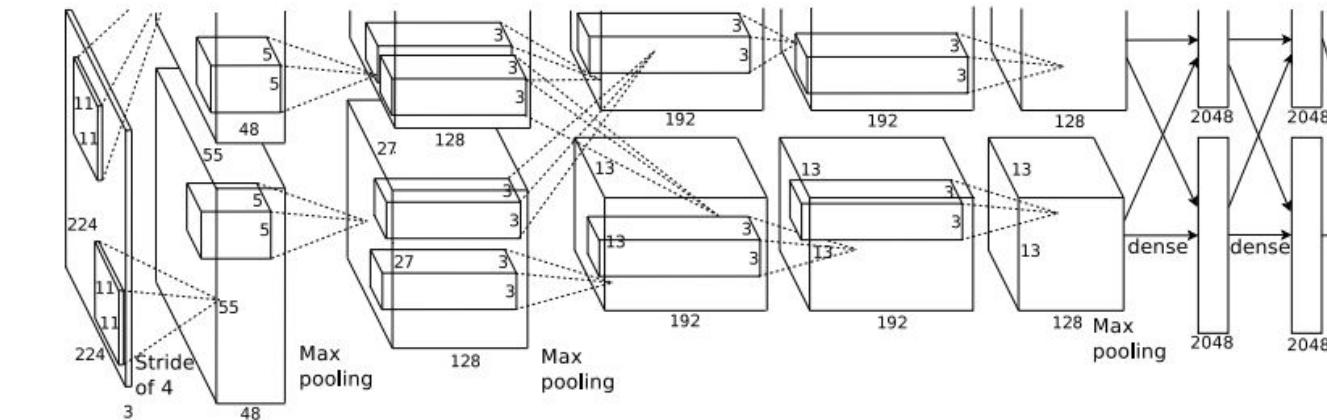
CAT: (x, y, w, h) 4 numbers

DOG: (x, y, w, h)
DOG: (x, y, w, h) 16 numbers
CAT: (x, y, w, h)

DUCK: (x, y, w, h) Many
DUCK: (x, y, w, h) numbers!
....

Object Detection : Multiple Objects with Sliding Window

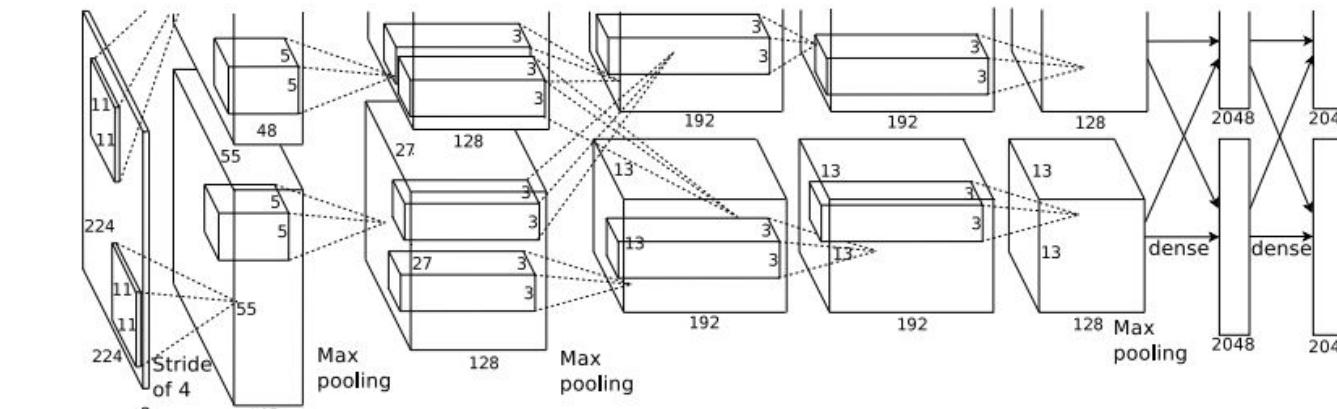
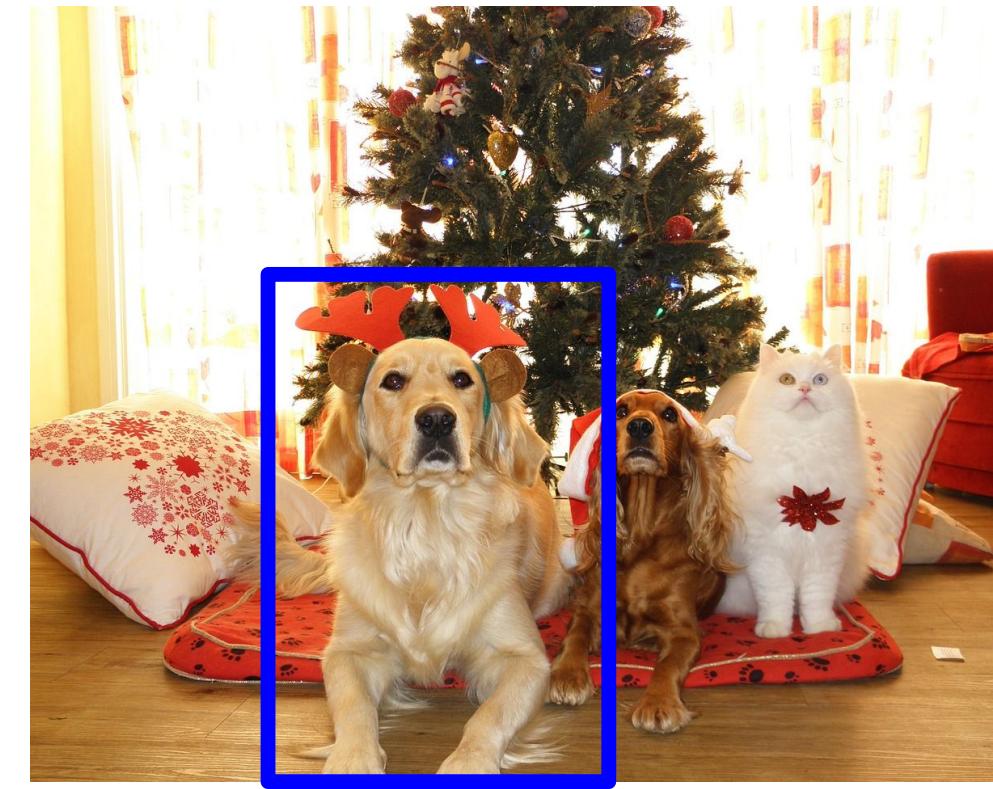
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

Object Detection : Multiple Objects with Sliding Window

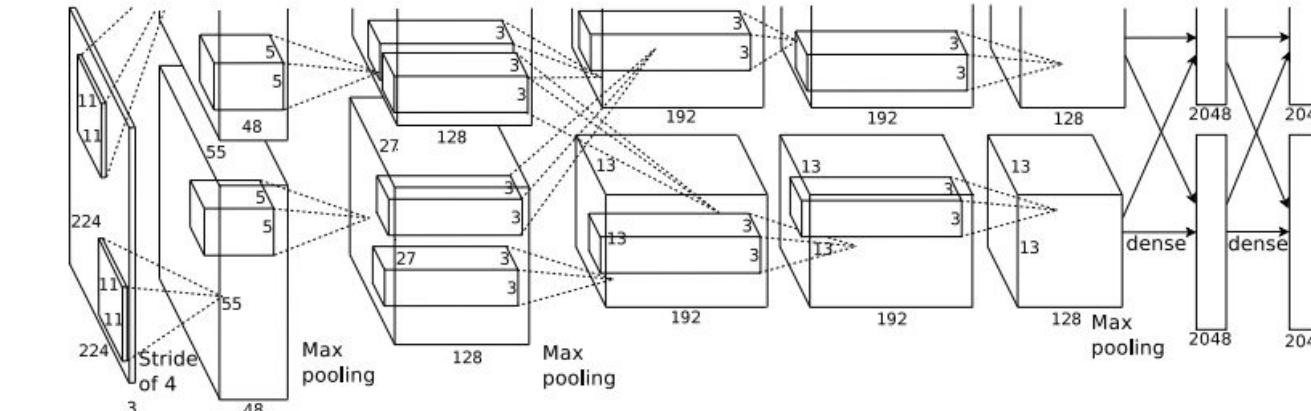
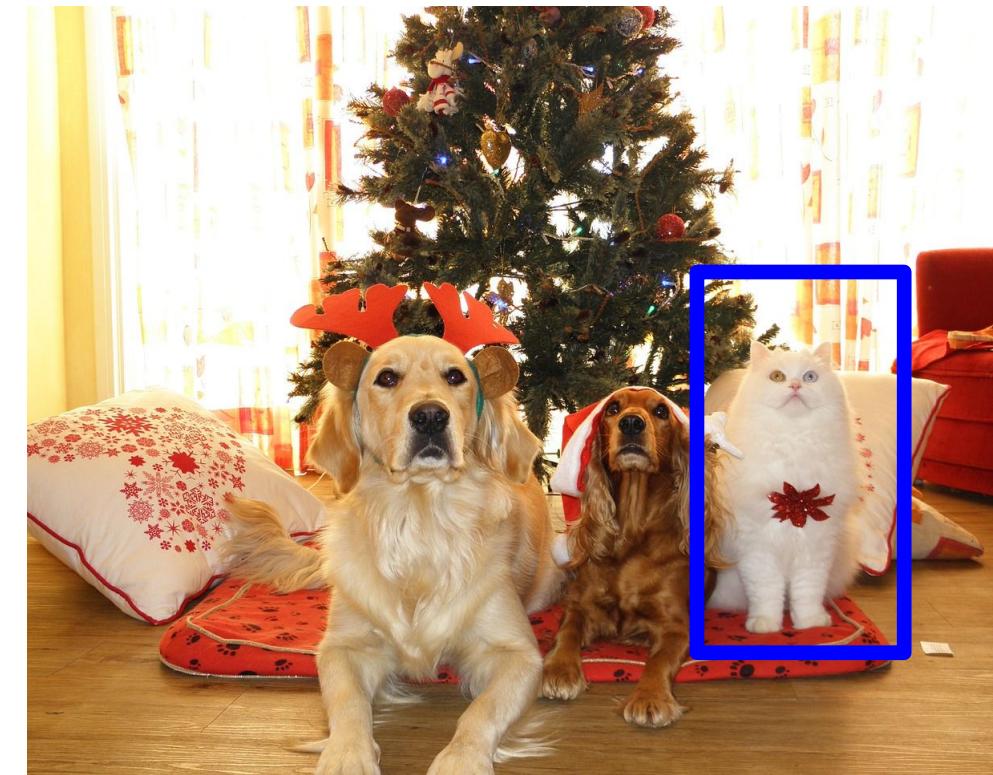
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Object Detection : Multiple Objects with Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

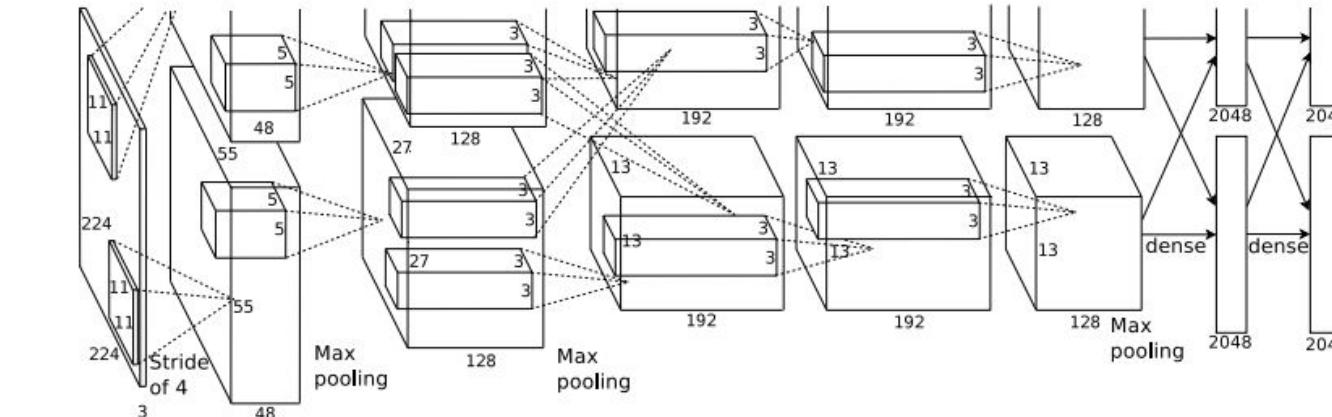
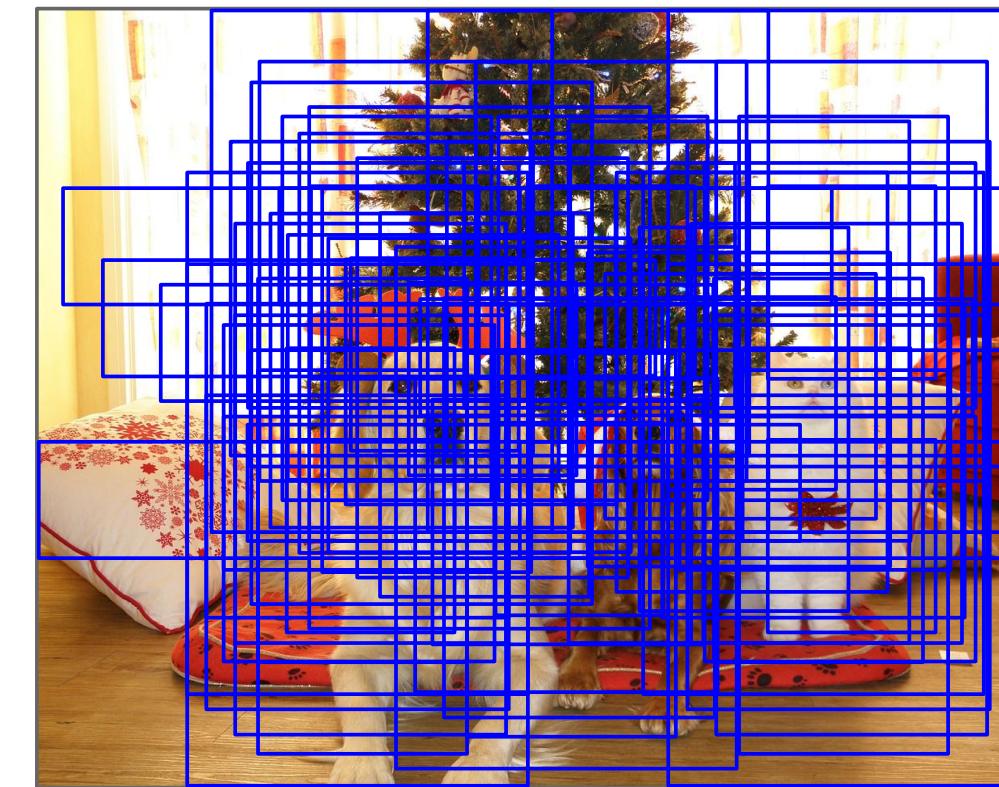


Dog? NO
Cat? YES
Background? NO

What's the problem with this approach?

Object Detection : Multiple Objects with Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

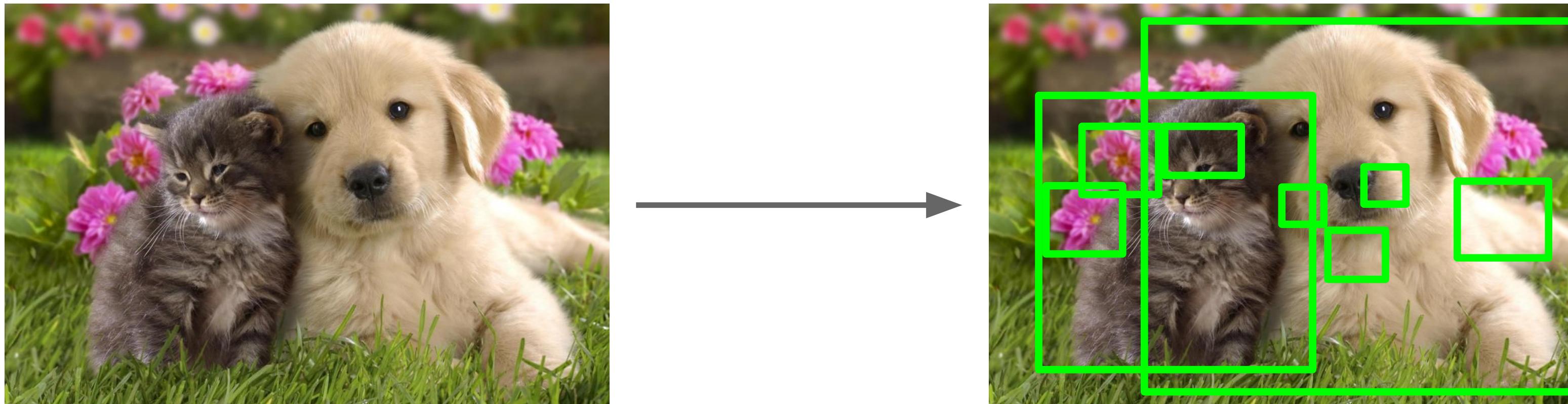


Problem: Need to apply CNN to huge number of locations, scales, and aspect ratios, very computationally expensive!

Dog? NO
Cat? YES
Background? NO

Regions Proposals : Selective Search

- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 2000 region proposals in a few seconds on CPU



Alexe et al, "Measuring the objectness of image windows", TPAMI 2012

Uijlings et al, "Selective Search for Object Recognition", IJCV 2013

Cheng et al, "BING: Binarized normed gradients for objectness estimation at 300fps", CVPR 2014

Zitnick and Dollar, "Edge boxes: Locating object proposals from edges", ECCV 2014

R-CNN



Input image

Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

R-CNN

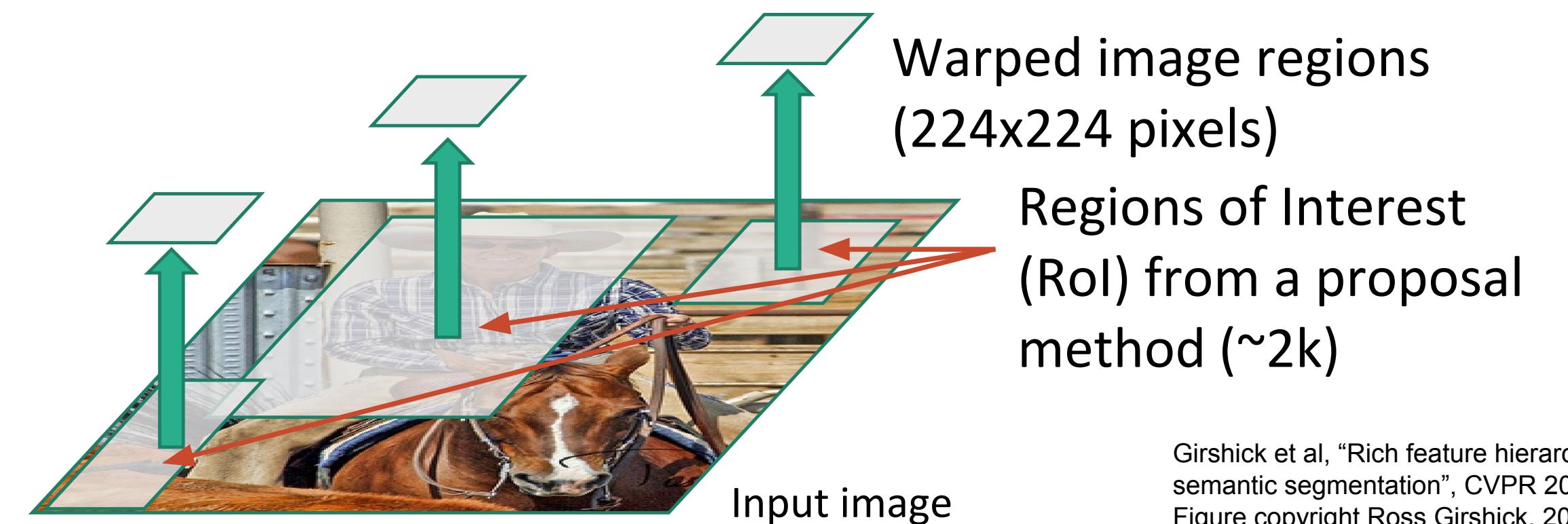


Input image

Regions of Interest
(RoI) from a proposal
method (~2k)

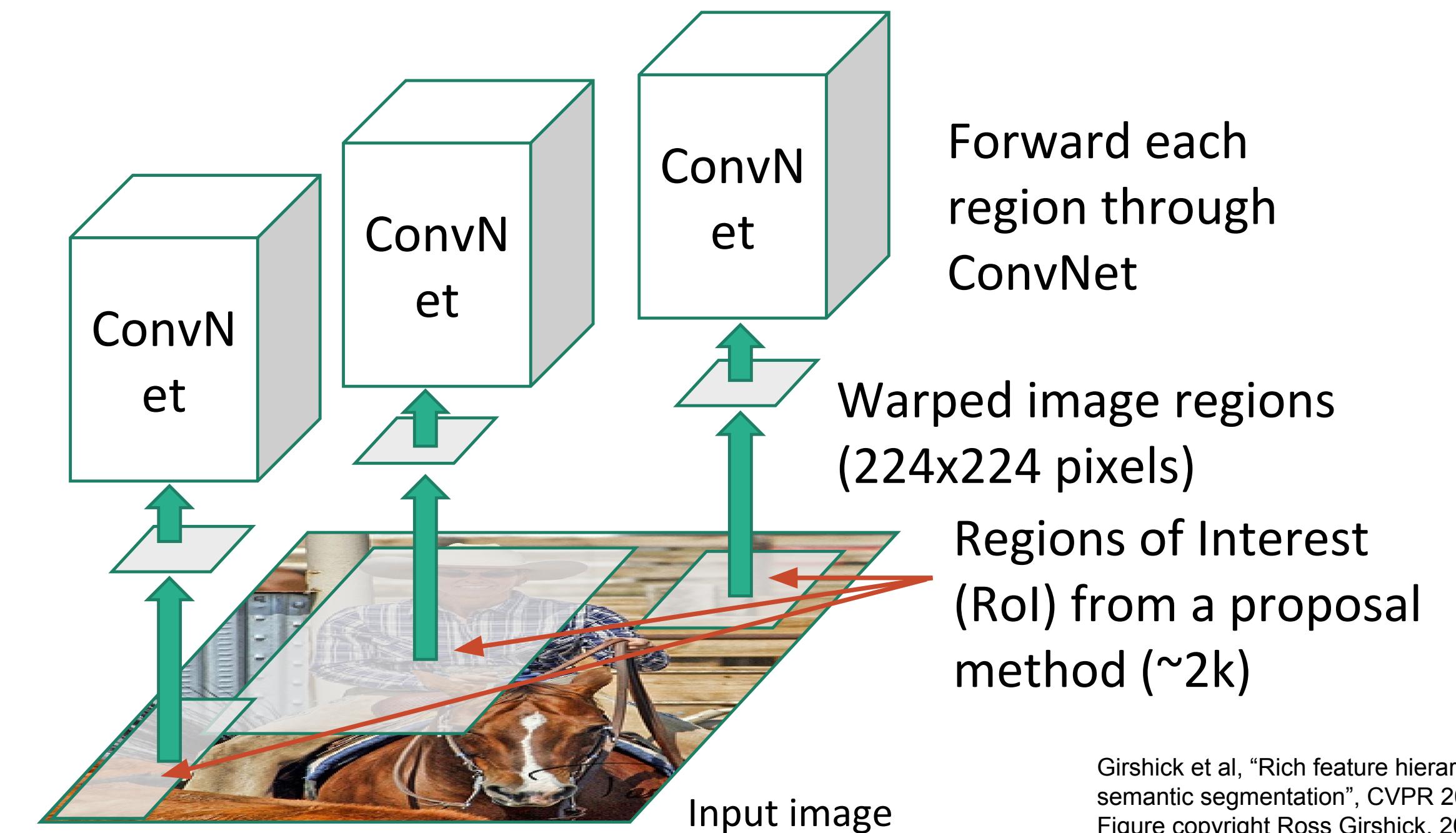
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

R-CNN



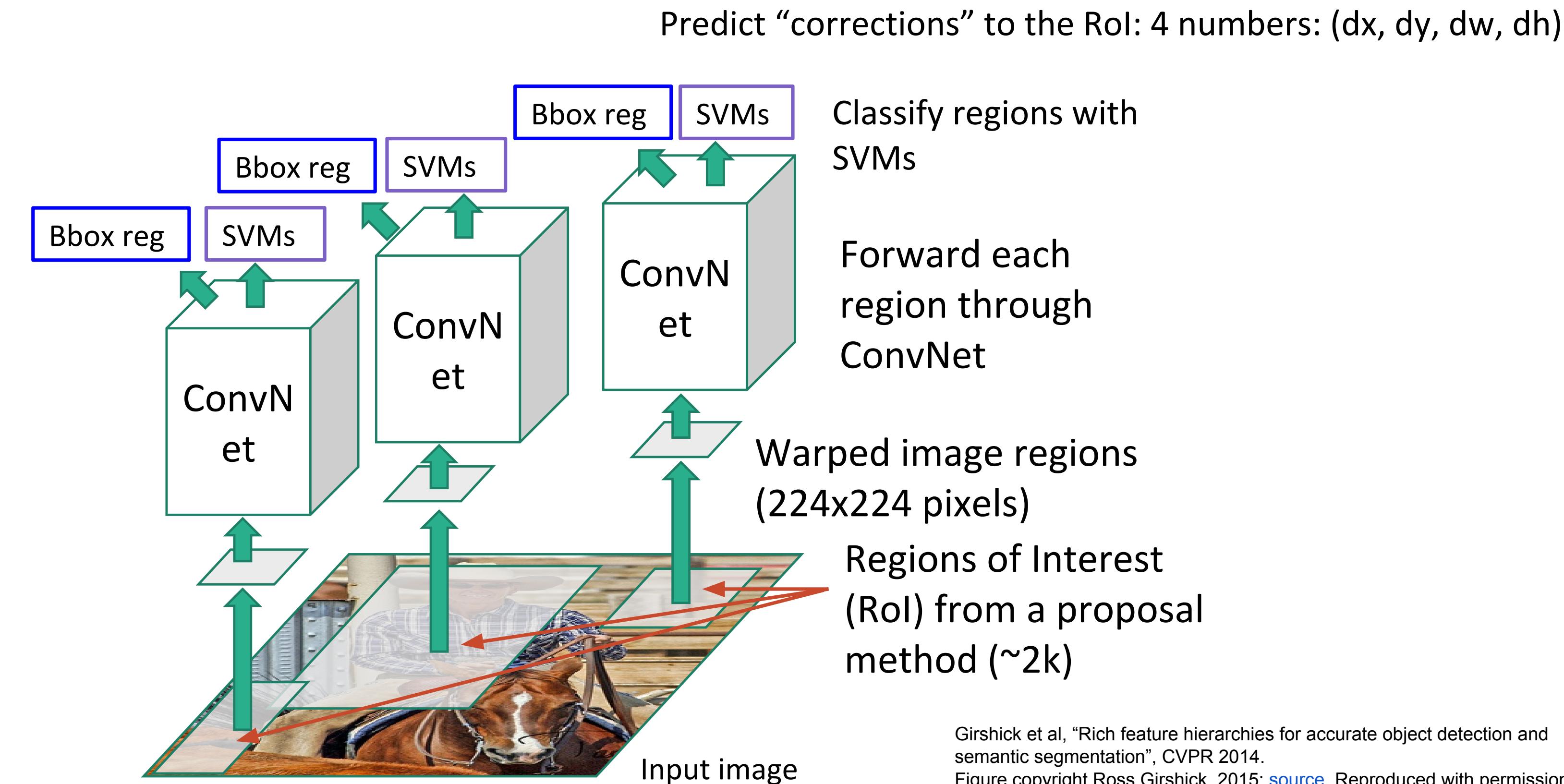
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

R-CNN

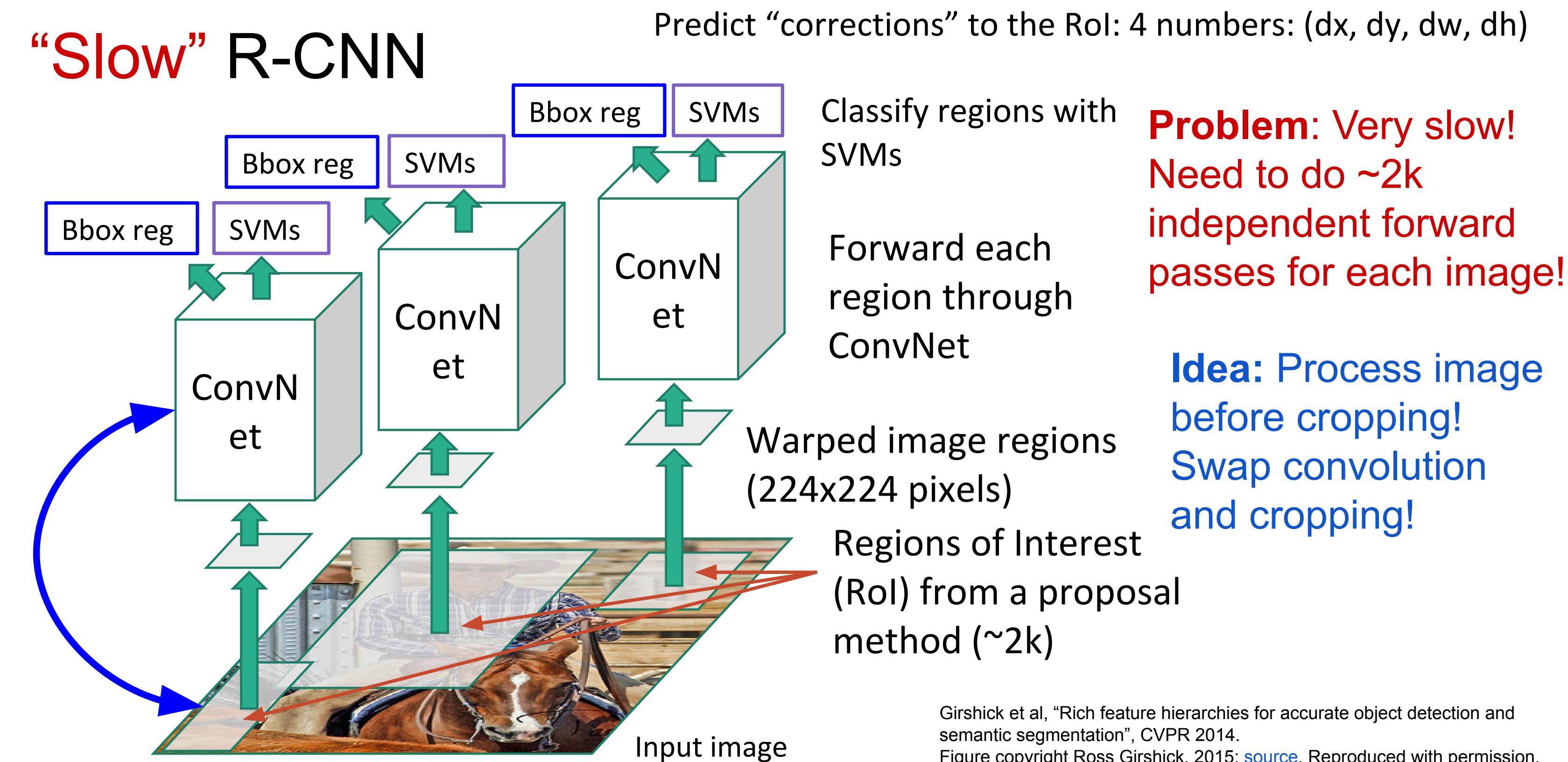


Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

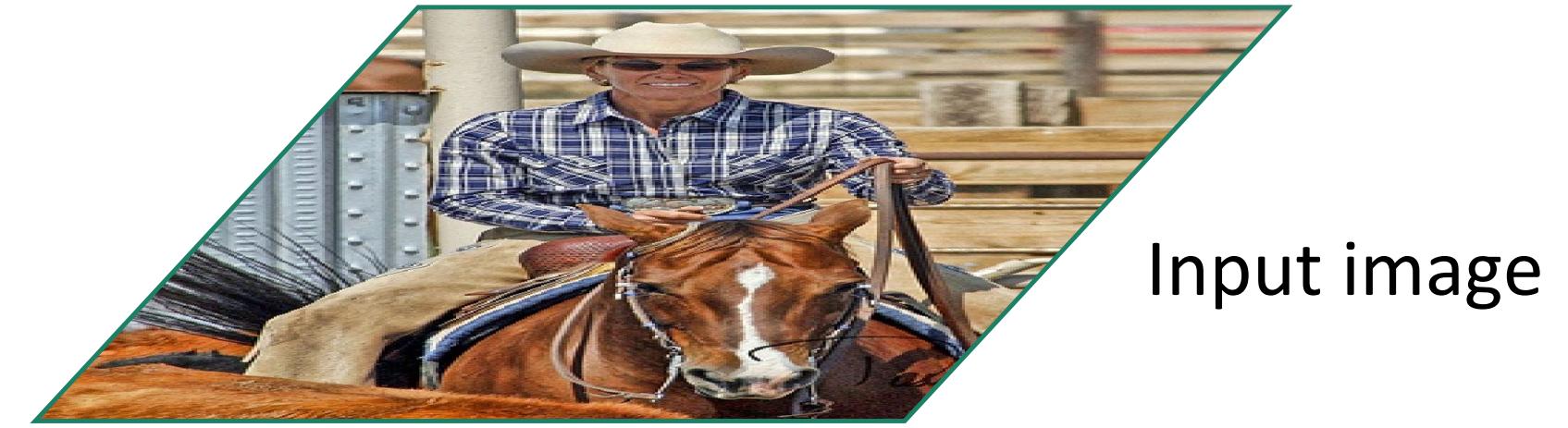
R-CNN



R-CNN

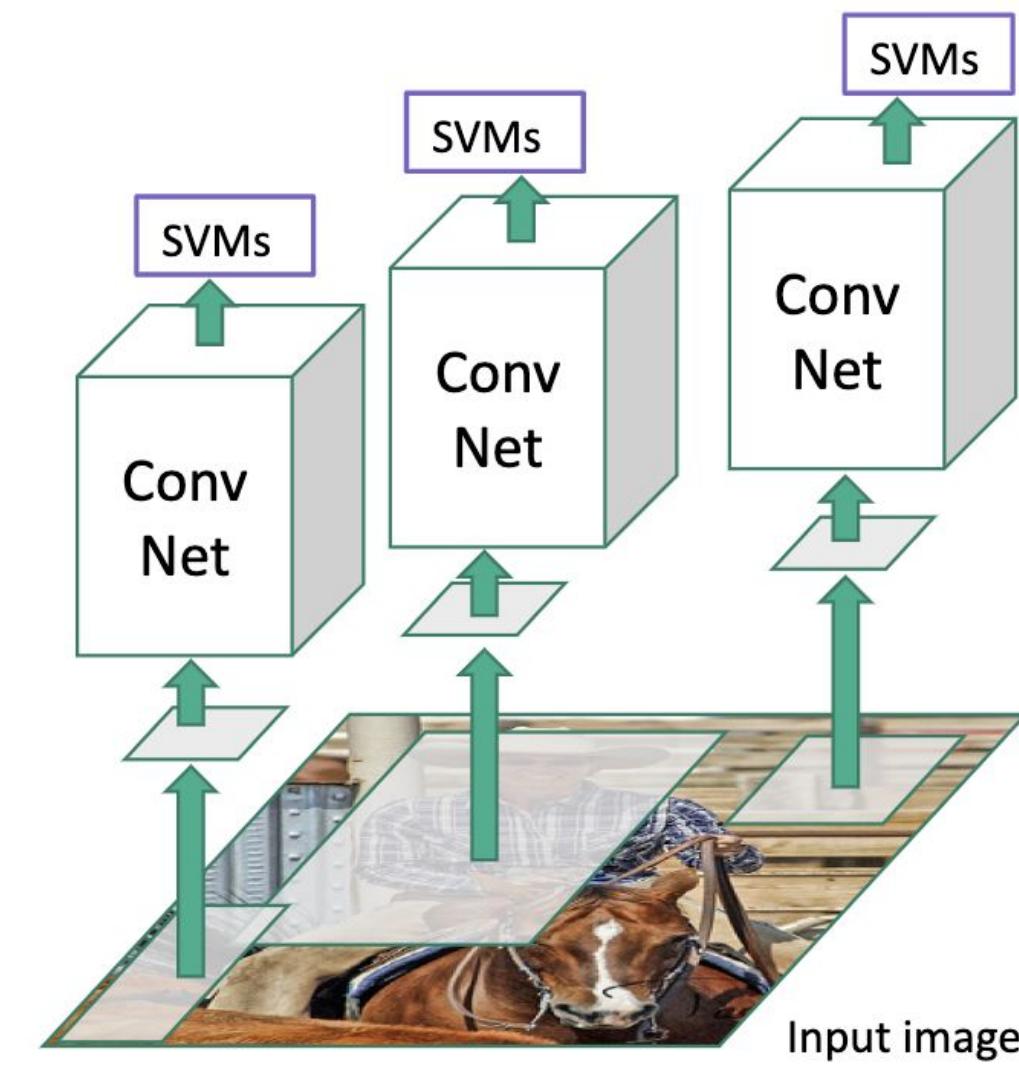


Fast R-CNN



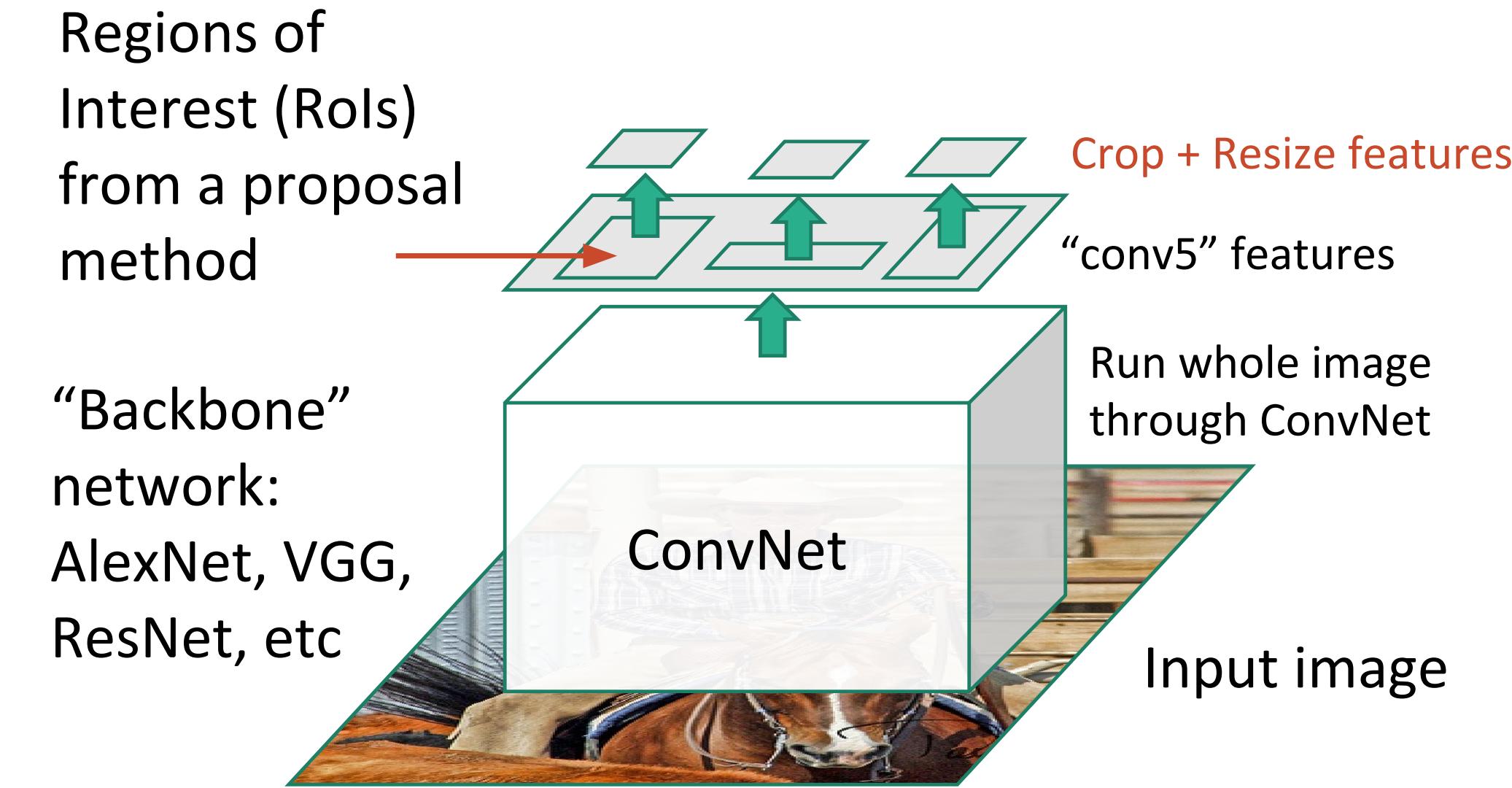
Input image

“Slow” R-CNN

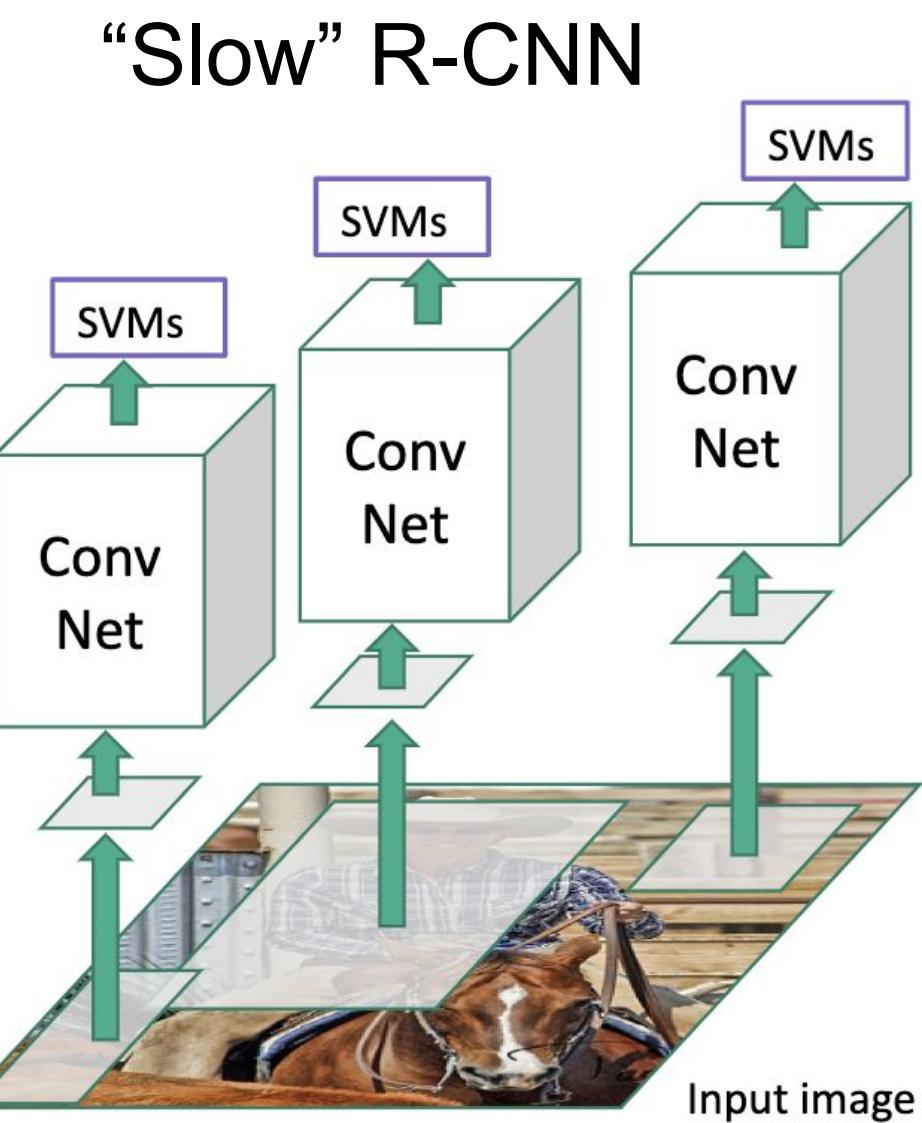


Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

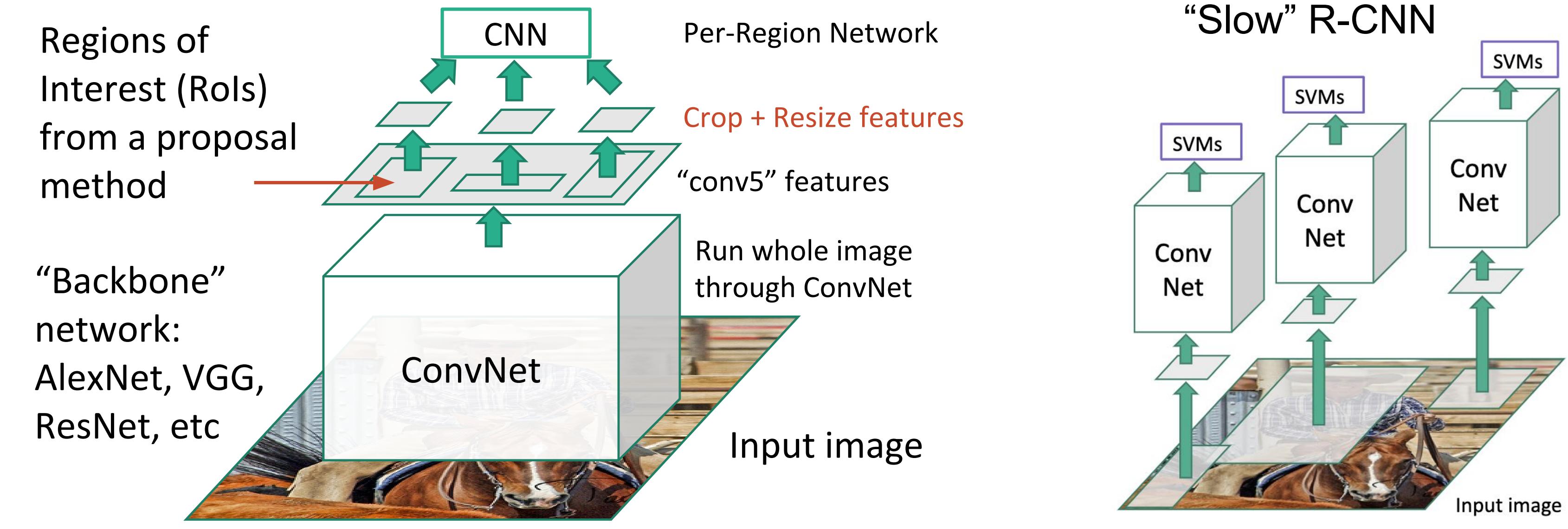
Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

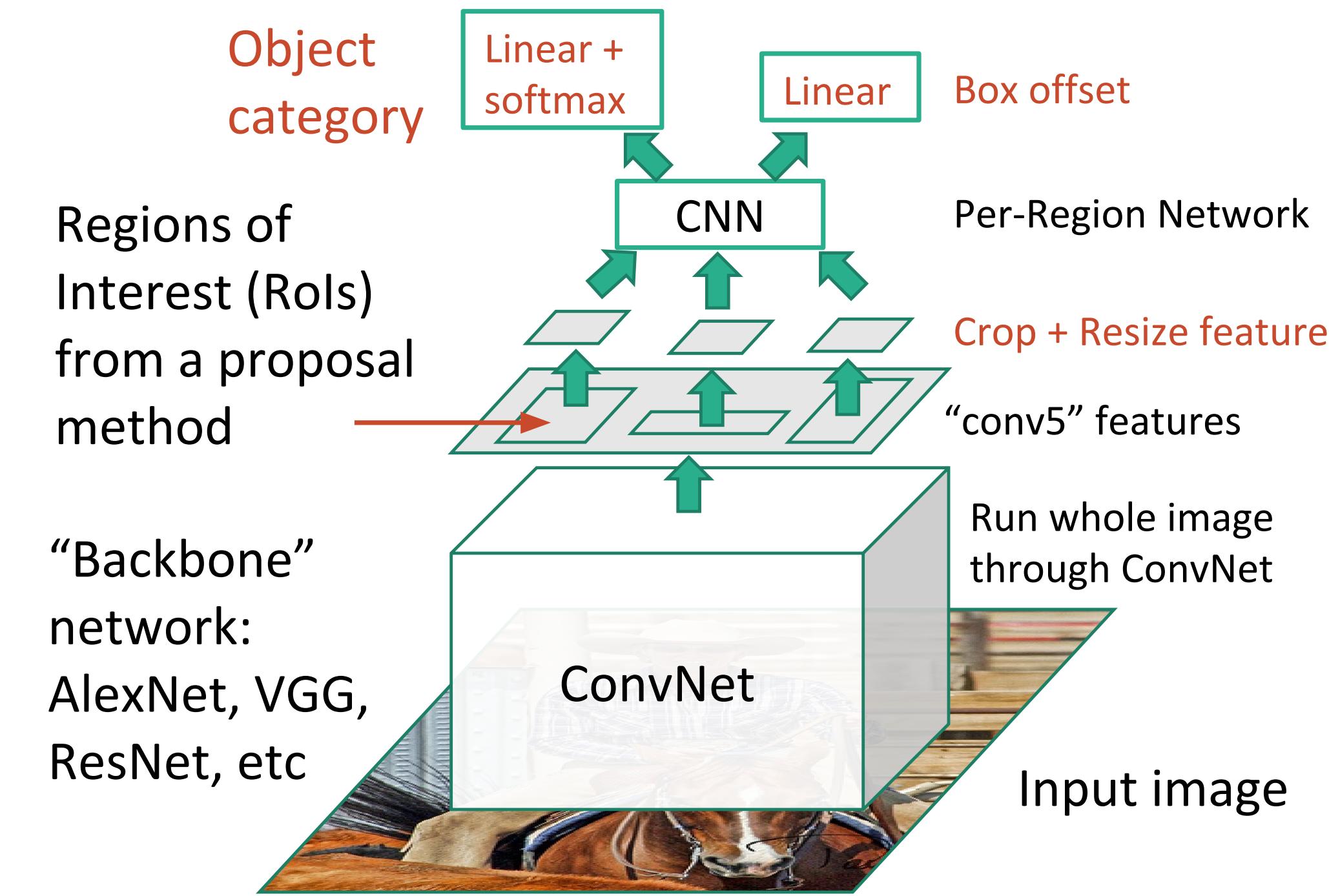


Fast R-CNN

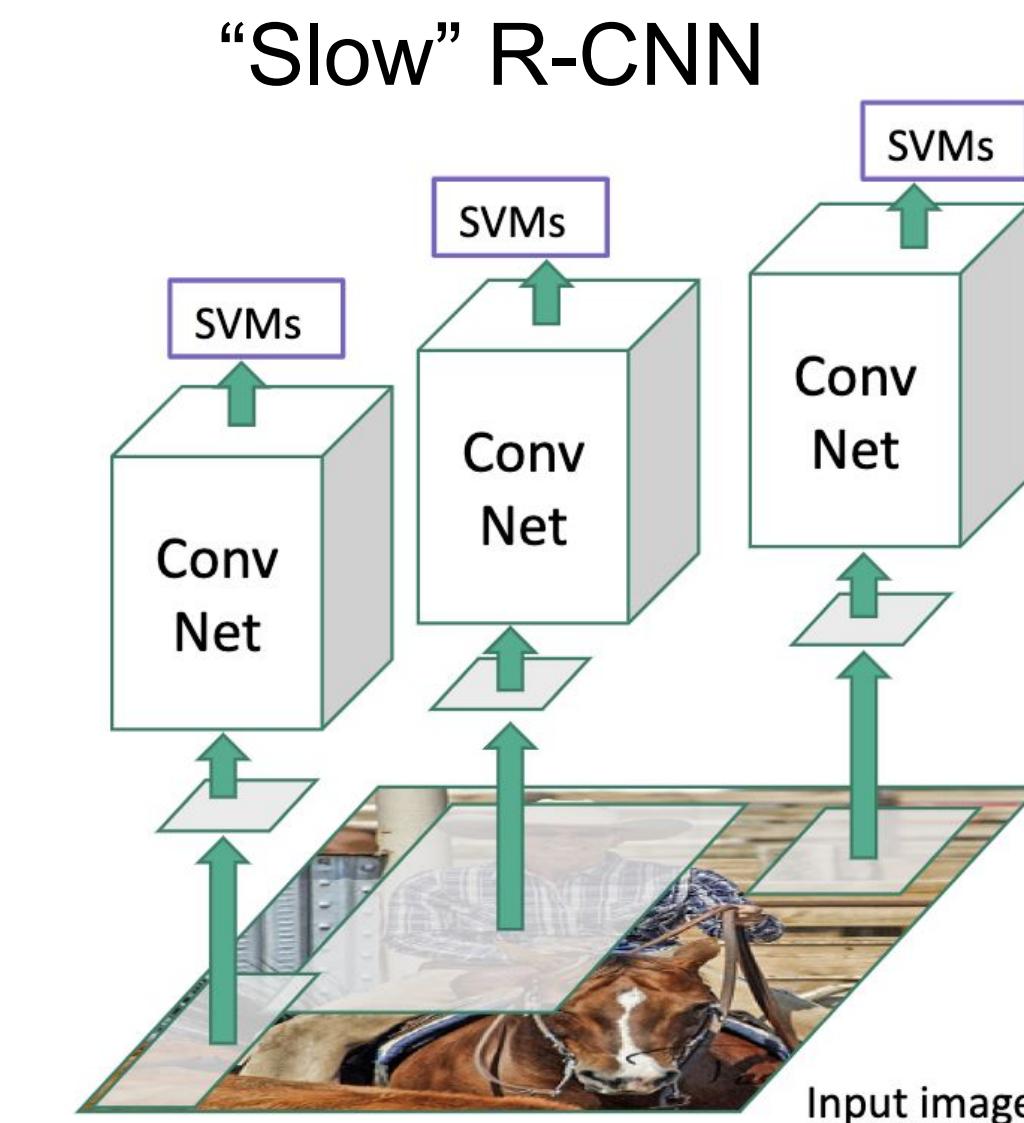


Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

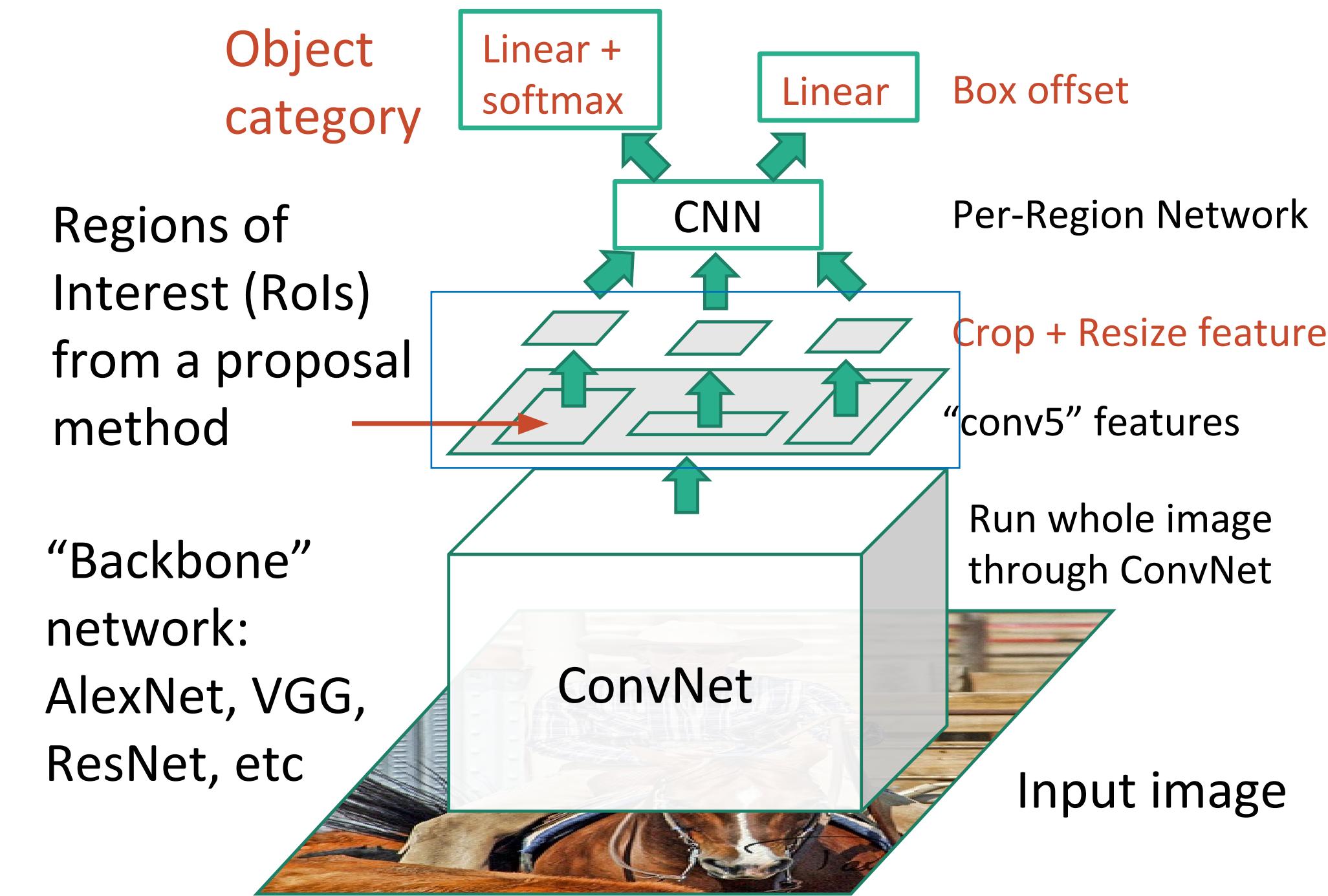
Fast R-CNN



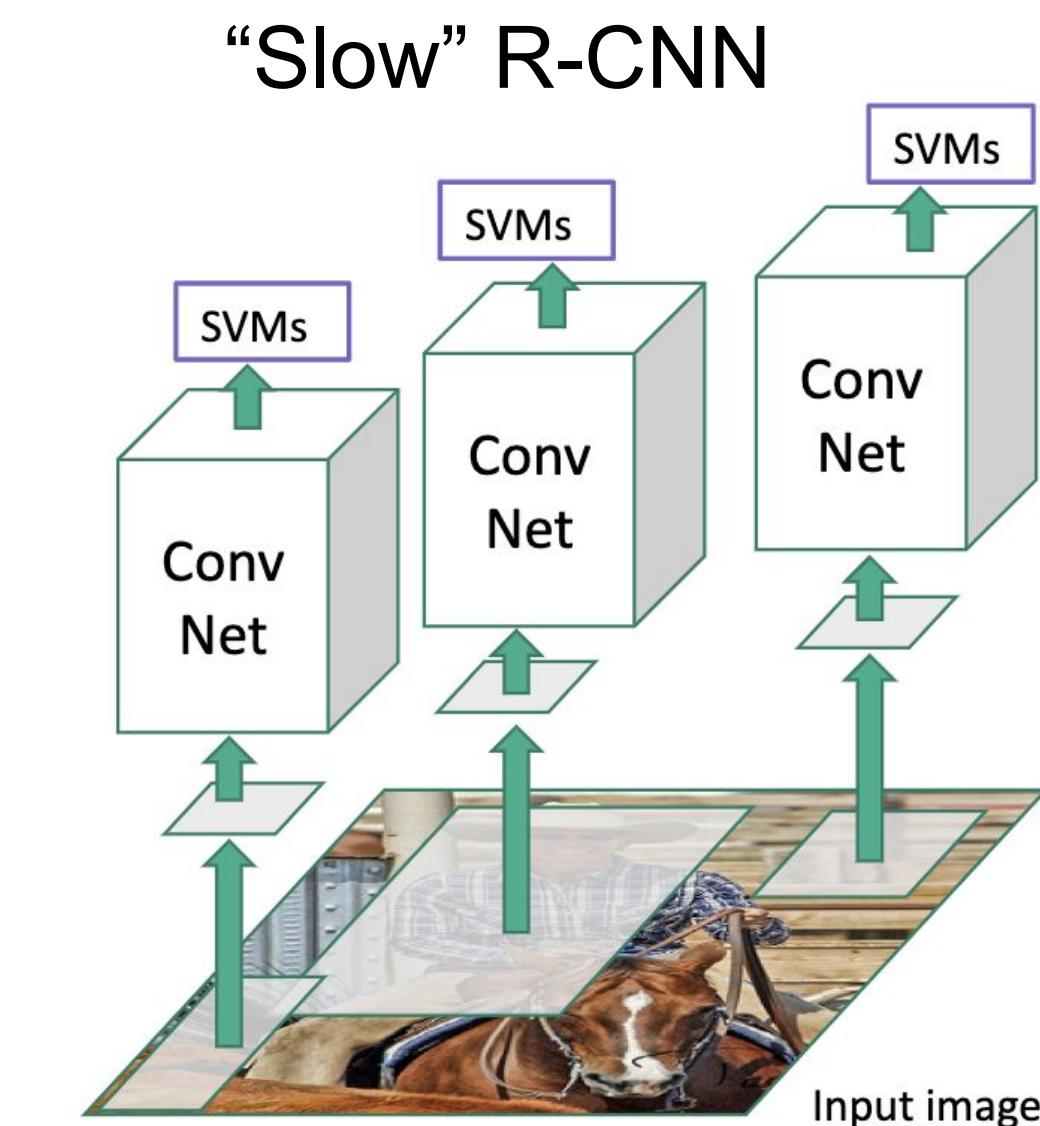
Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.



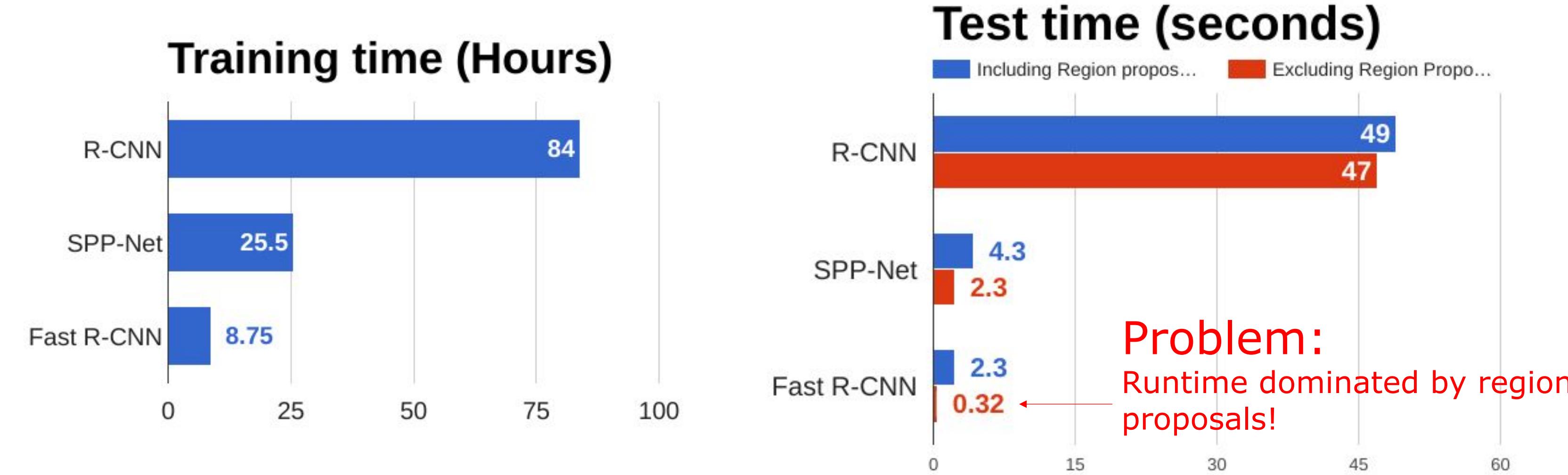
Fast R-CNN



Girshick, “Fast R-CNN”, ICCV 2015. Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.



R-CNN vs Fast R-CNN



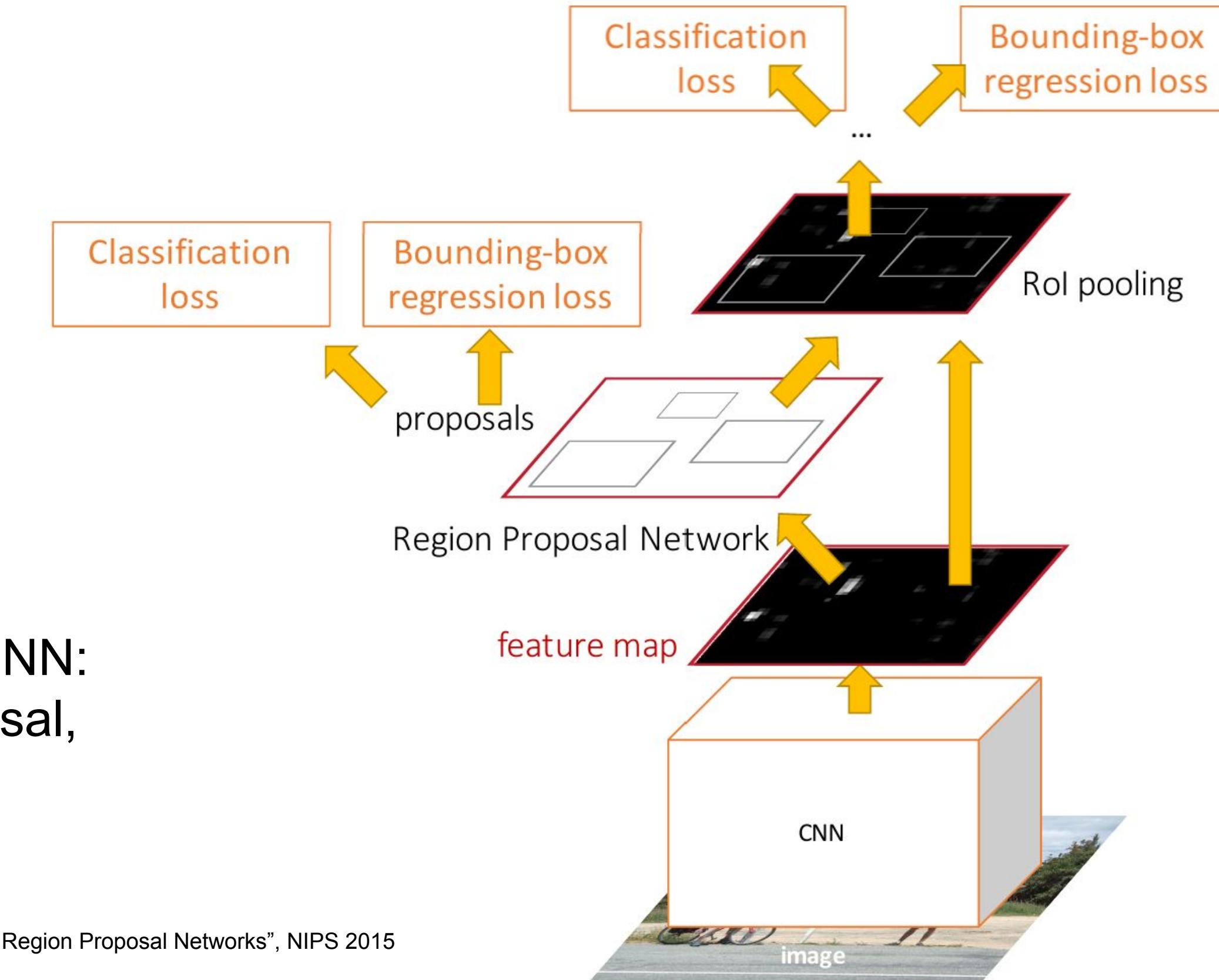
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
 He et al, "Spatial pyramid pooling in deep convolutional networks for visual recognition", ECCV 2014
 Girshick, "Fast R-CNN", ICCV 2015

Faster R-CNN

Make CNN do proposals!

Insert **Region Proposal Network (RPN)** to predict proposals from features

Otherwise same as Fast R-CNN:
Crop features for each proposal,
classify each one



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

Region Proposal Network



Input Image
(e.g. $3 \times 640 \times 480$)



CNN

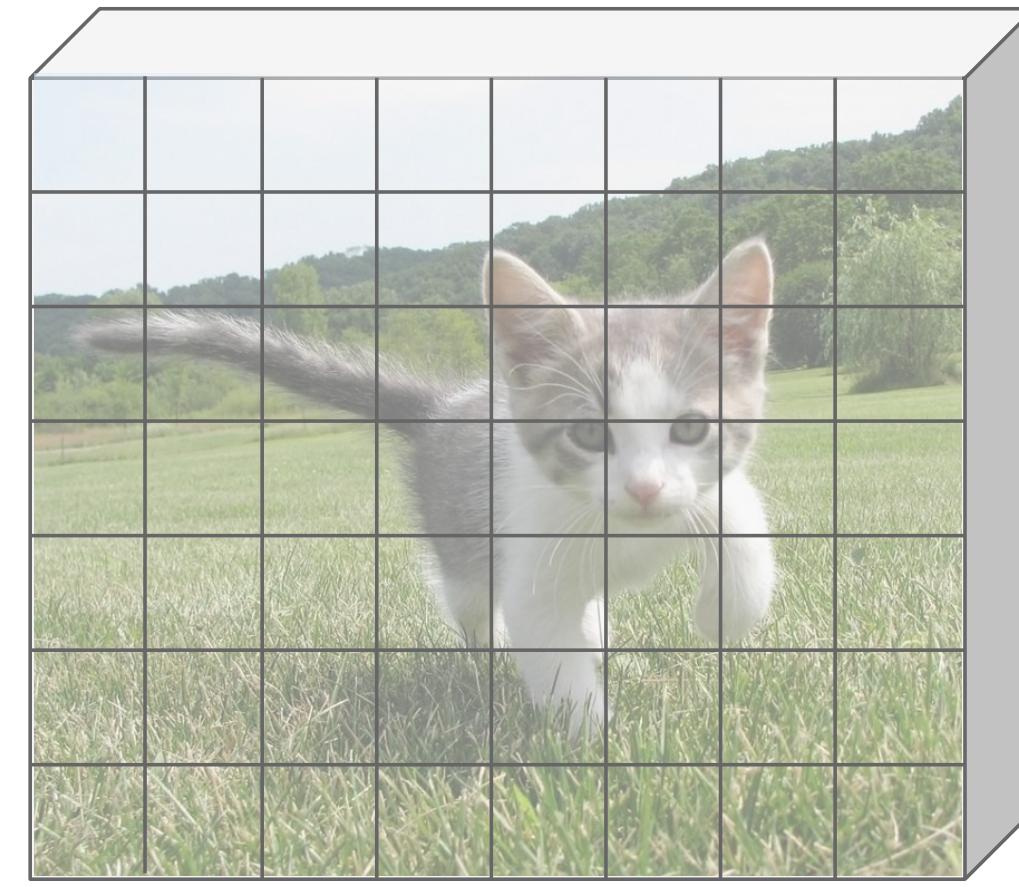
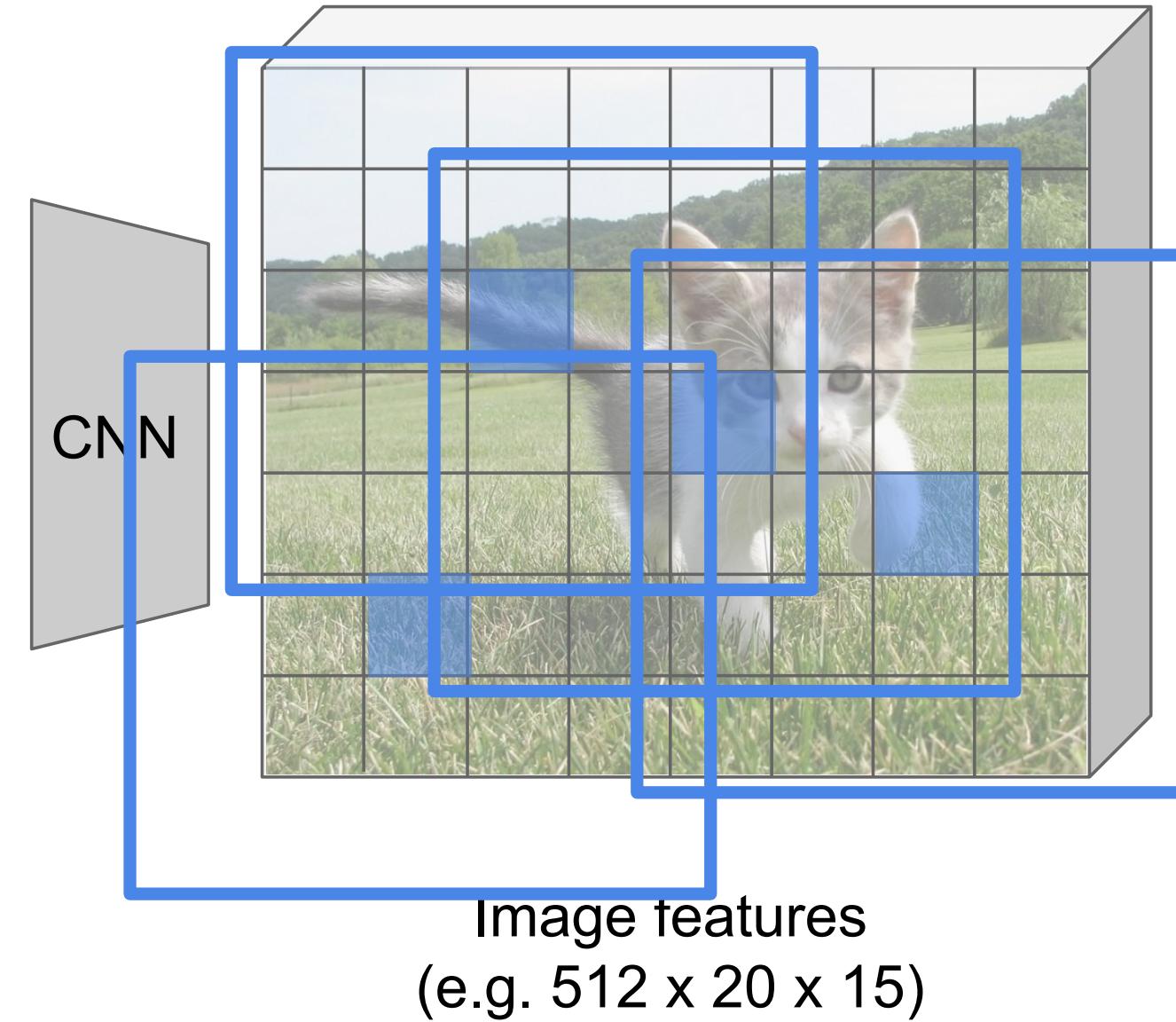


Image features
(e.g. $512 \times 20 \times 15$)

Region Proposal Network



Input Image
(e.g. $3 \times 640 \times 480$)

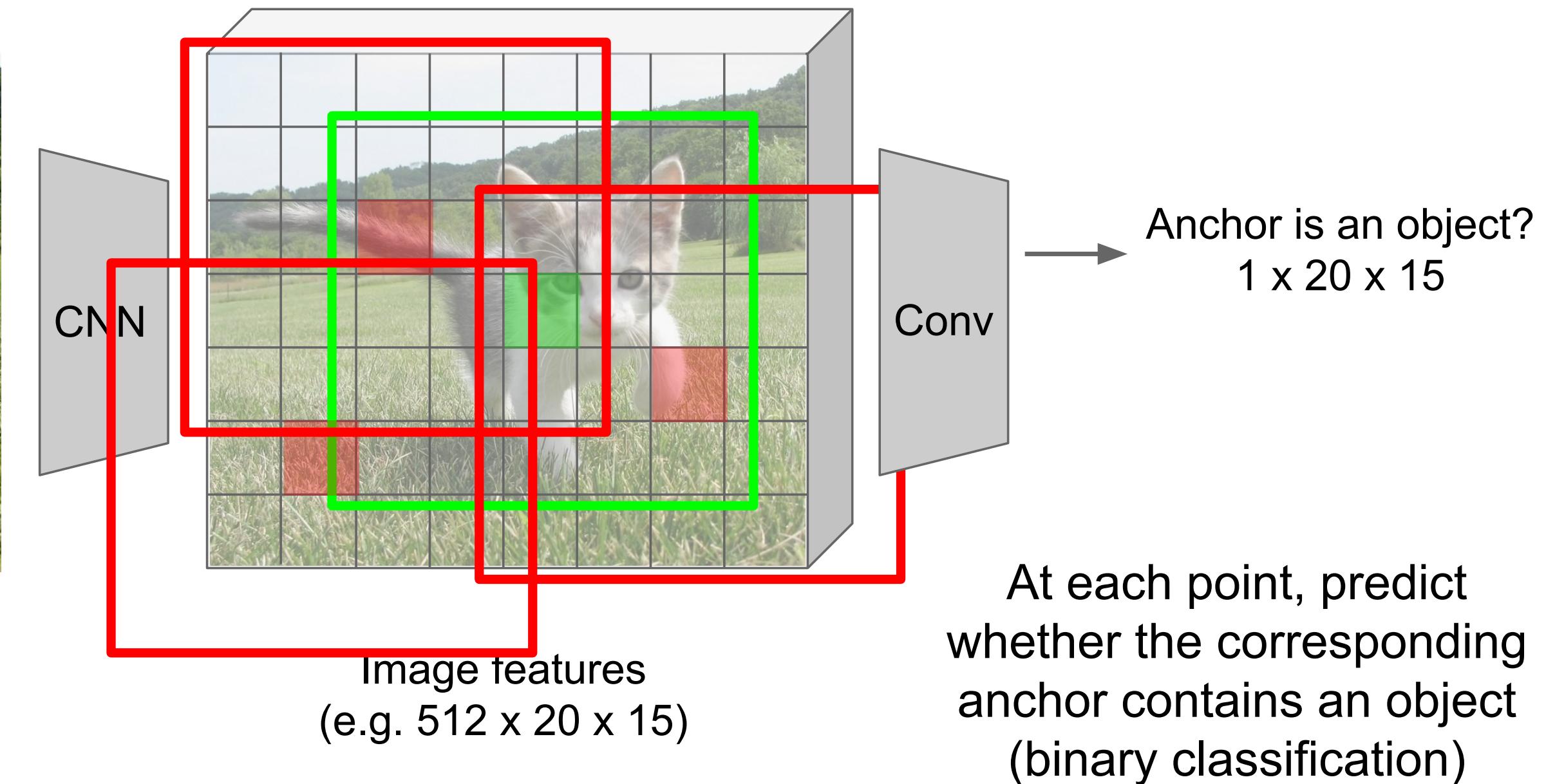


Imagine an **anchor box**
of fixed size at each
point in the feature map

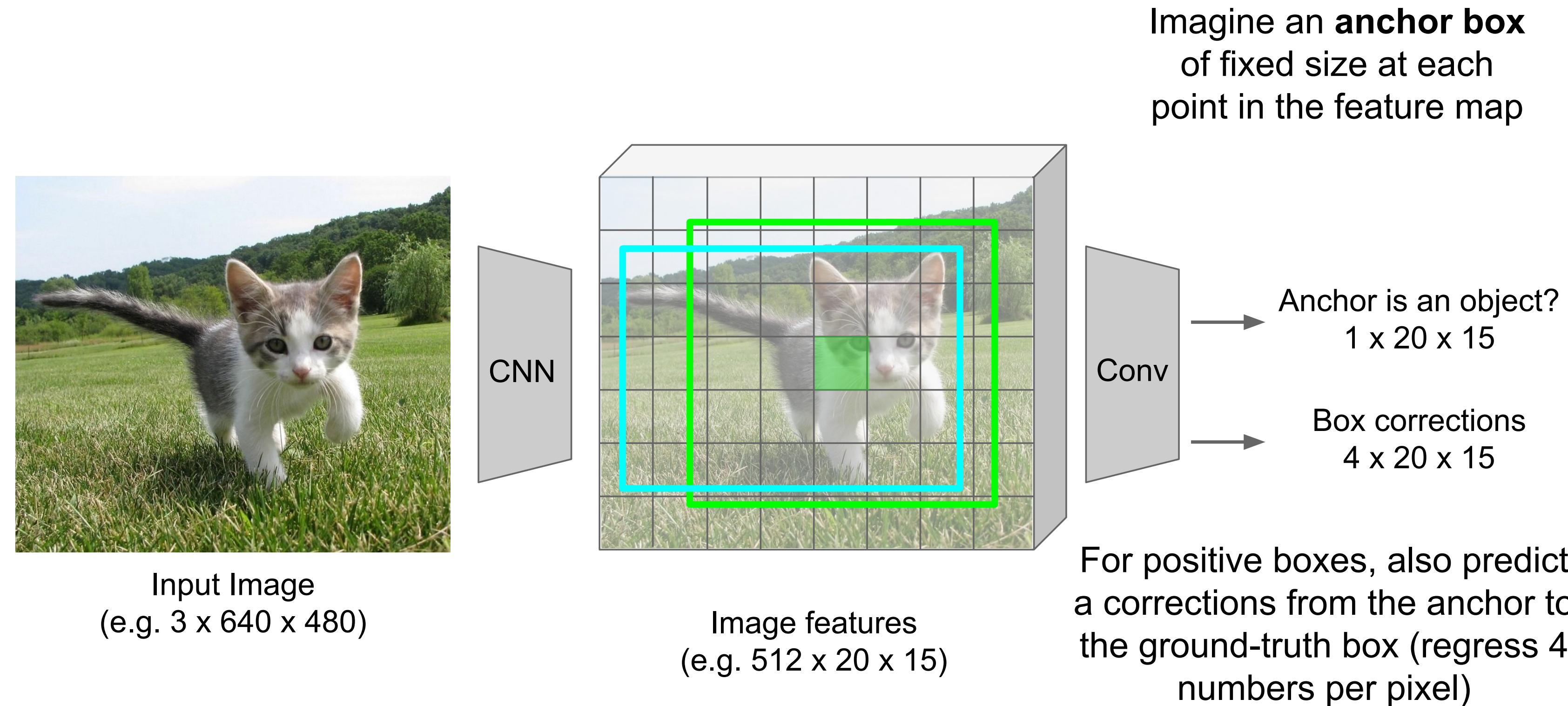
Region Proposal Network



Input Image
(e.g. $3 \times 640 \times 480$)



Region Proposal Network

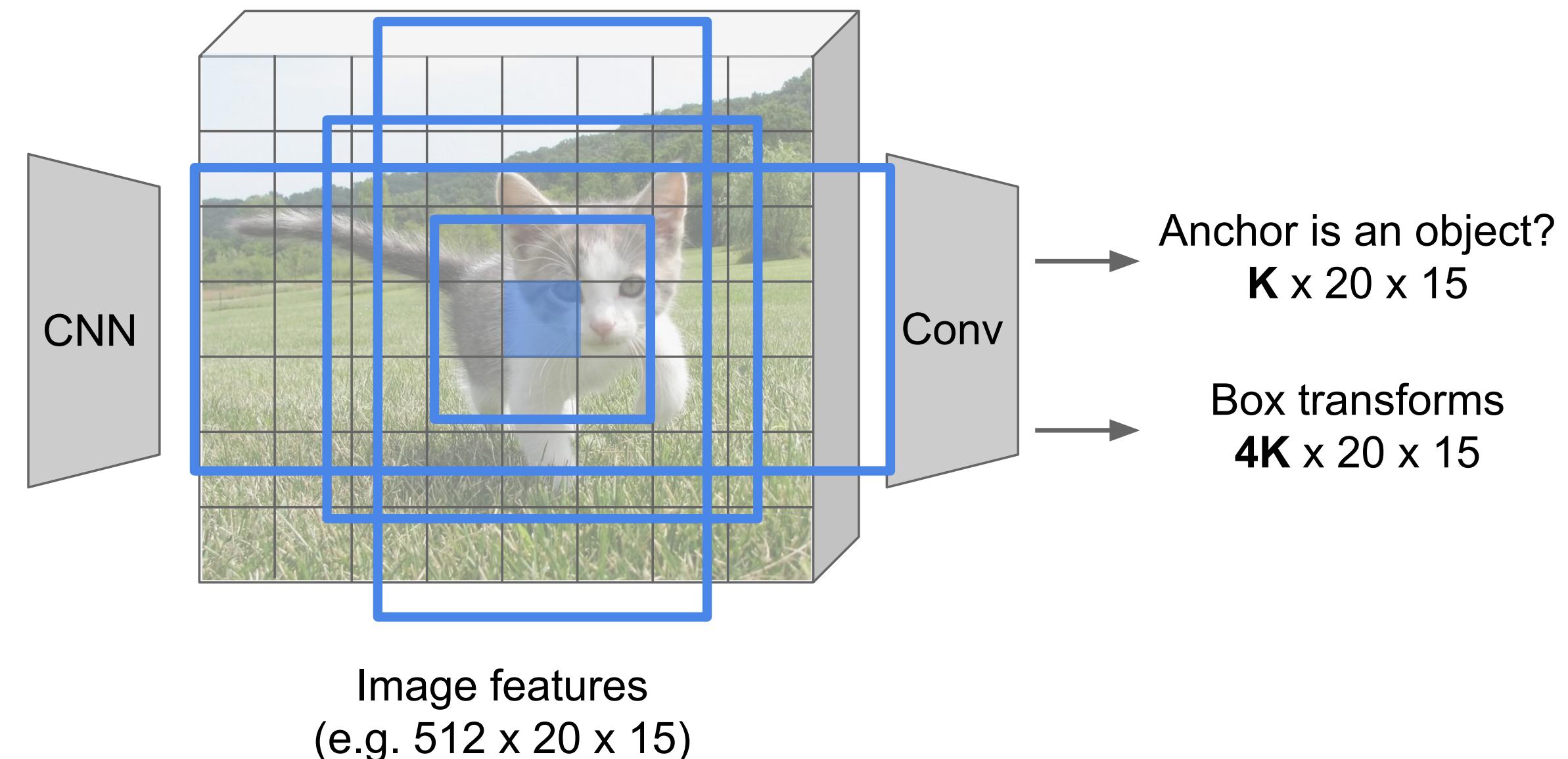


Region Proposal Network

In practice use K different anchor boxes of different size / scale at each point



Input Image
(e.g. $3 \times 640 \times 480$)



Region Proposal Network



Input Image
(e.g. $3 \times 640 \times 480$)

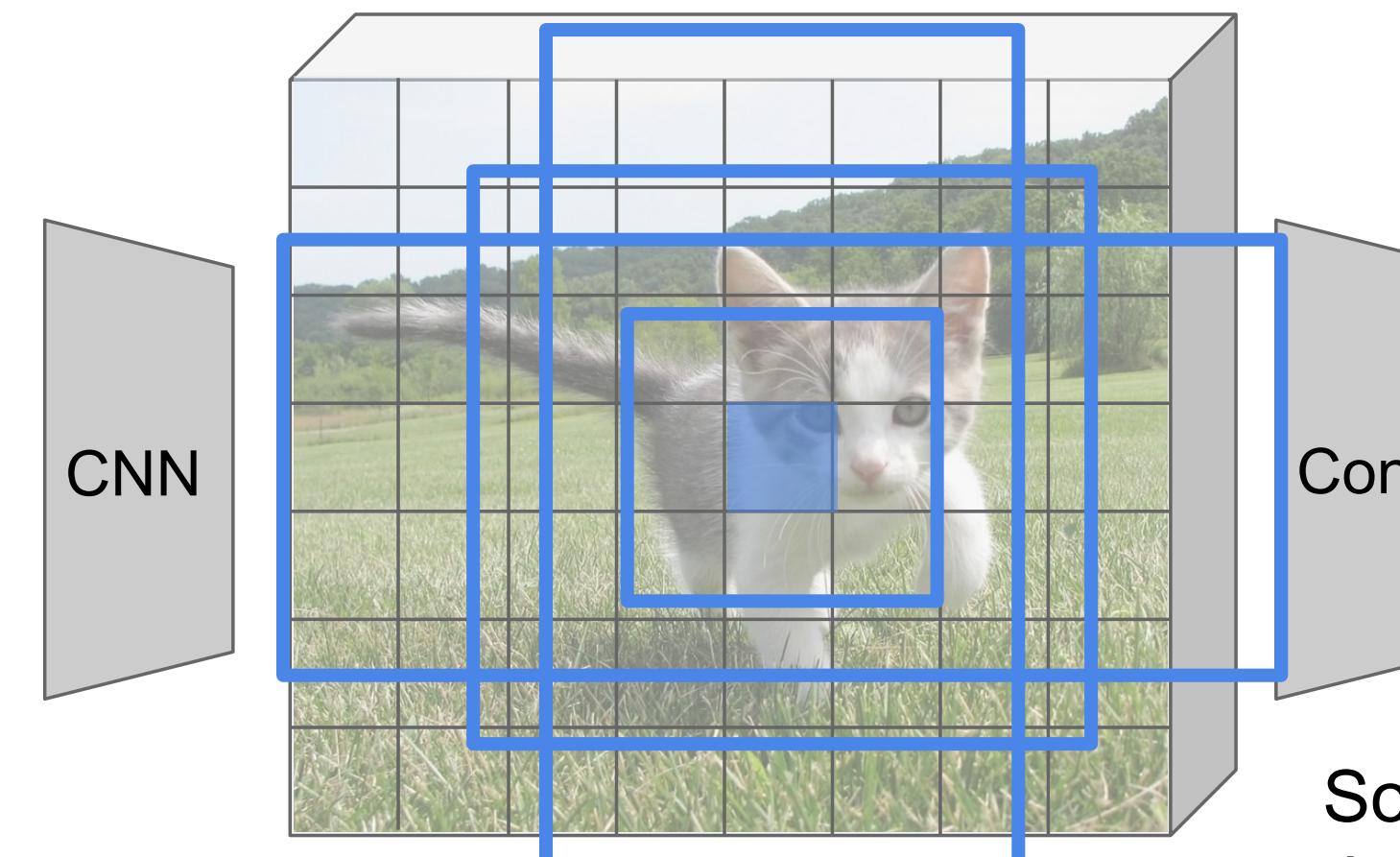


Image features
(e.g. $512 \times 20 \times 15$)

In practice use K different anchor boxes of different size / scale at each point

Anchor is an object?
 $K \times 20 \times 15$

Box transforms
 $4K \times 20 \times 15$

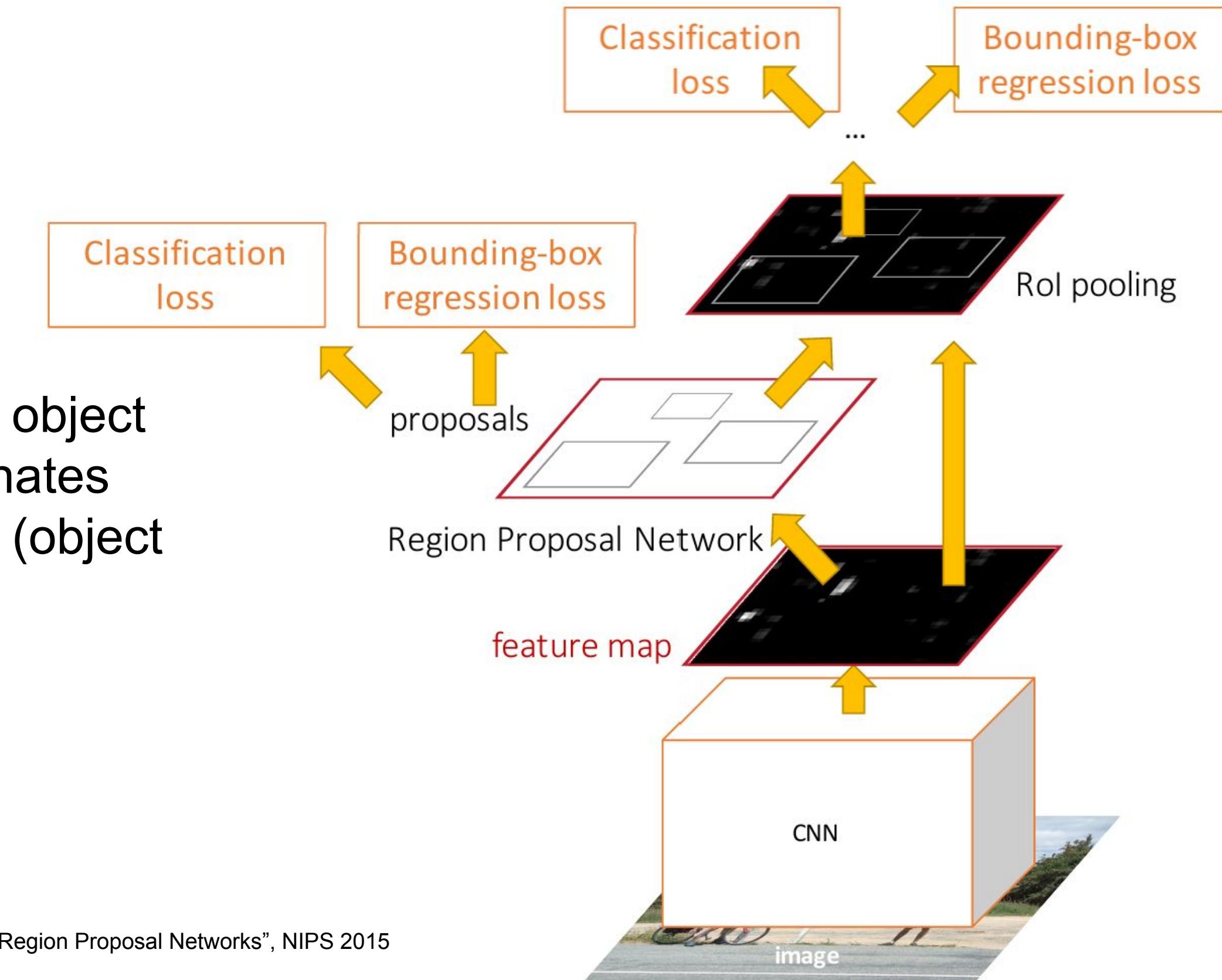
Sort the $K \times 20 \times 15$ boxes by their “objectness” score, take top ~ 300 as our proposals

Faster R-CNN

Make CNN do proposals!

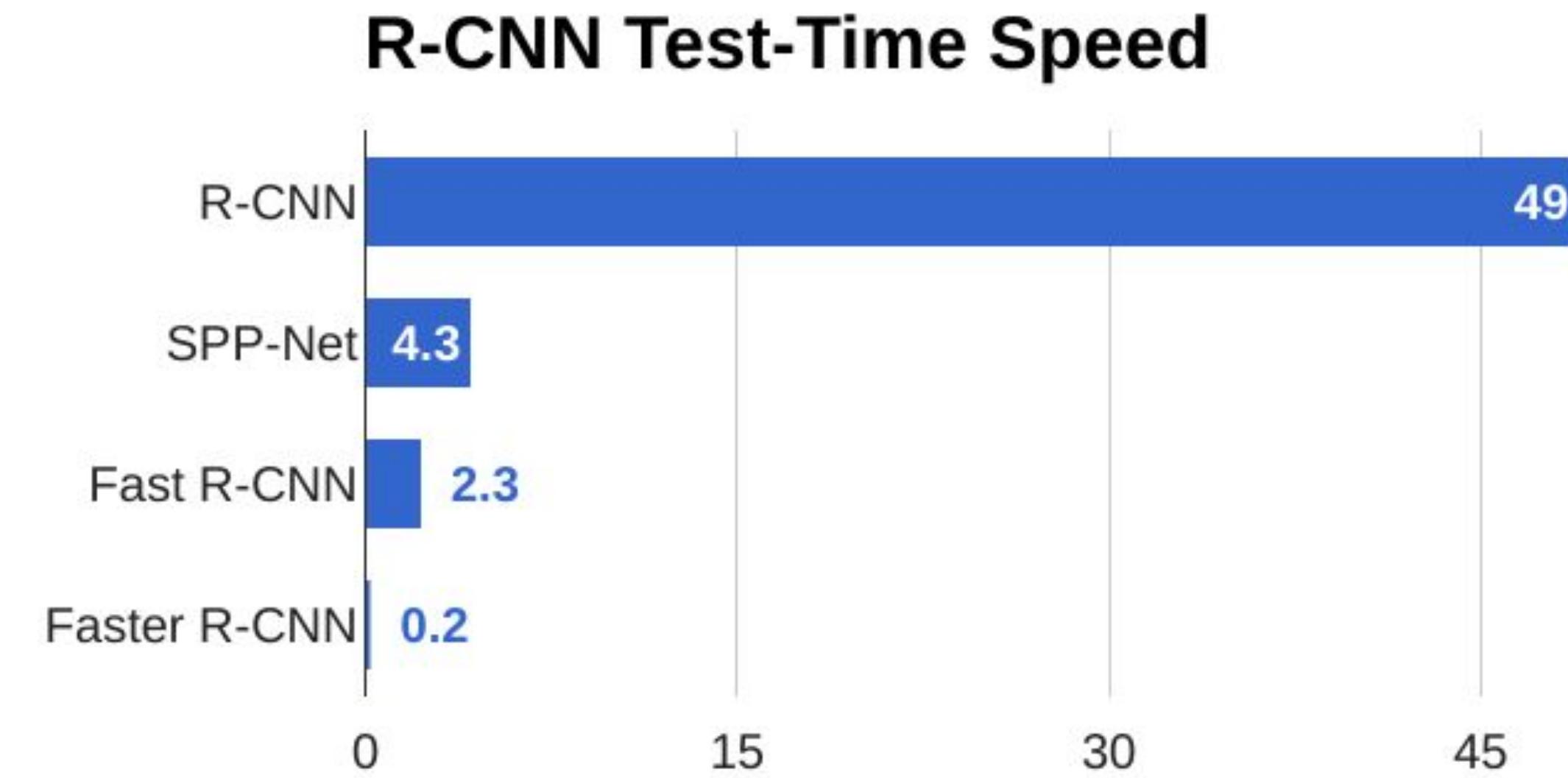
Jointly train with 4 losses:

1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates



Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

Faster R-CNN: Make CNN do proposals



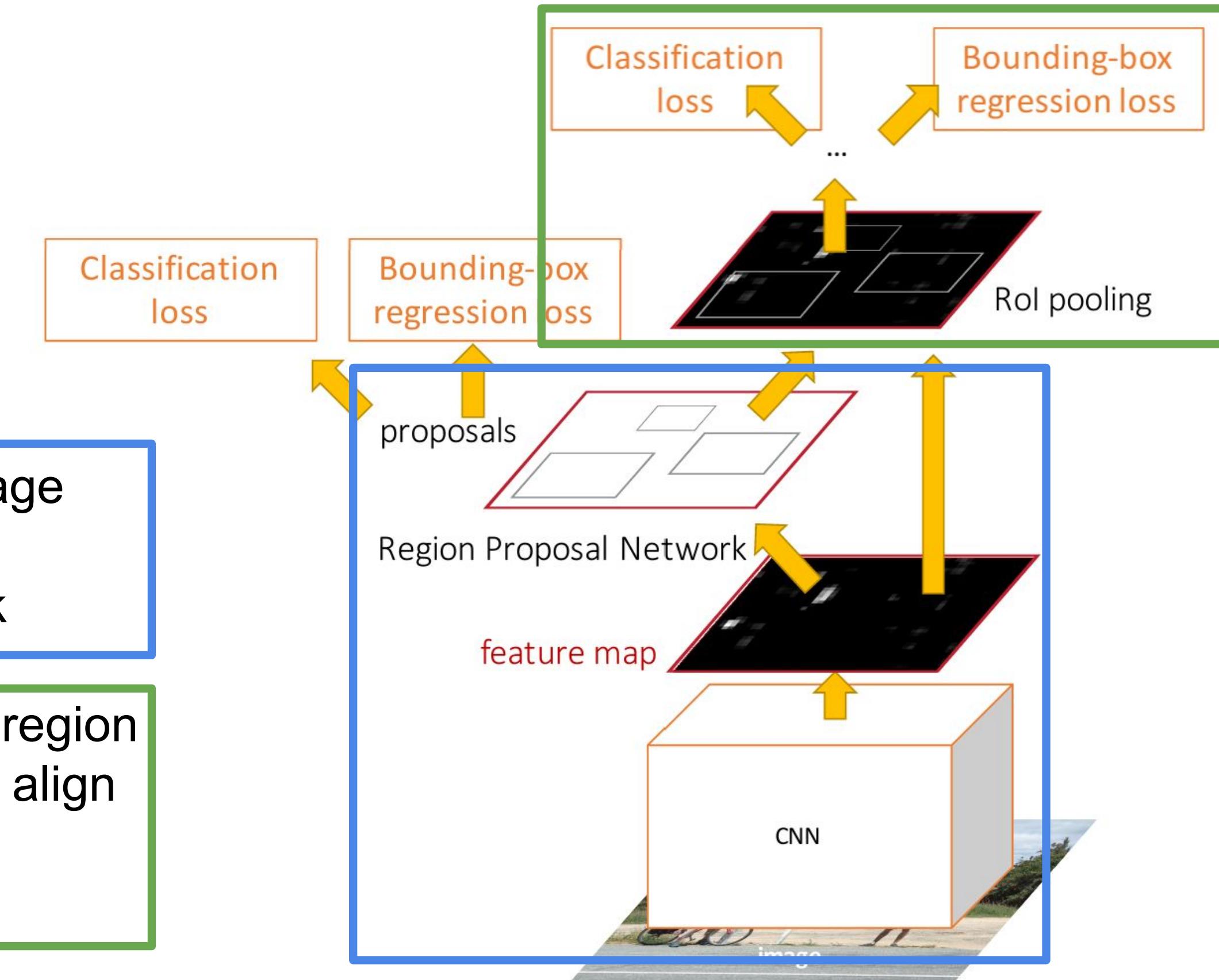
Faster R-CNN

Faster R-CNN: Make CNN do proposals!

Faster R-CNN is a
Two-stage object detector

- First stage: Run once per image
 - Backbone network
 - Region proposal network

- Second stage: Run once per region
 - Crop features: RoI pool / align
 - Predict object class
 - Prediction bbox offset



Faster R-CNN

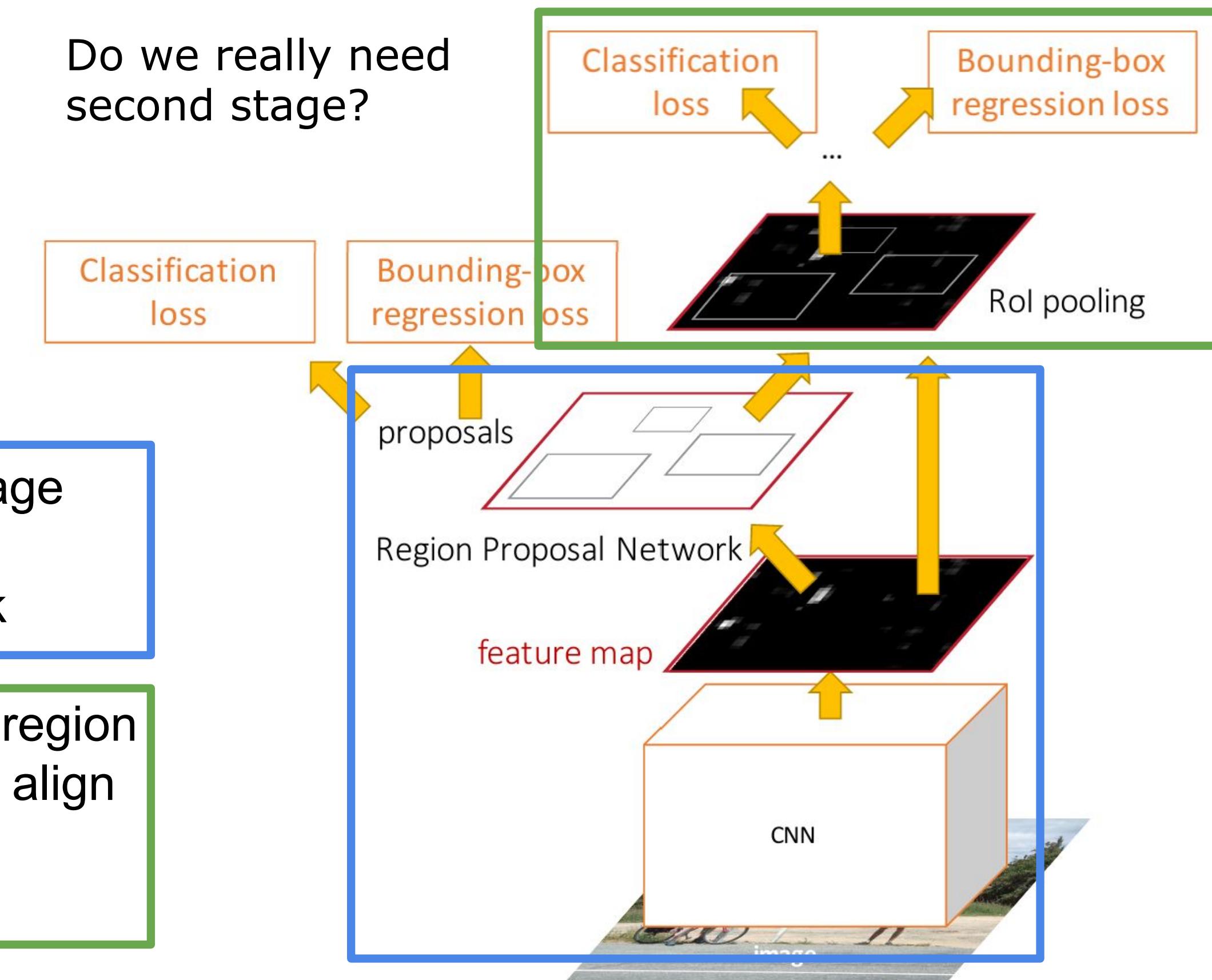
Faster R-CNN: Make CNN do proposals!

Faster R-CNN is a
Two-stage object detector

- First stage: Run once per image
- Backbone network
 - Region proposal network

- Second stage: Run once per region
- Crop features: RoI pool / align
 - Predict object class
 - Prediction bbox offset

Do we really need
second stage?

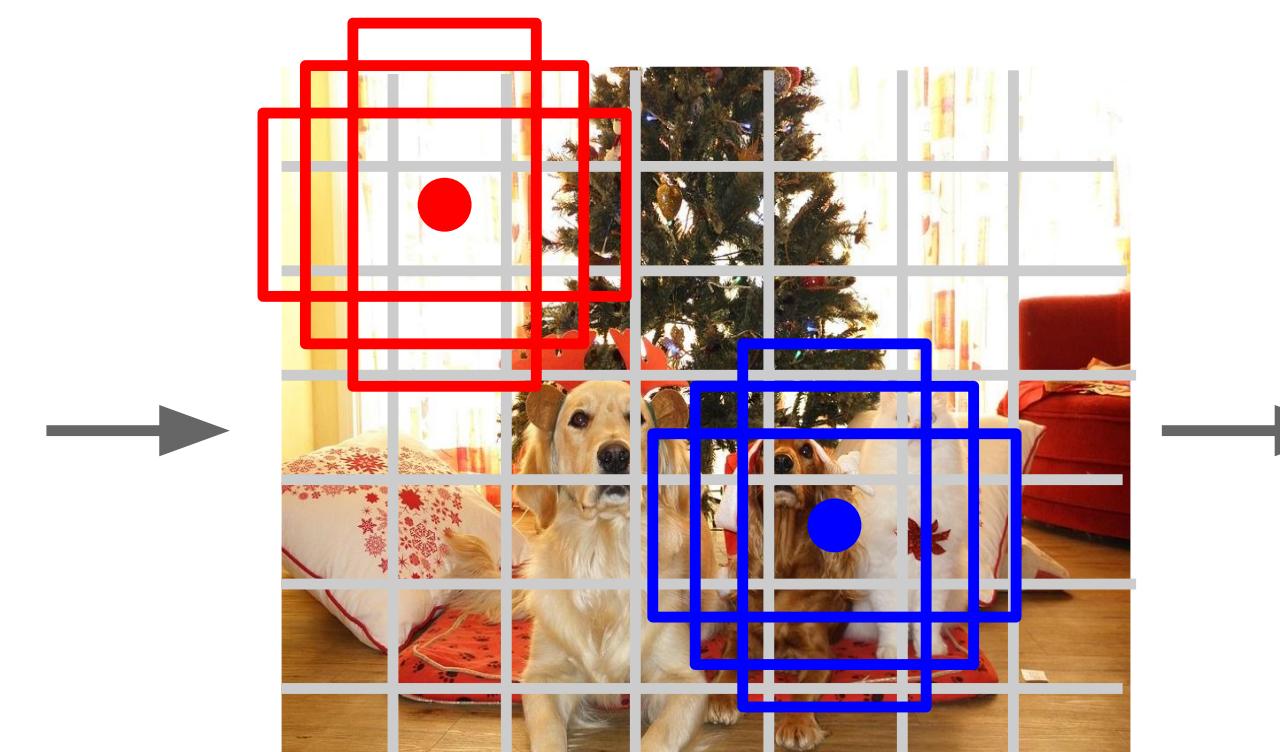


Single-Stage Object Detectors: YOLO / SSD / RetinaNet



Input image
 $3 \times H \times W$

Redmon et al, "You Only Look Once:
Unified, Real-Time Object Detection", CVPR 2016
Liu et al, "SSD: Single-Shot MultiBox Detector", ECCV 2016
Lin et al, "Focal Loss for Dense Object Detection", ICCV 2017



Divide image into grid

7×7

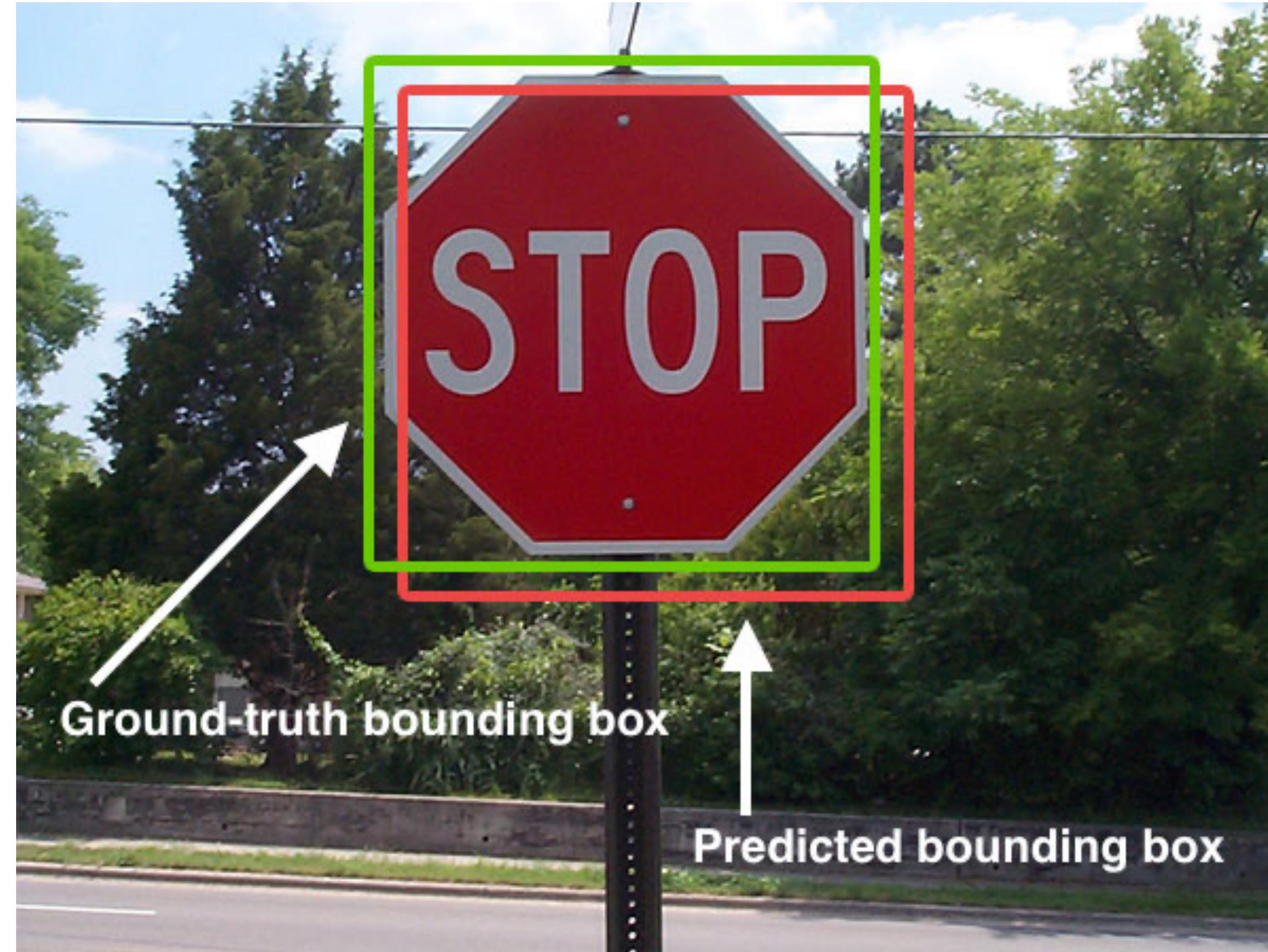
Image a set of **base boxes**
centered at each grid cell
Here $B = 3$

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers: $(dx, dy, dh, dw, \text{confidence})$
- Predict scores for each of C classes (including background as a class)
- Looks a lot like RPN, but category-specific!

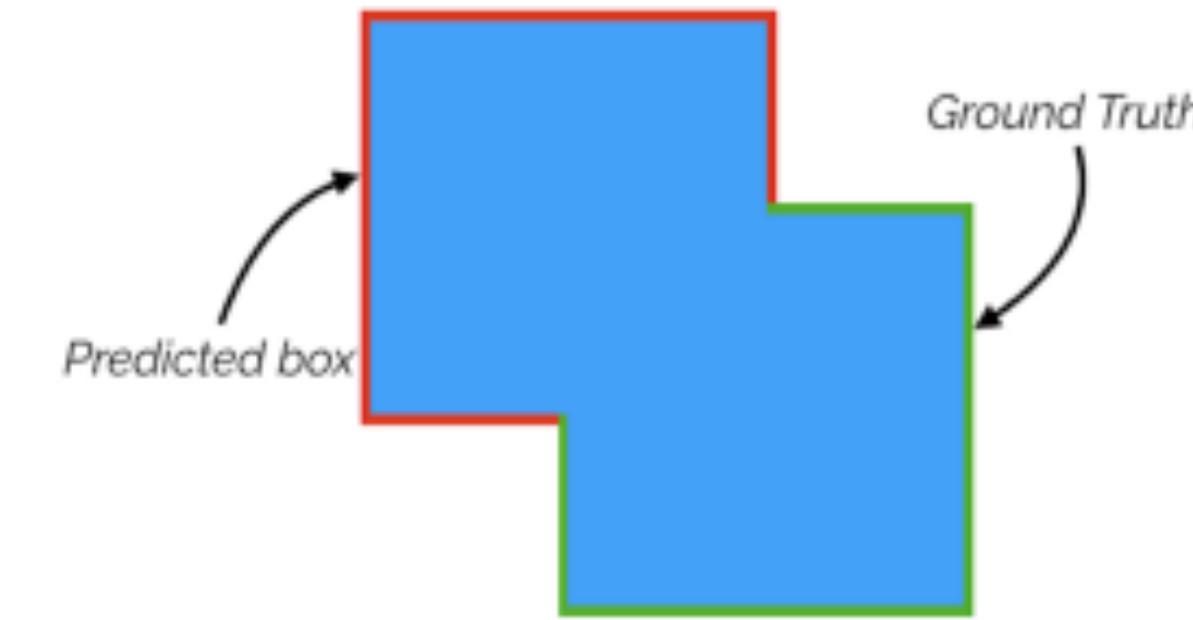
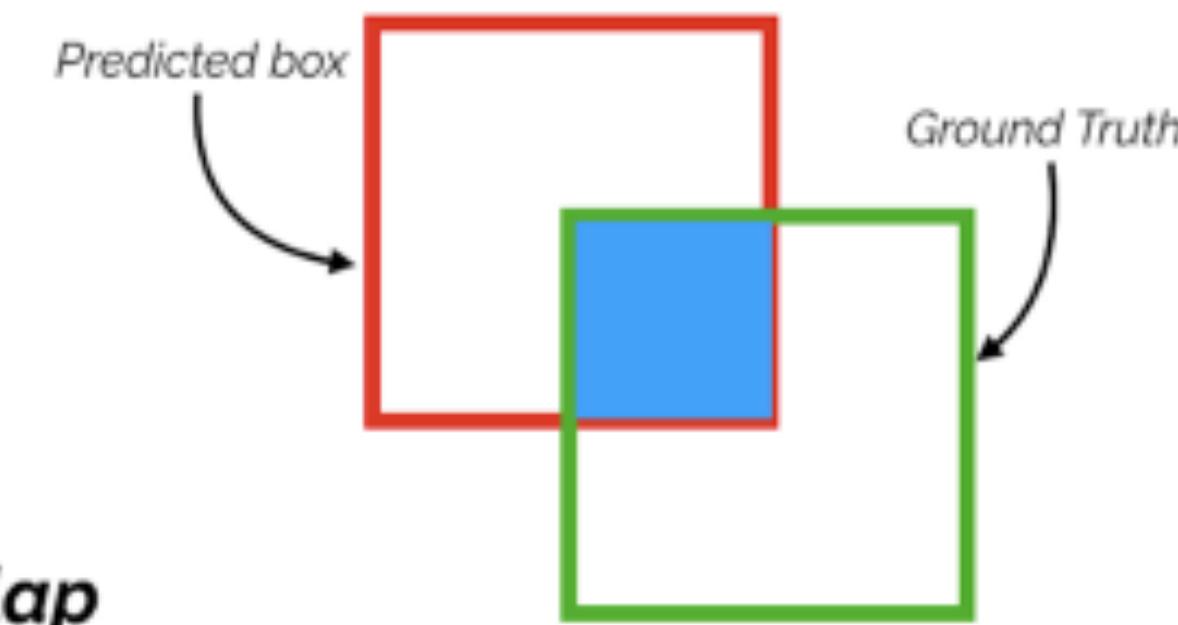
Output:
 $7 \times 7 \times (5 * B + C)$

Assessment of Object Detection

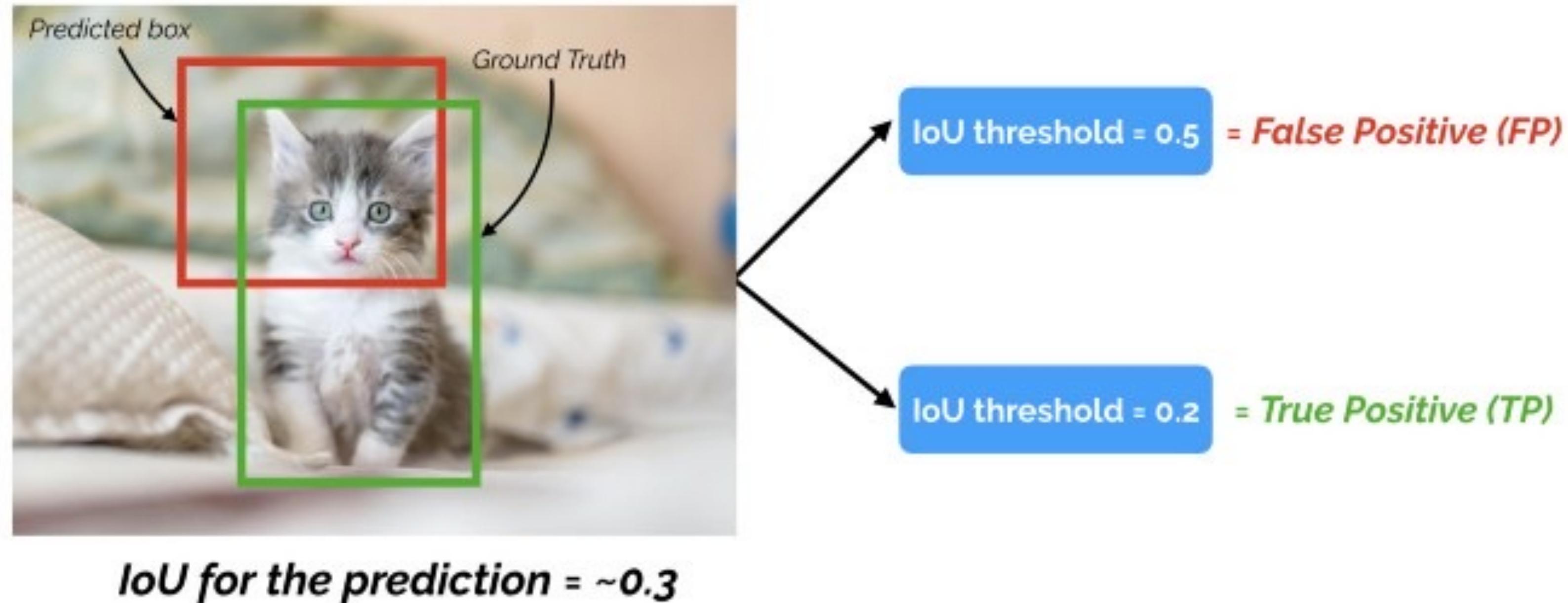


Assessment of Object Detection (Jaccard Index)

$$\text{Intersection over Union (IoU)} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$



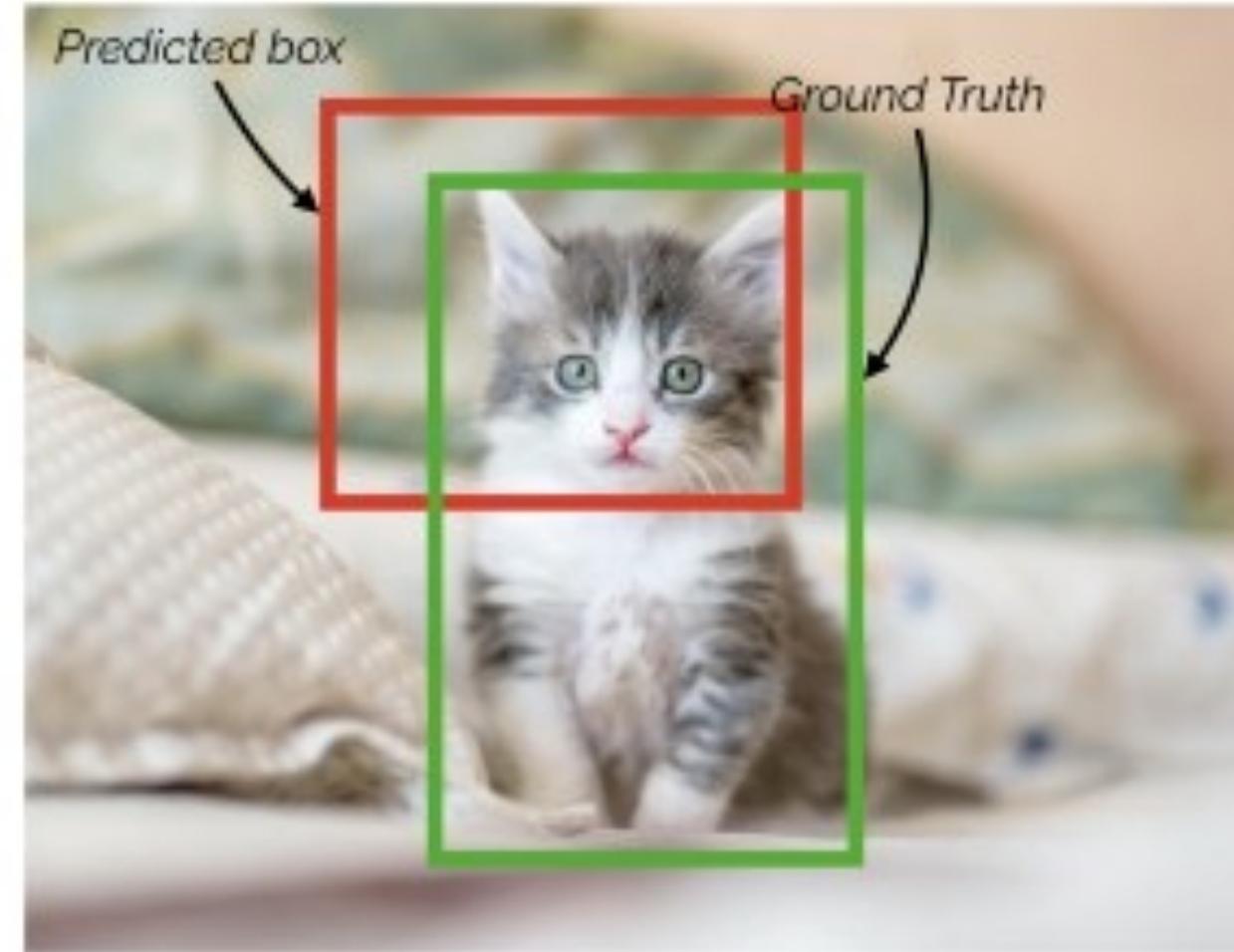
Assessment of Object Detection: Intersection of Union



Assessment of Object Detection: Intersection of Union

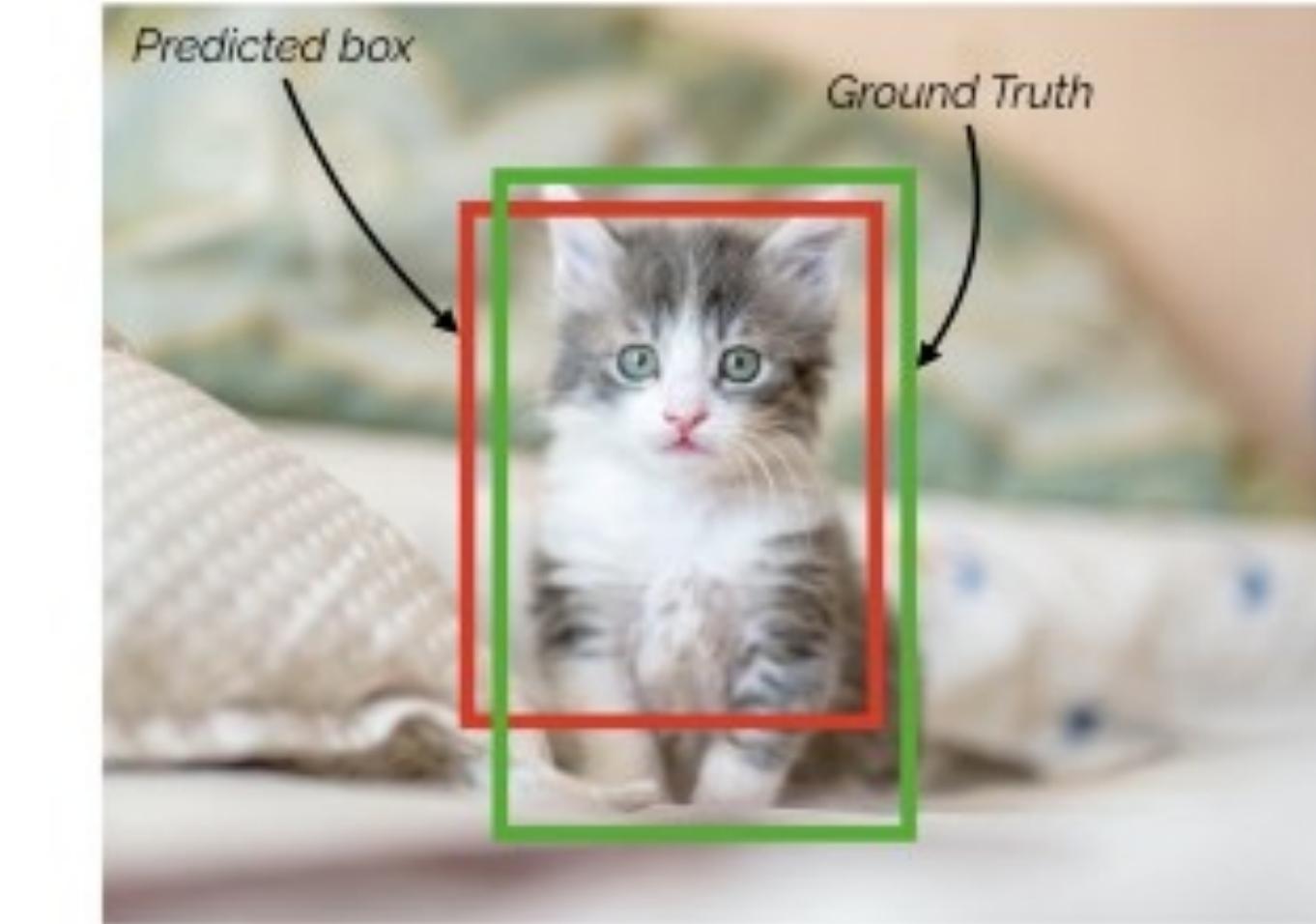
If IoU threshold = 0.5

False Positive (FP)



$IoU = \sim 0.3$

True Positive (TP)



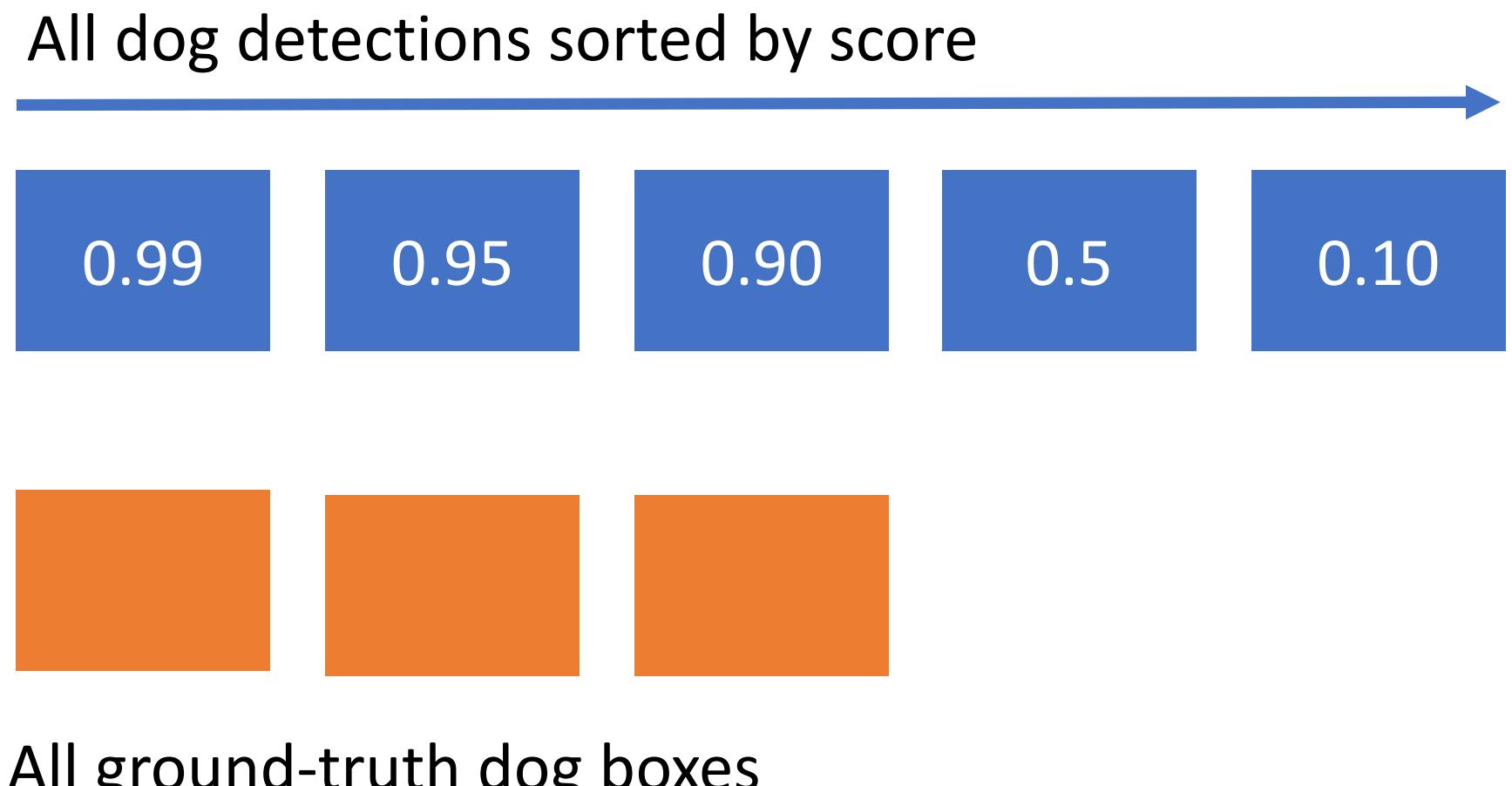
$IoU = \sim 0.7$

Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) =
area under Precision vs Recall Curve

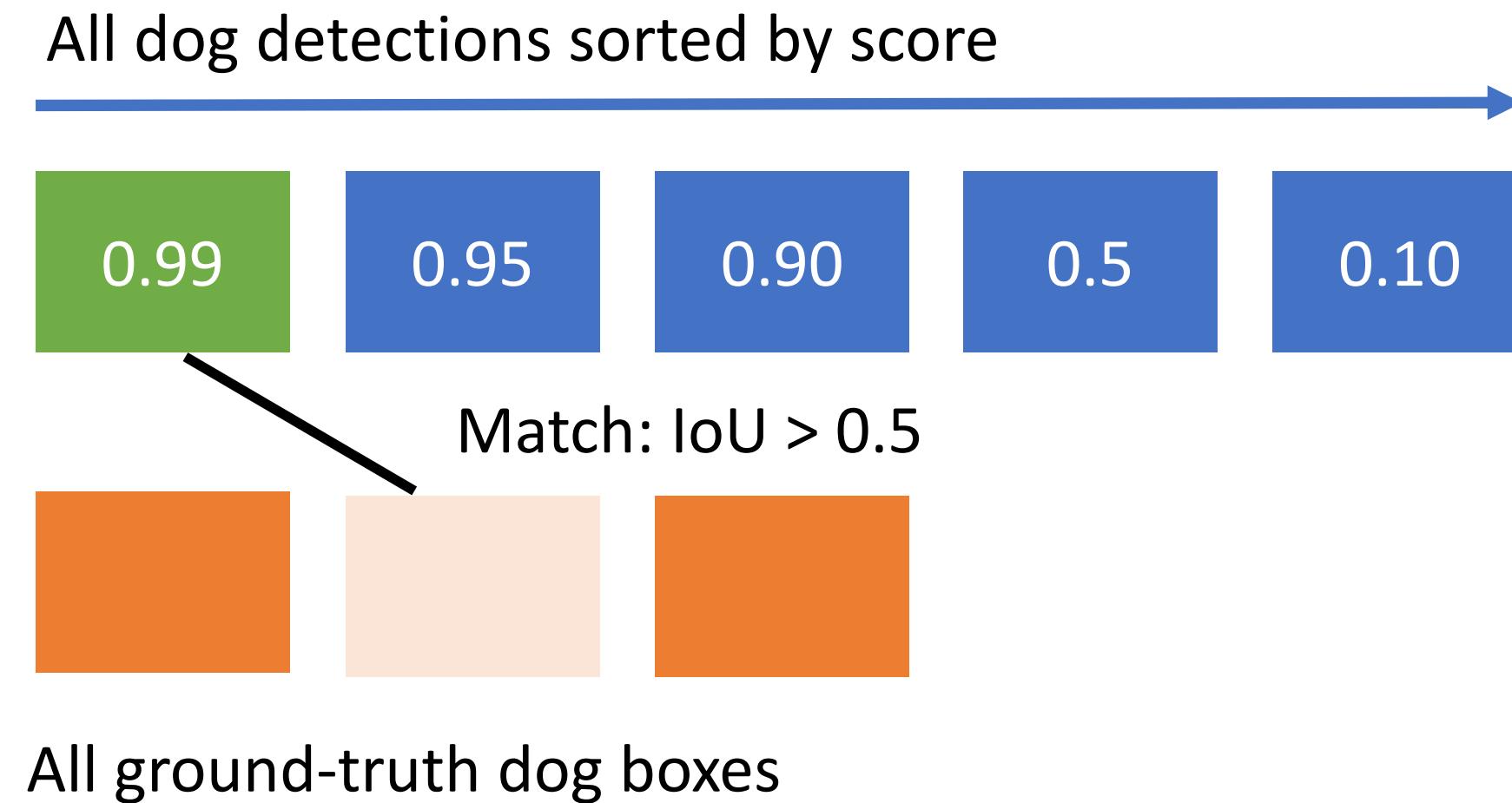
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)



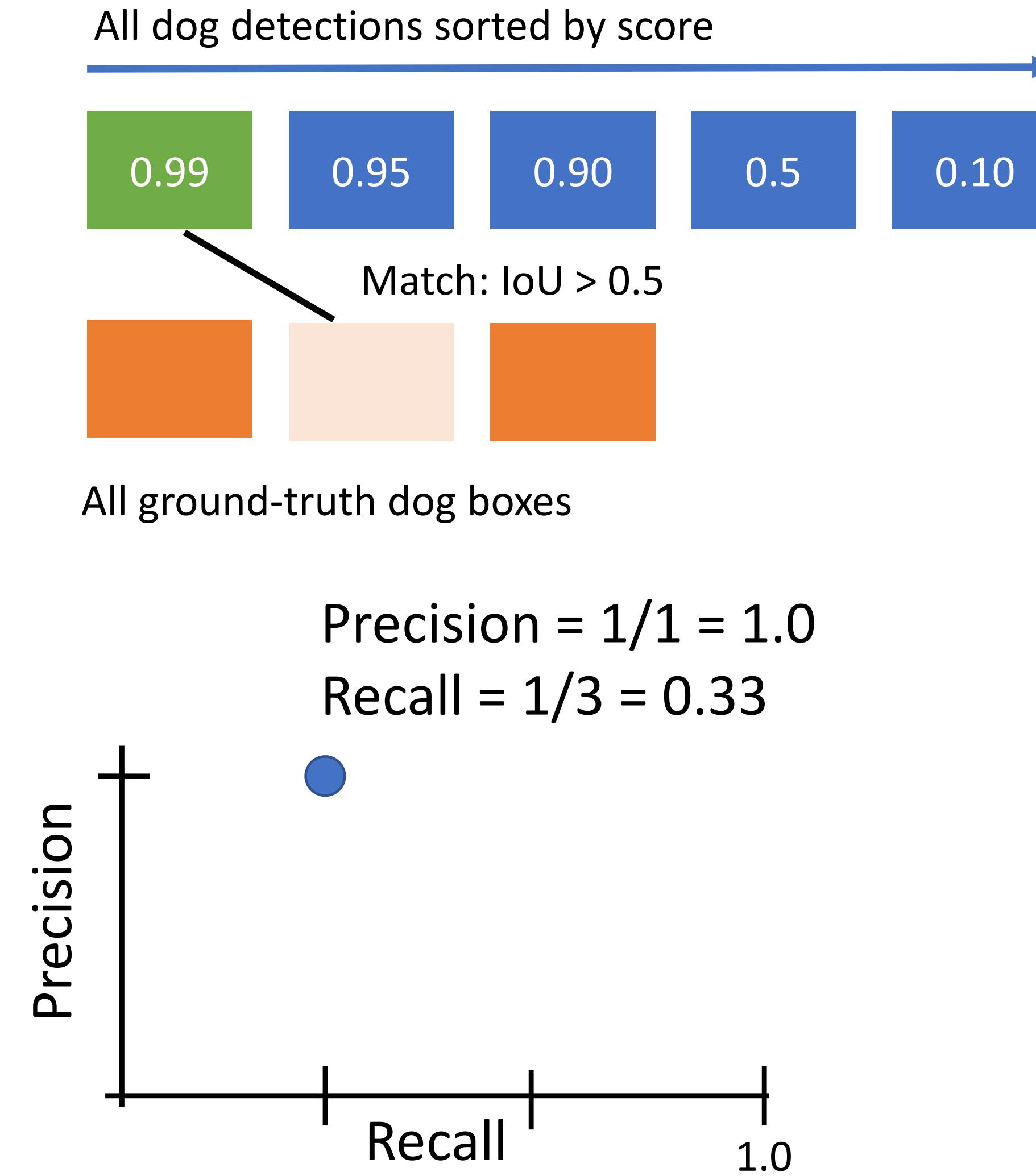
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with $\text{IoU} > 0.5$, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative



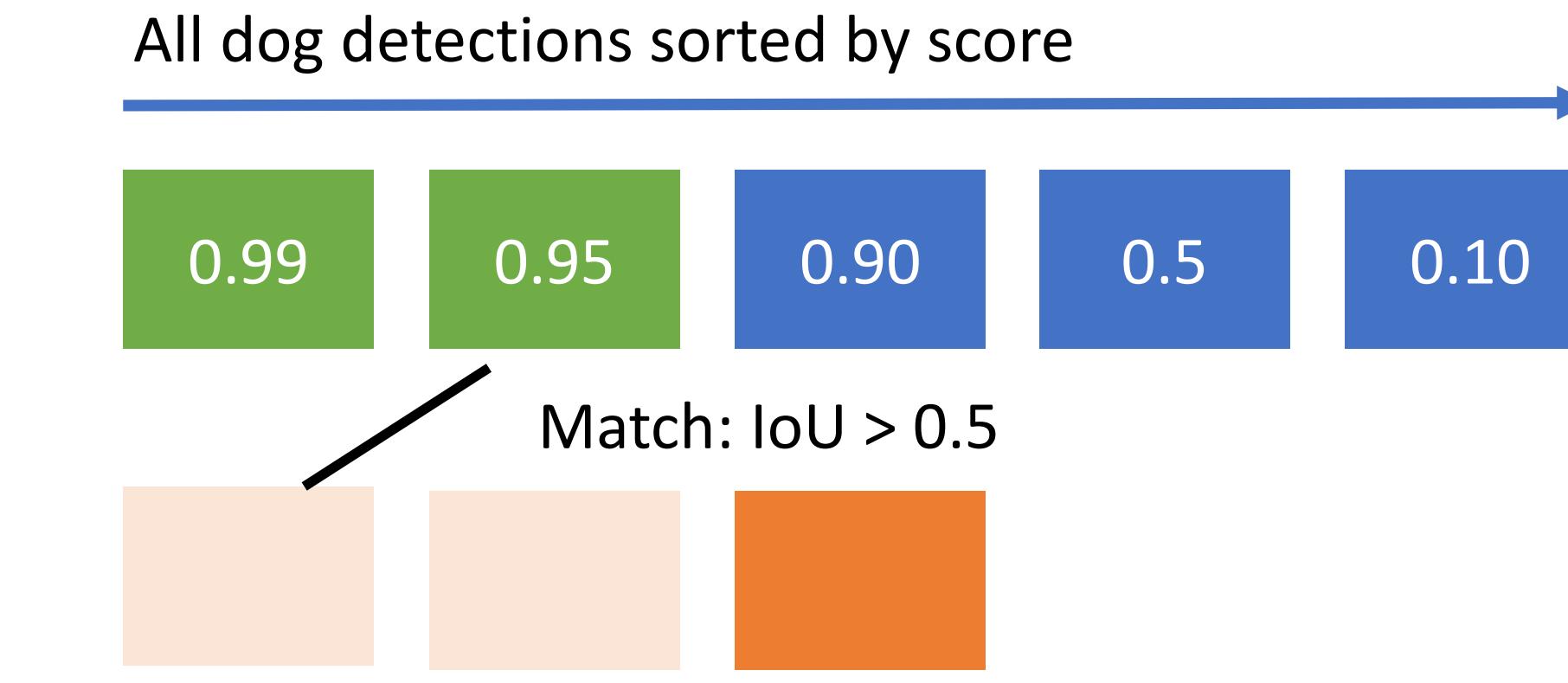
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with $\text{IoU} > 0.5$, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve



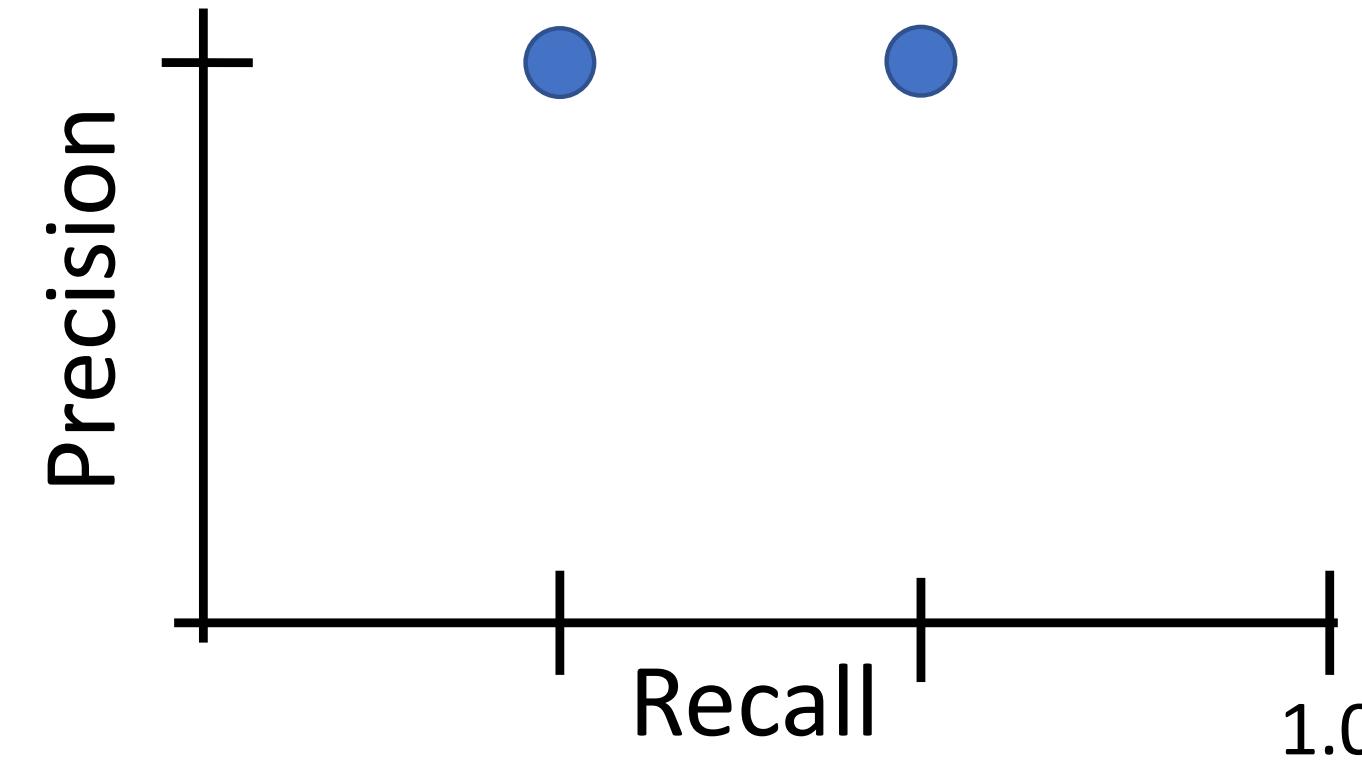
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with $\text{IoU} > 0.5$, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve



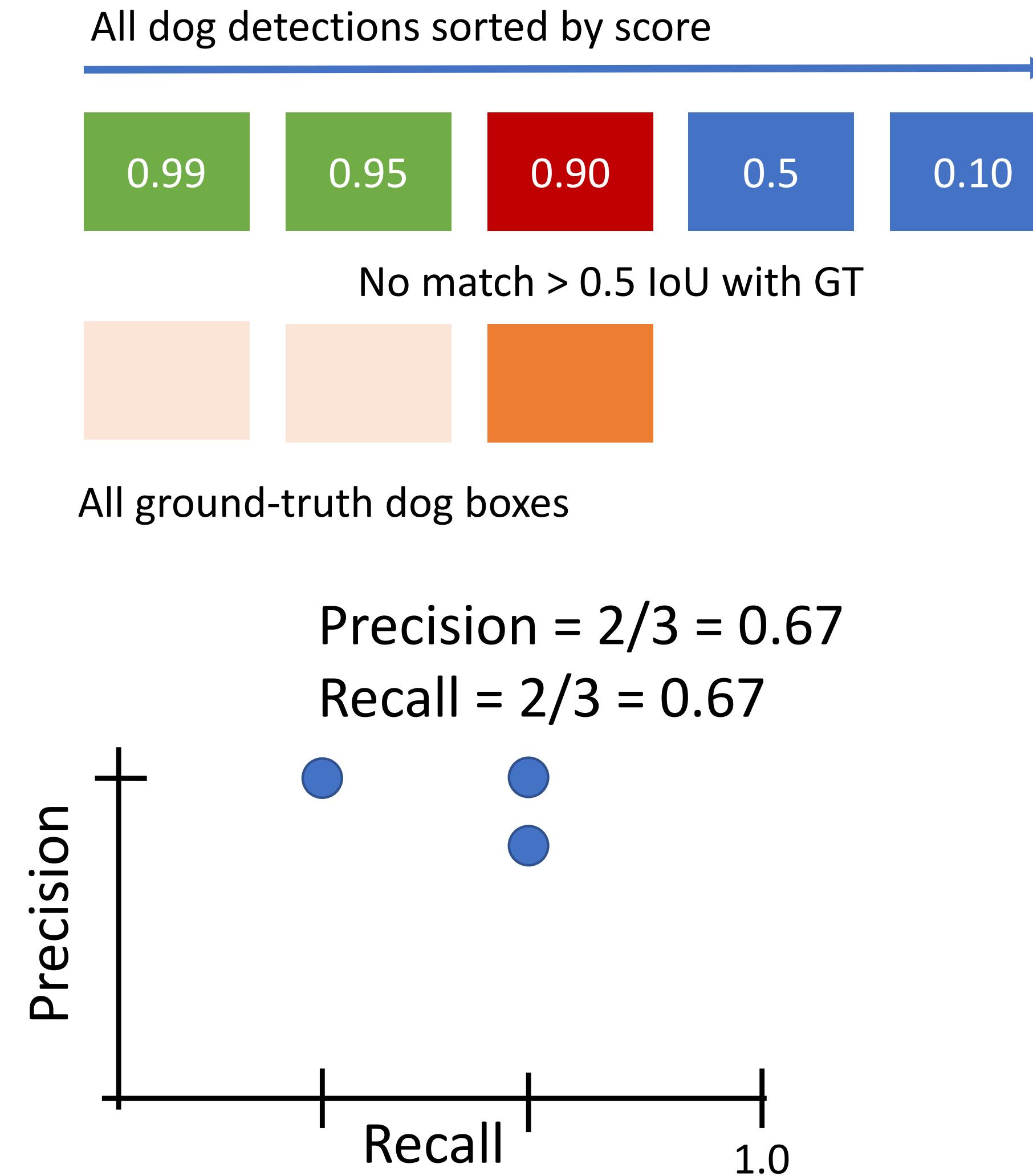
All ground-truth dog boxes

$\text{Precision} = 2/2 = 1.0$
 $\text{Recall} = 2/3 = 0.67$



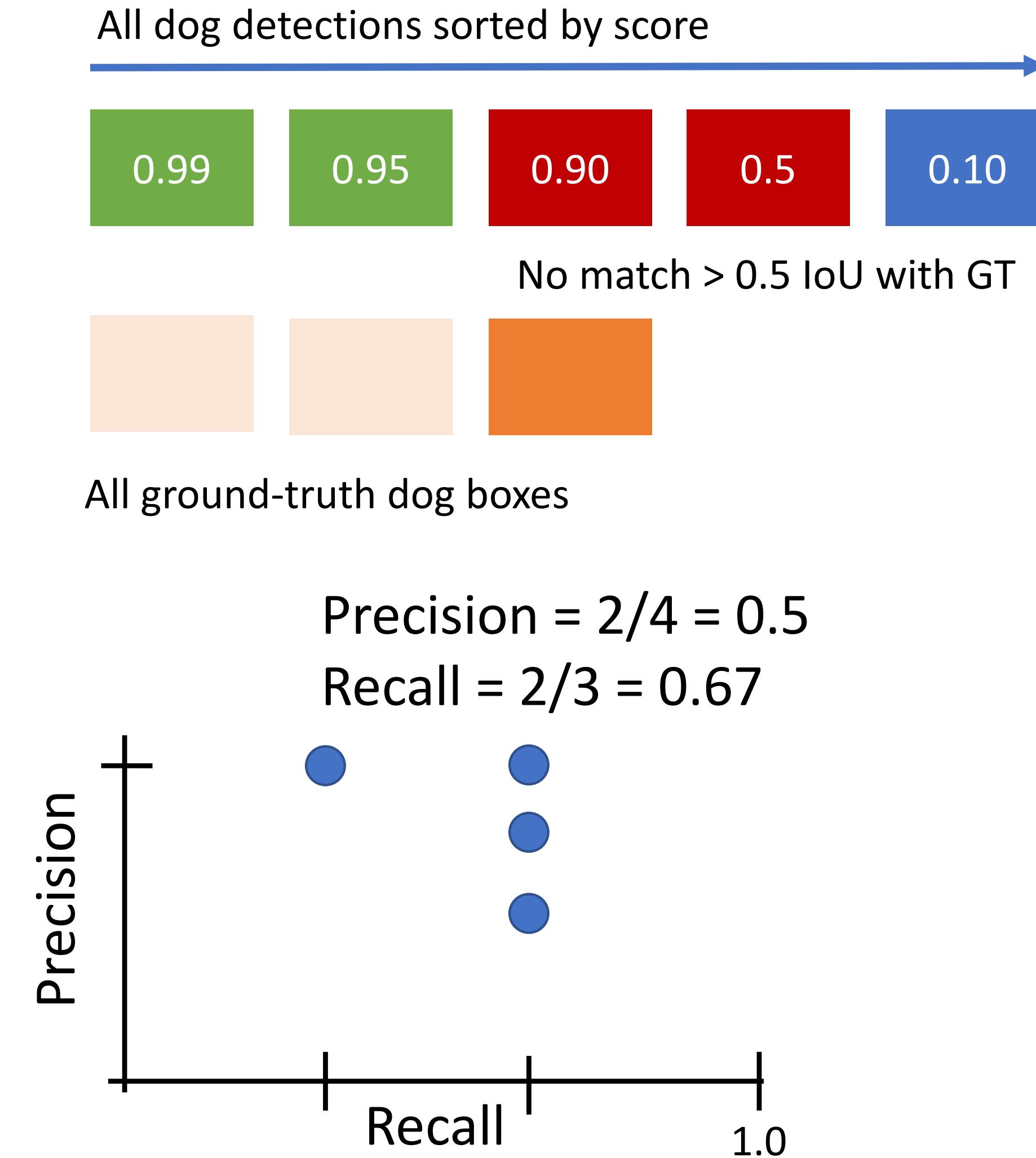
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve



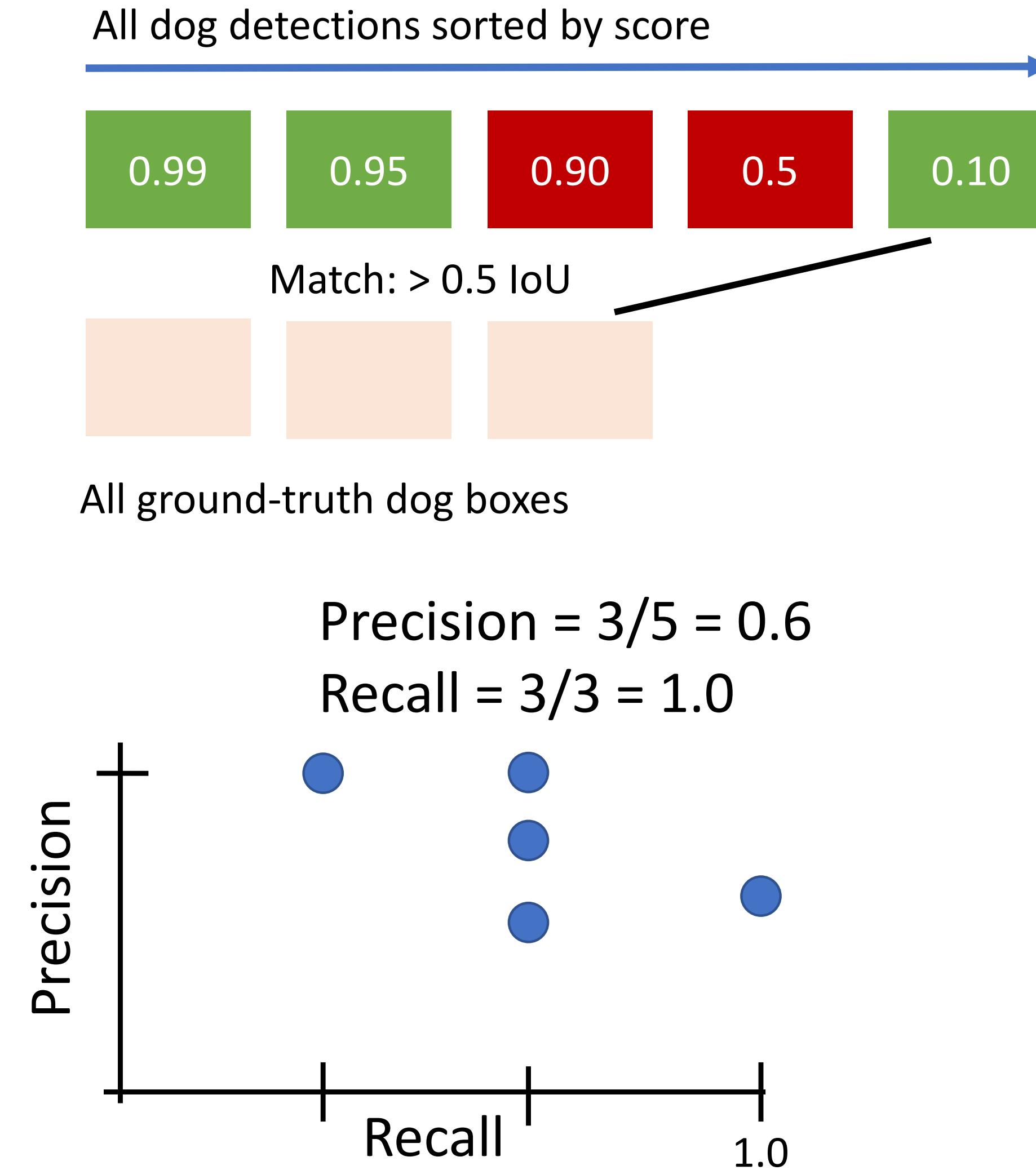
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve



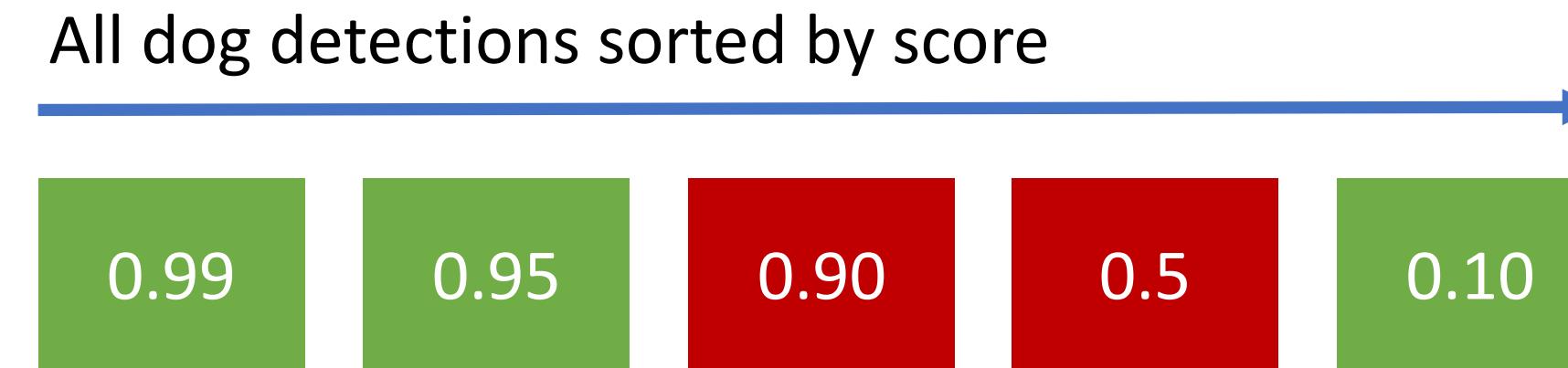
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve

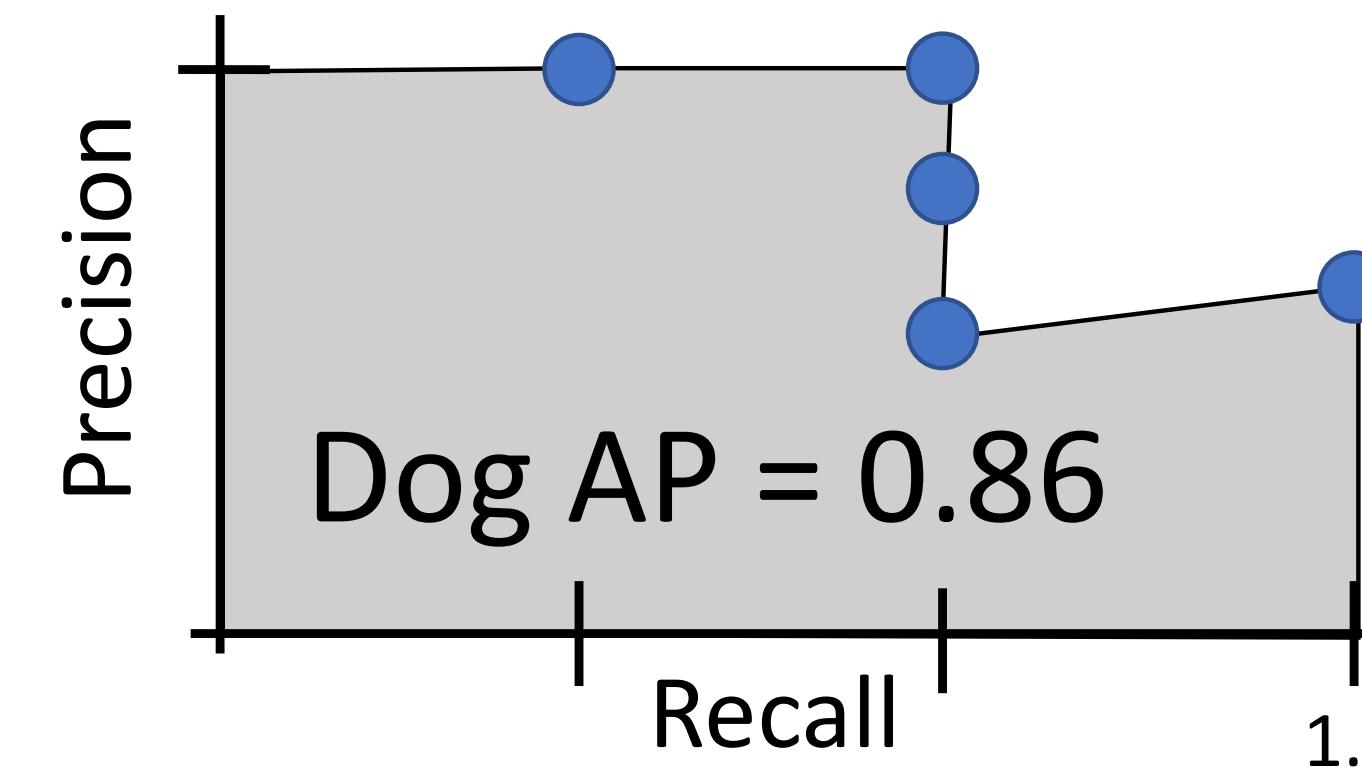


Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with $\text{IoU} > 0.5$, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve
 2. Average Precision (AP) = area under PR curve



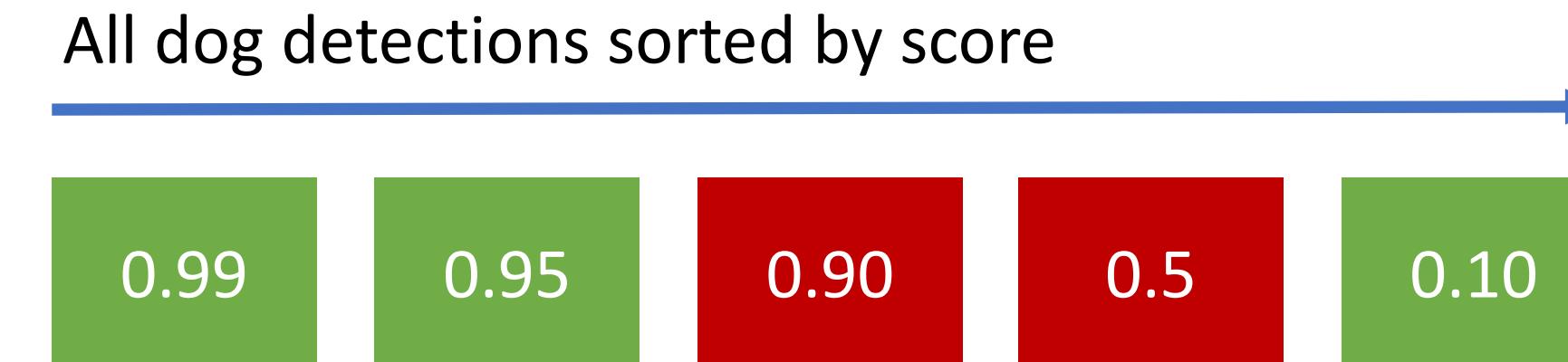
All ground-truth dog boxes



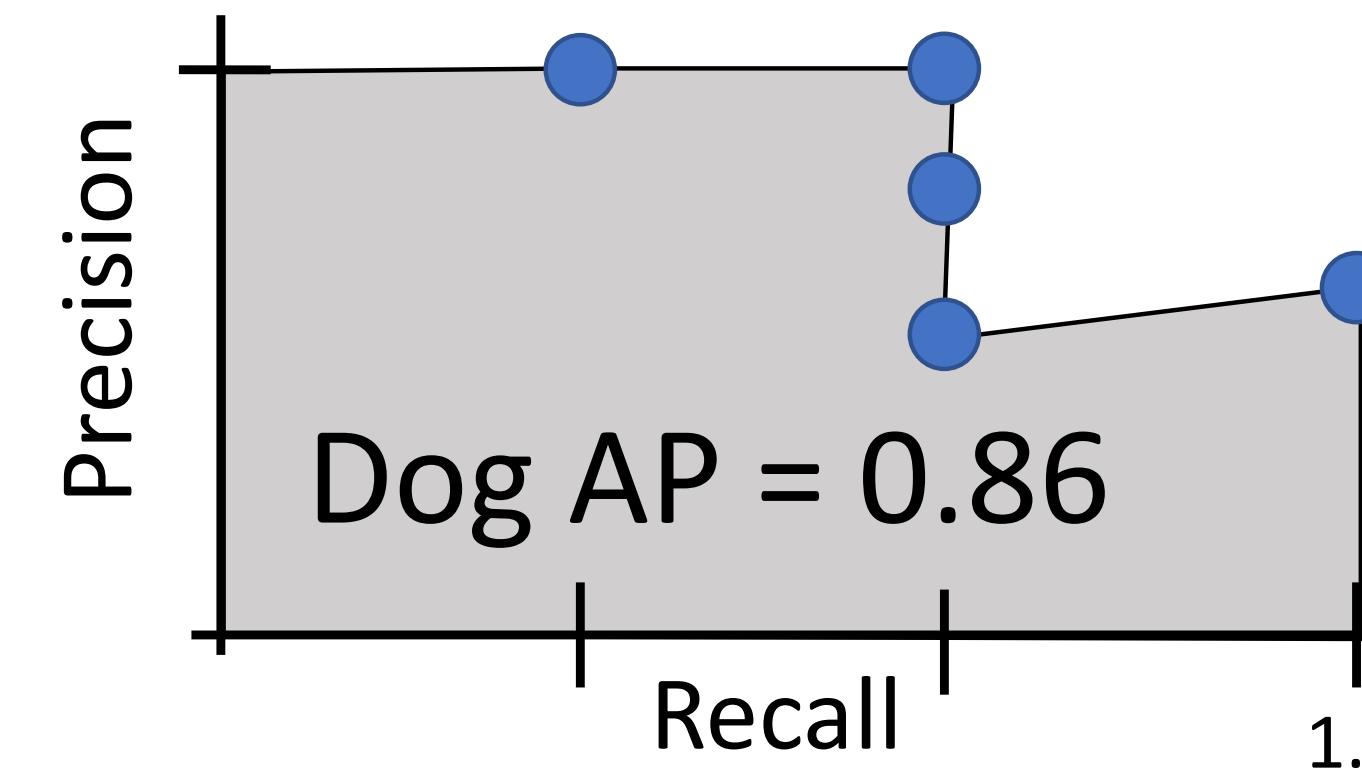
Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with $\text{IoU} > 0.5$, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve
 2. Average Precision (AP) = area under PR curve

How to get AP = 1.0: Hit all GT boxes with $\text{IoU} > 0.5$, and have no “false positive” detections ranked above any “true positives”



All ground-truth dog boxes



Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve
 2. Average Precision (AP) = area under PR curve
3. Mean Average Precision (mAP) = average of AP for each category

Car AP = 0.65

Cat AP = 0.80

Dog AP = 0.86

mAP@0.5 = 0.77

Assessment of Object Detection: mAP

1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
 1. For each detection (highest score to lowest score)
 1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
 2. Otherwise mark it as negative
 3. Plot a point on PR Curve
 2. Average Precision (AP) = area under PR curve
3. Mean Average Precision (mAP) = average of AP for each category
4. For “COCO mAP”: Compute mAP@thresh for each IoU threshold (0.5, 0.55, 0.6, ..., 0.95) and take average

mAP@0.5 = 0.77

mAP@0.55 = 0.71

mAP@0.60 = 0.65

...

mAP@0.95 = 0.2

COCO mAP = 0.4