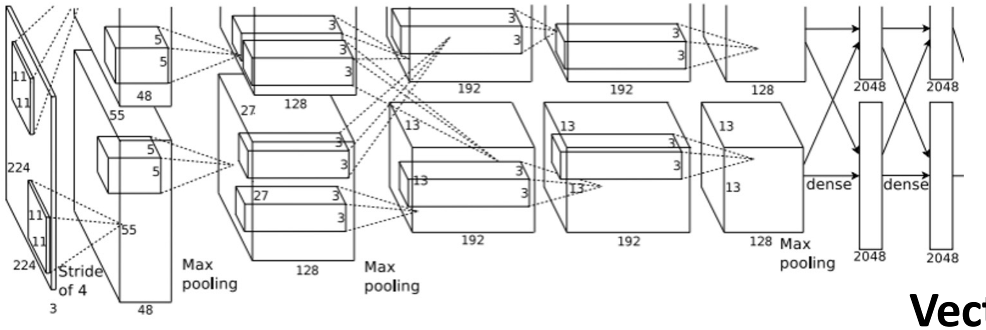
Lecture 12: Where is my cat?

Instructor: Roni Sengupta

ULA: Andrea Dunn, William Li, Liujie Zheng

Course Website: Scan Me!

So far: Image Classification



Vector:
4096

Fully-Connected:
4096 to 1000

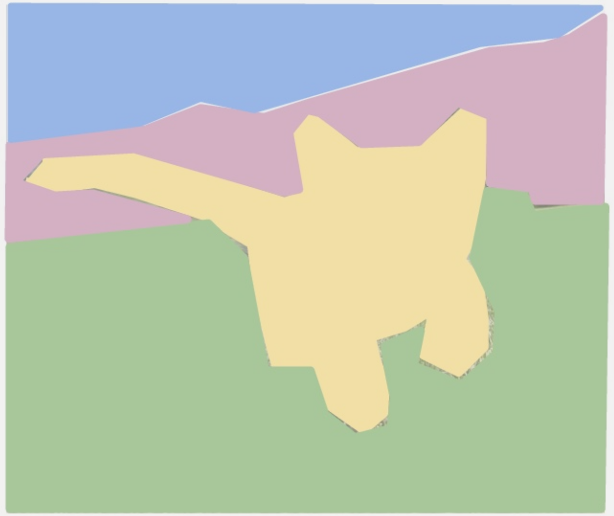
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Classification

CAT

No spatial extent

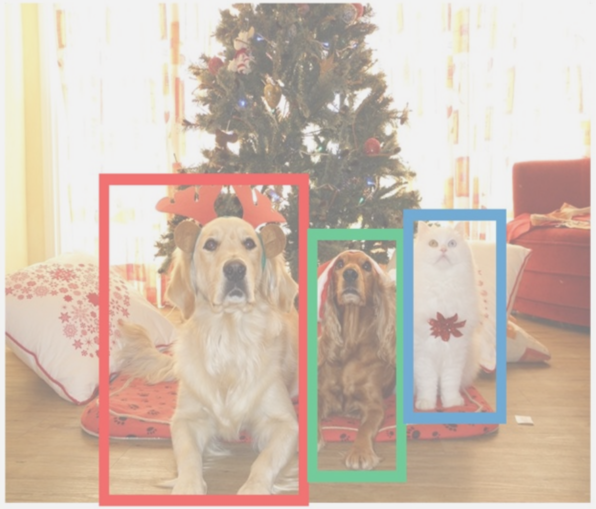
Semantic Segmentation



GRASS, CAT, TREE, SKY

No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation

DOG, DOG, CAT

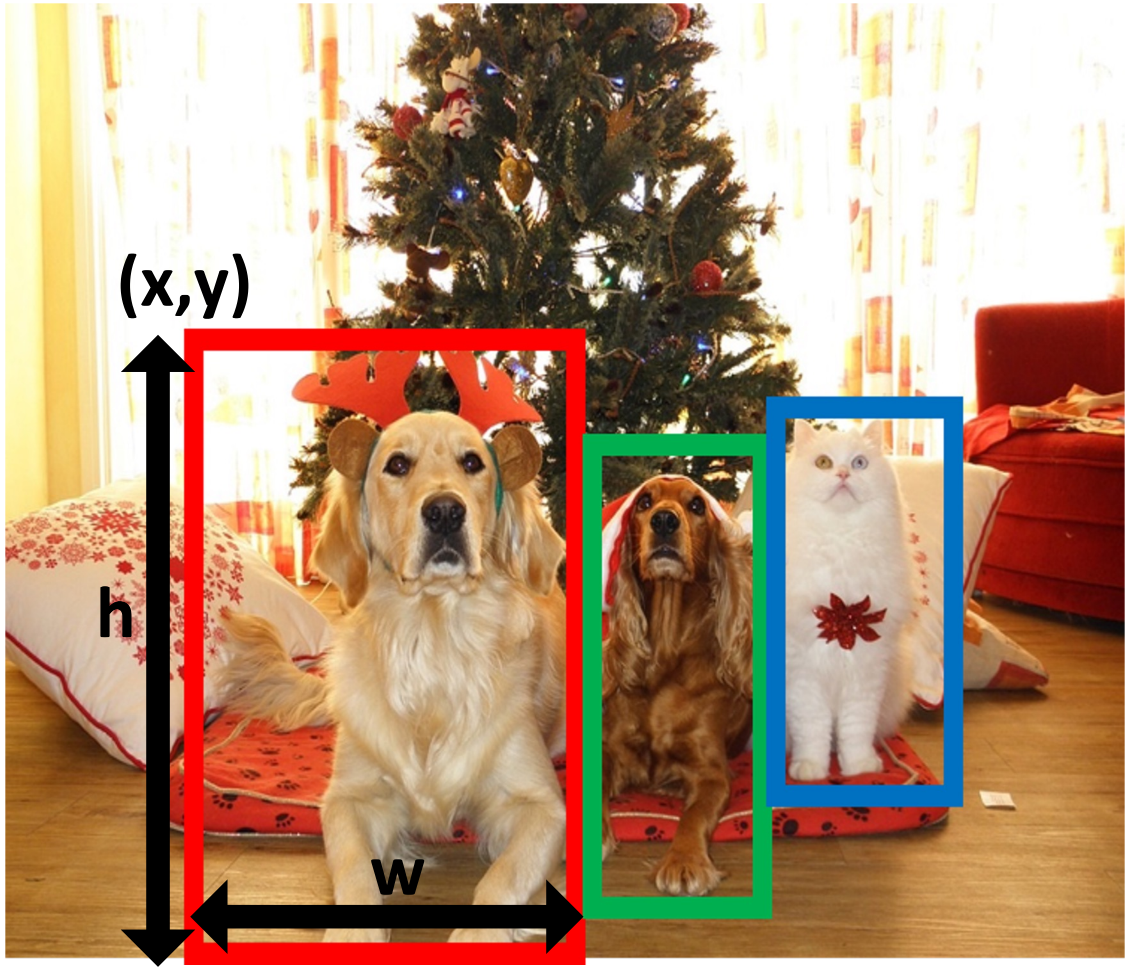
Multiple Objects

Object Detection: Task Definition

Input: Single RGB Image

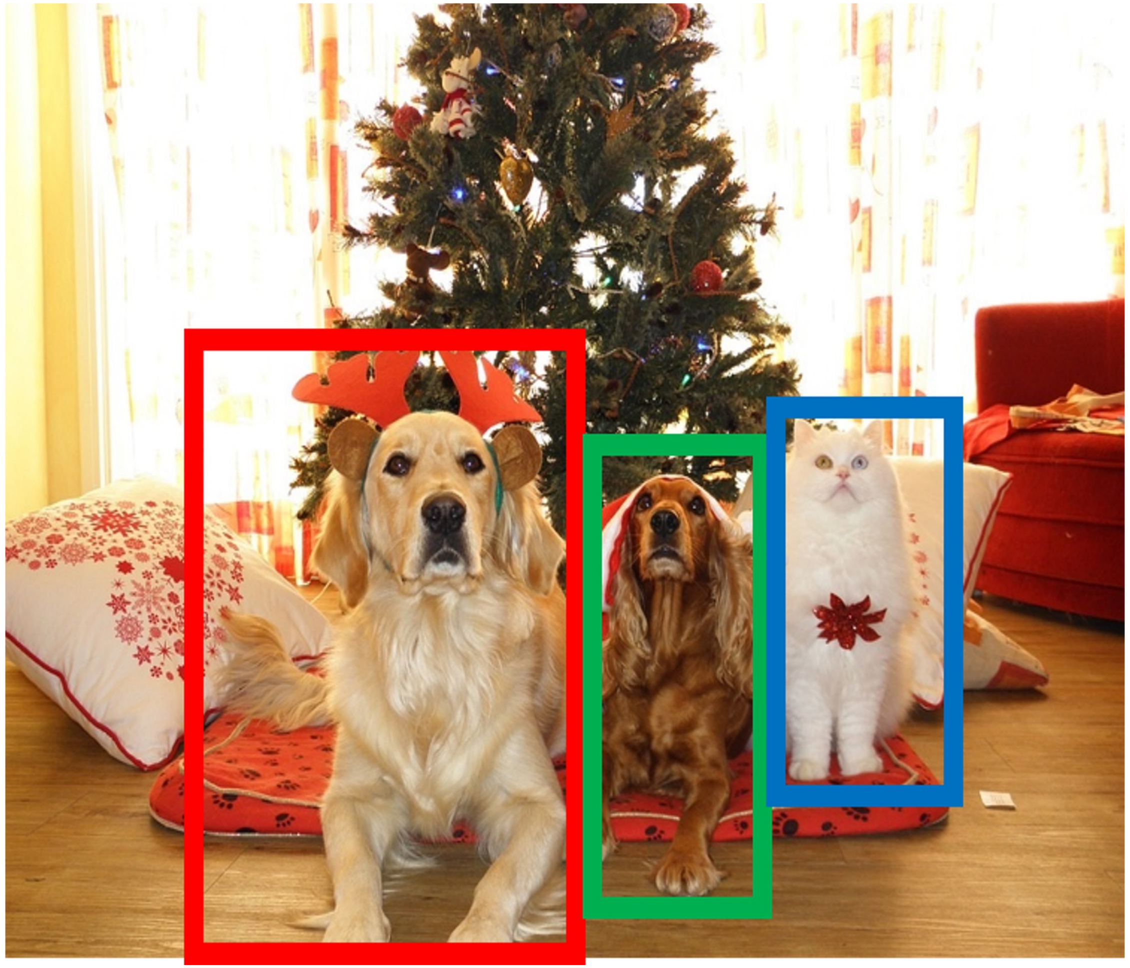
Output: A set of detected objects;
For each object predict:

1. Category label (from fixed, known set of categories)
2. Bounding box (four numbers: x, y, width, height)



Object Detection: Challenges

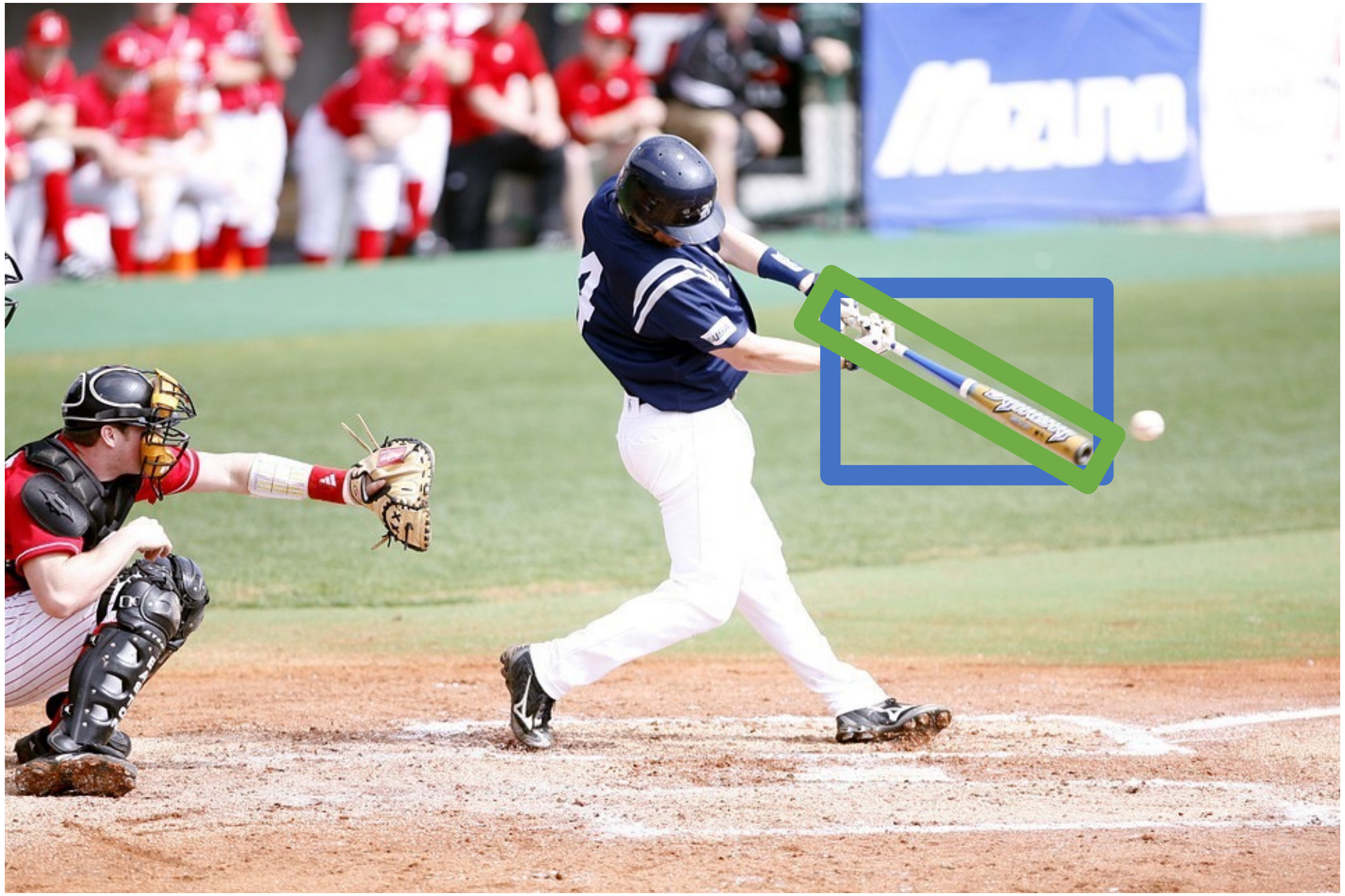
- **Multiple outputs**: Need to output variable numbers of objects per image
- **Multiple types of output**: Need to predict ”what” (category label) as well as “where” (bounding box)
- **Large images**: Classification works at 224x224; need higher resolution for detection, often ~800x600



Bounding Boxes

Bounding boxes are typically *axis-aligned*

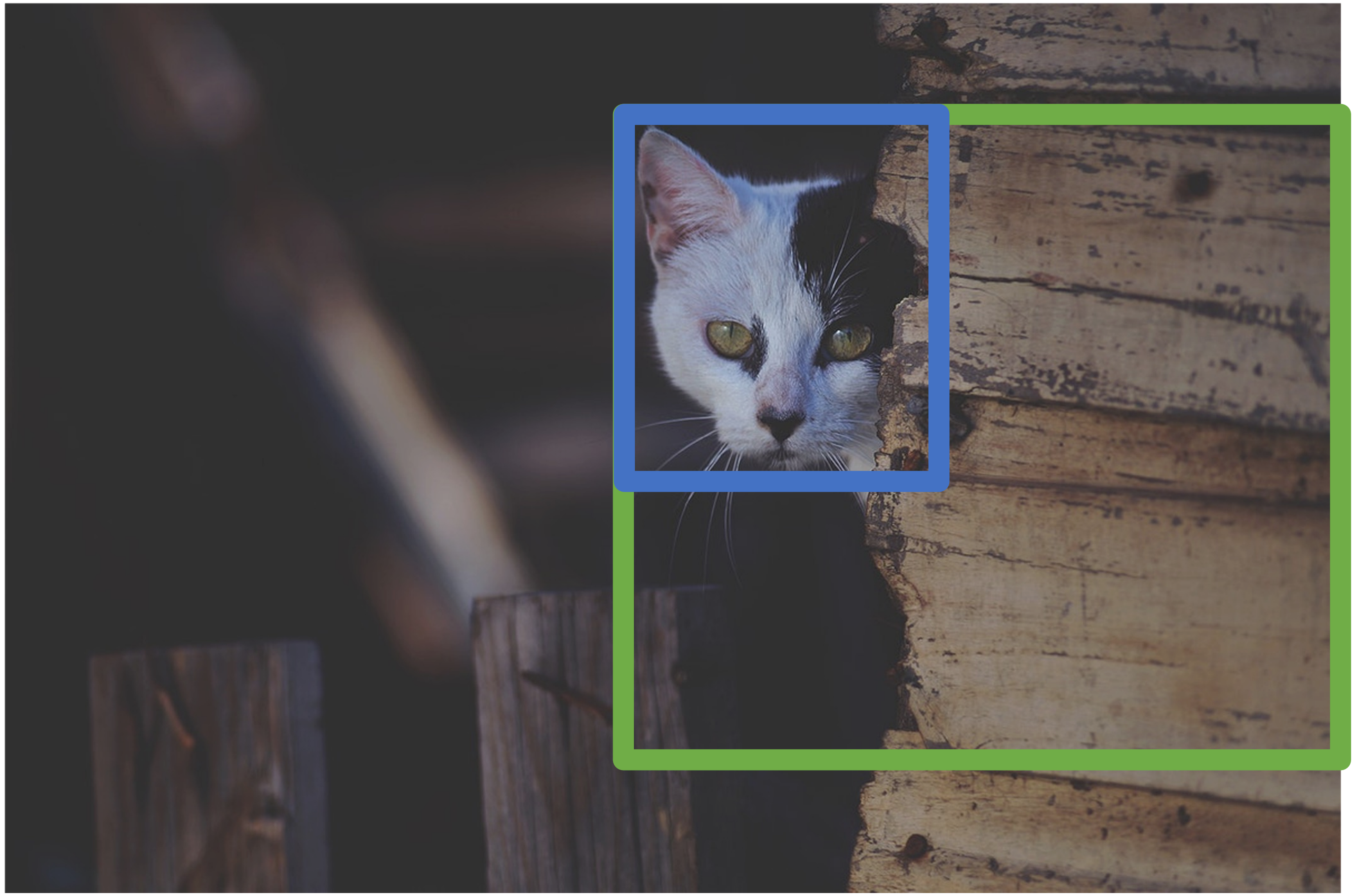
Oriented boxes are much less common



Object Detection: Modal vs Amodal Boxes

"Modal" detection:
Bounding boxes (usually) cover only the visible portion of the object

Amodal detection:
box covers the entire extent of the object, even occluded parts



Zhu et al, "Semantic Amodal Segmentation", CVPR 2017

Today’s class

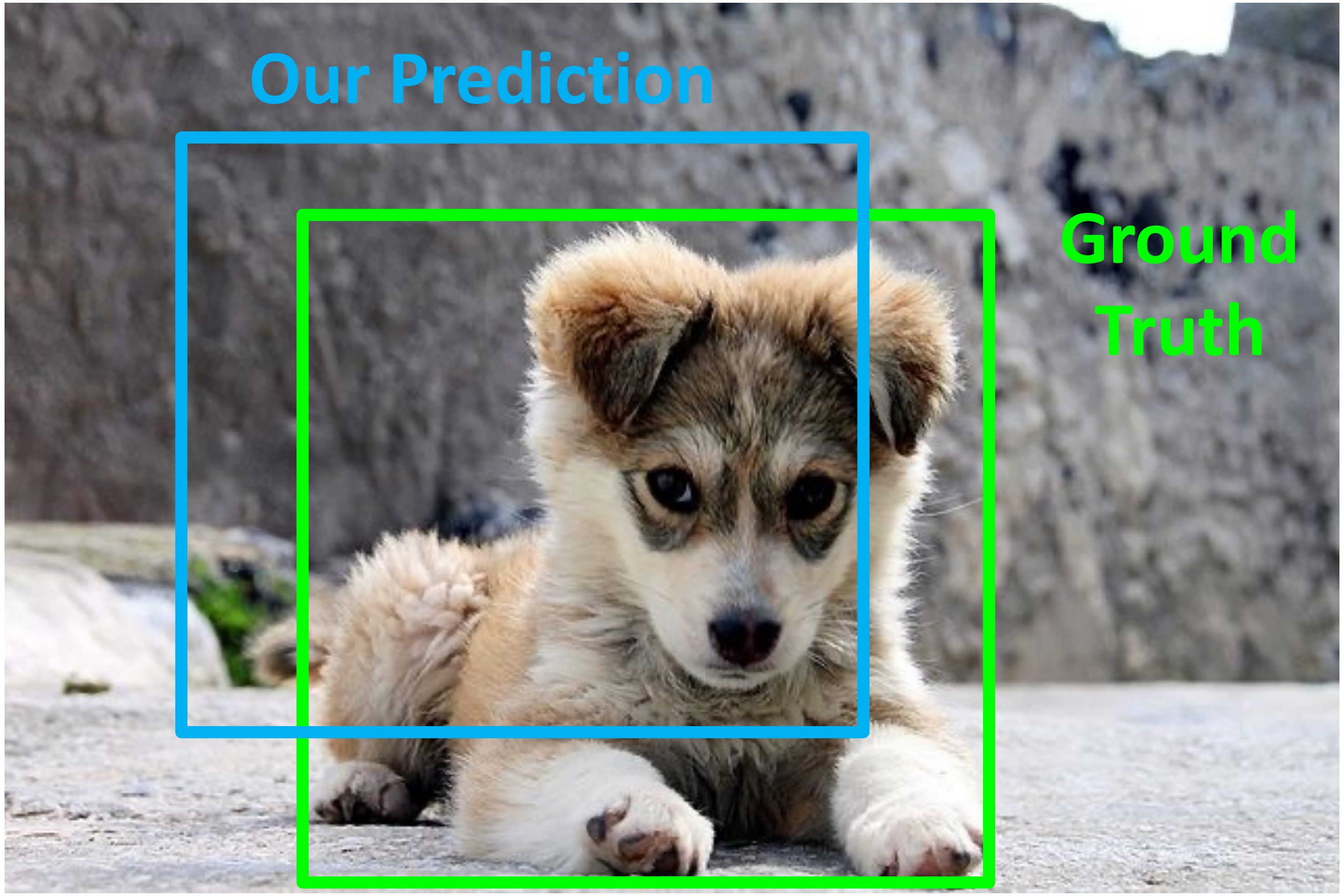
- How do we measure Object Detection accuracy?
- Naïve approaches & R-CNN
- Fast R-CNN
- Region Proposal Network & Faster R-CNN
- Advanced topics:
  - Feature Pyramid Networks to detect at scales
  - Single Shot detection

Today’s class

- How do we measure Object Detection accuracy?
- Naïve approaches & R-CNN
- Fast R-CNN
- Region Proposal Network & Faster R-CNN
- Advanced topics:
  - Feature Pyramid Networks to detect at scales
  - Single Shot detection

Comparing Boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

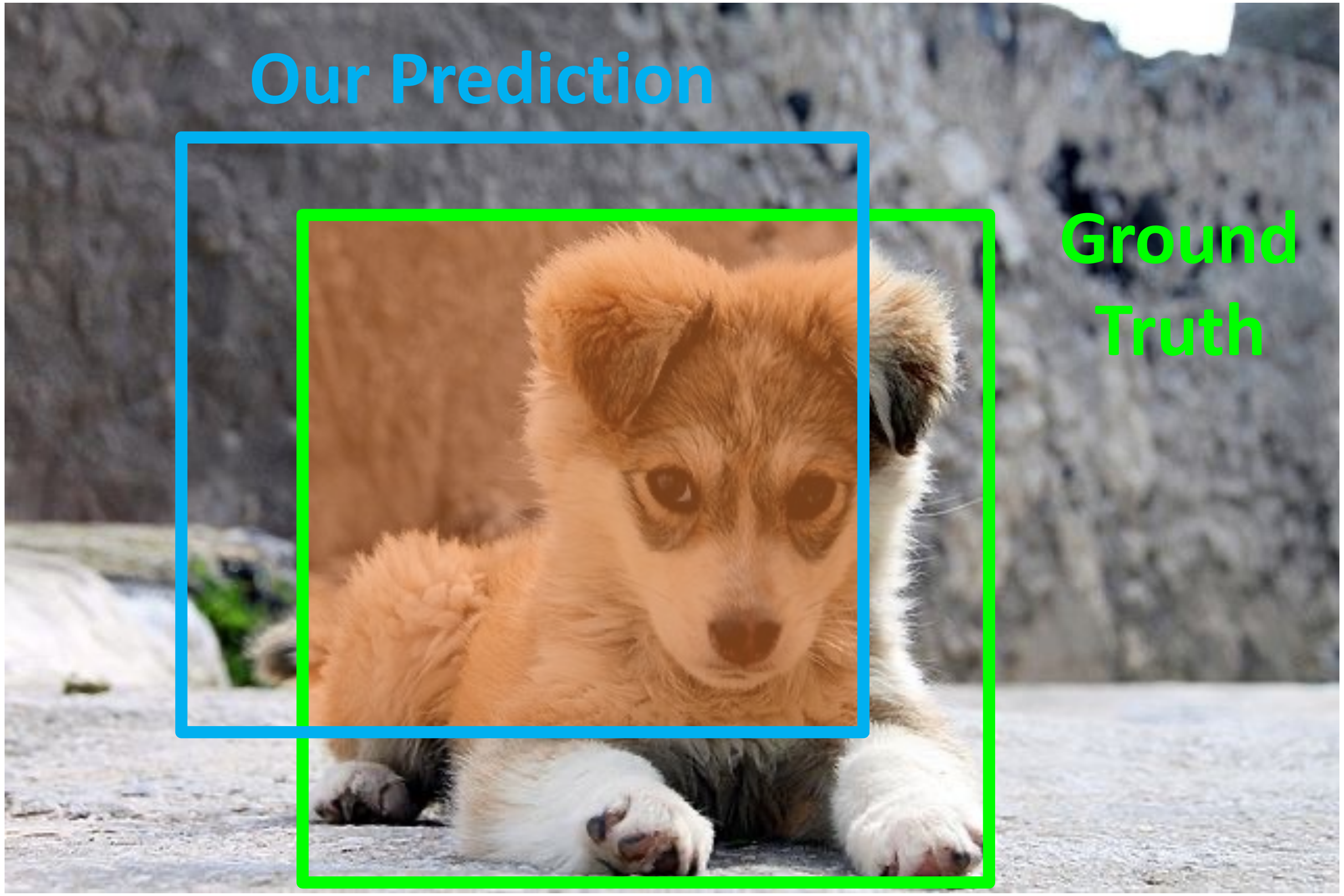


Comparing Boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU)
(Also called “Jaccard similarity” or “Jaccard index”):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

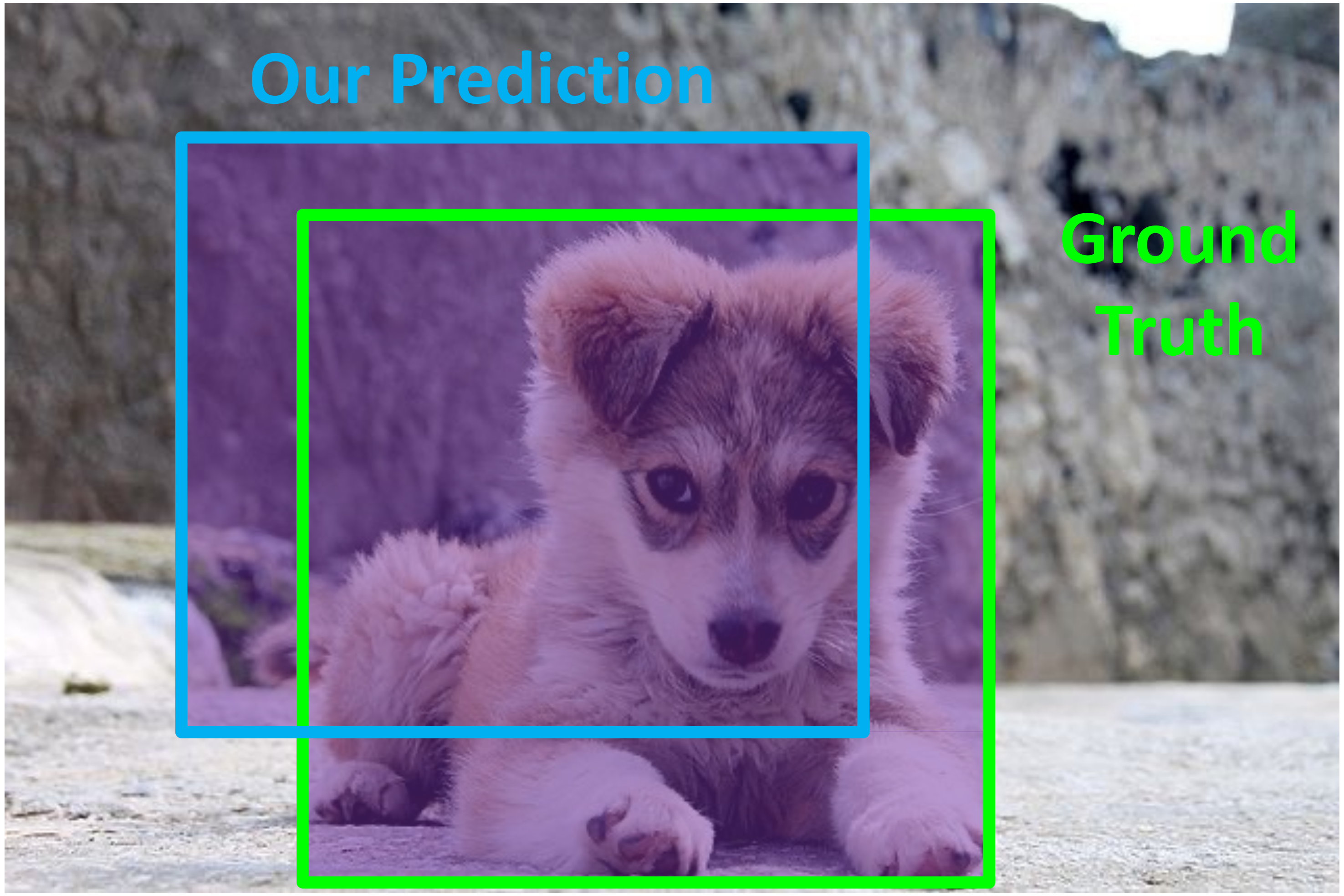


Comparing Boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU)
(Also called "Jaccard similarity" or "Jaccard index"):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$



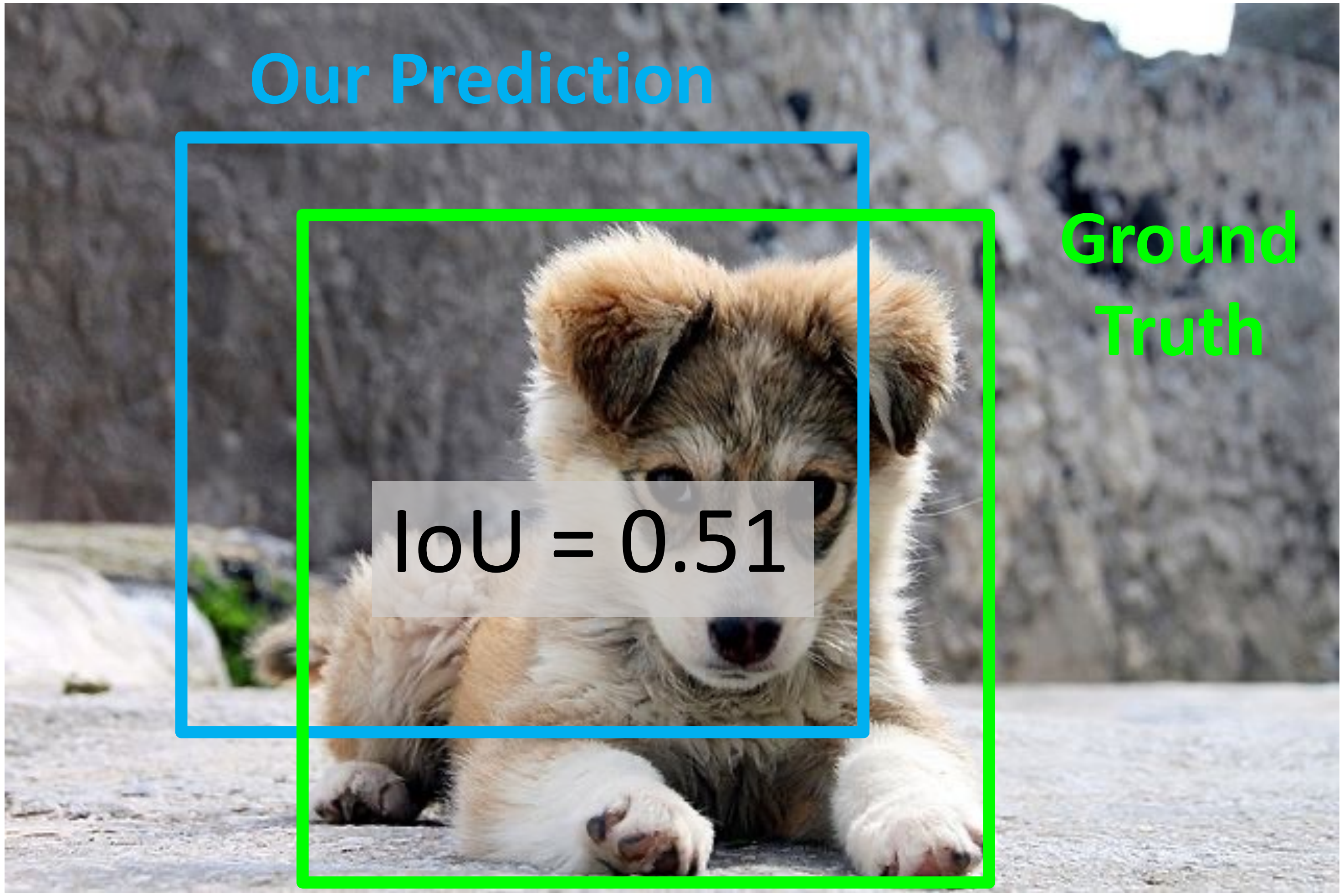
Comparing Boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU)
(Also called “Jaccard similarity” or “Jaccard index”):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

IoU > 0.5 is “decent”



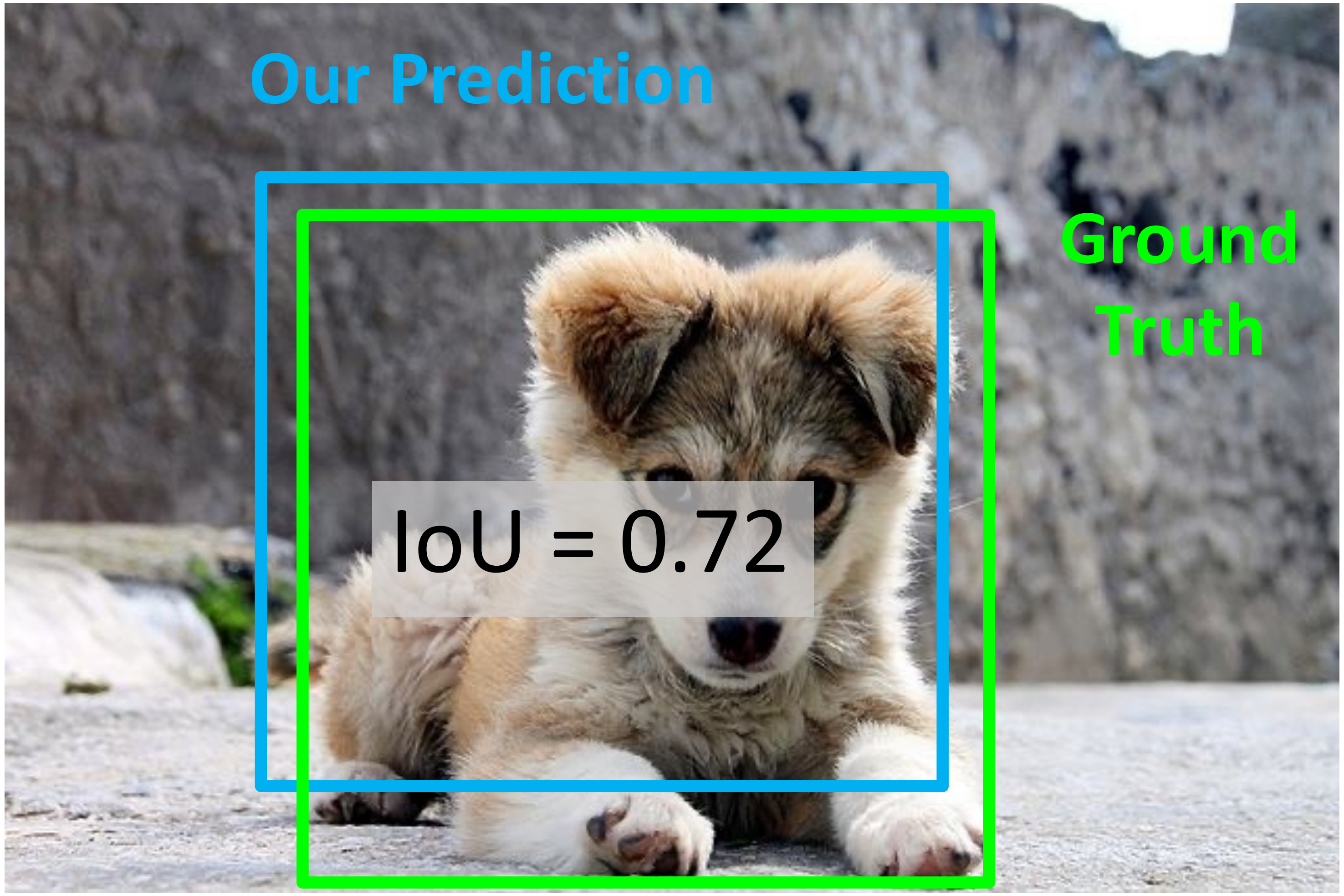
Comparing Boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU)
(Also called "Jaccard similarity" or "Jaccard index"):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

IoU > 0.5 is "decent",
IoU > 0.7 is "pretty good",



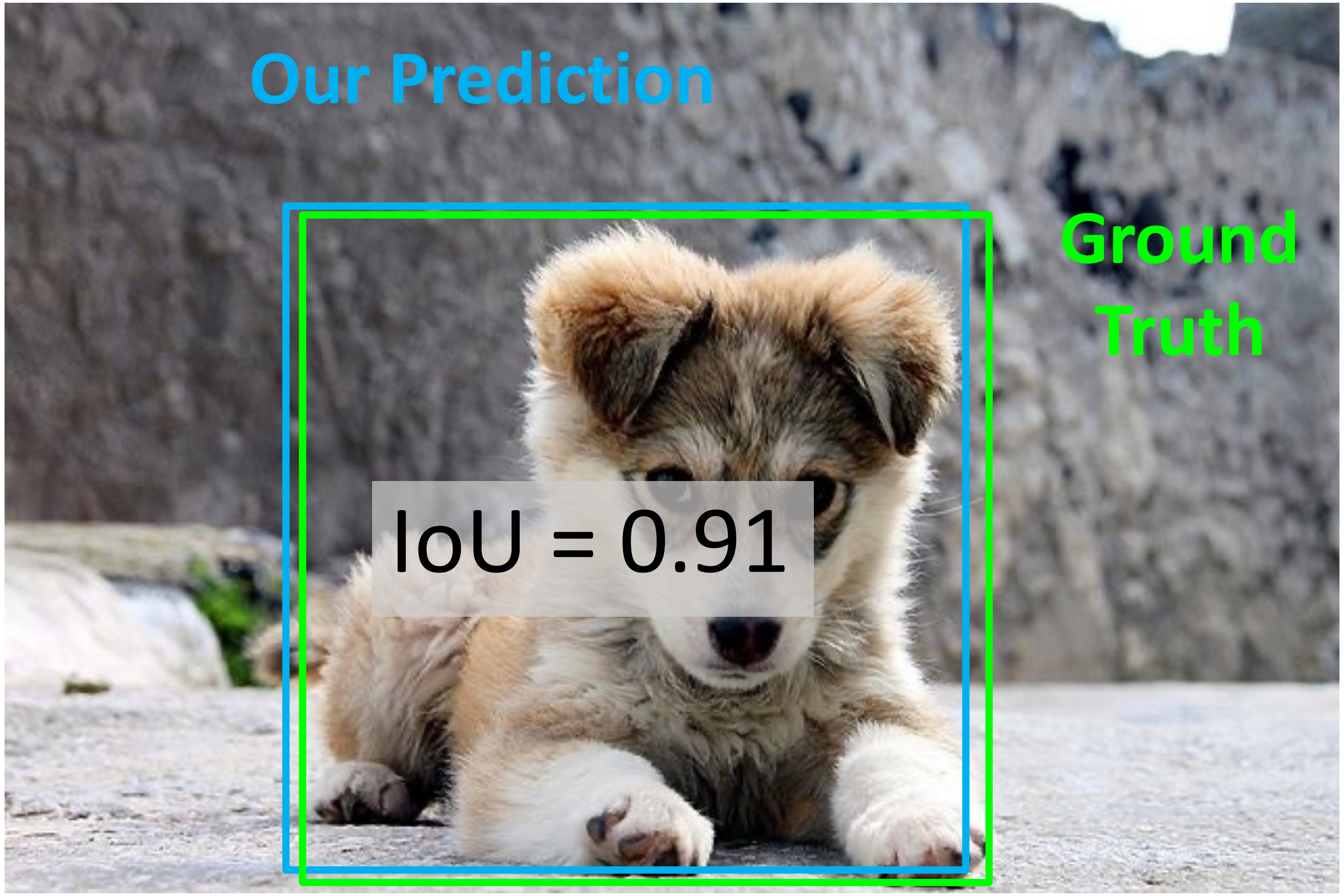
Comparing Boxes: Intersection over Union (IoU)

How can we compare our prediction to the ground-truth box?

Intersection over Union (IoU)
(Also called “Jaccard similarity” or “Jaccard index”):

$$\frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

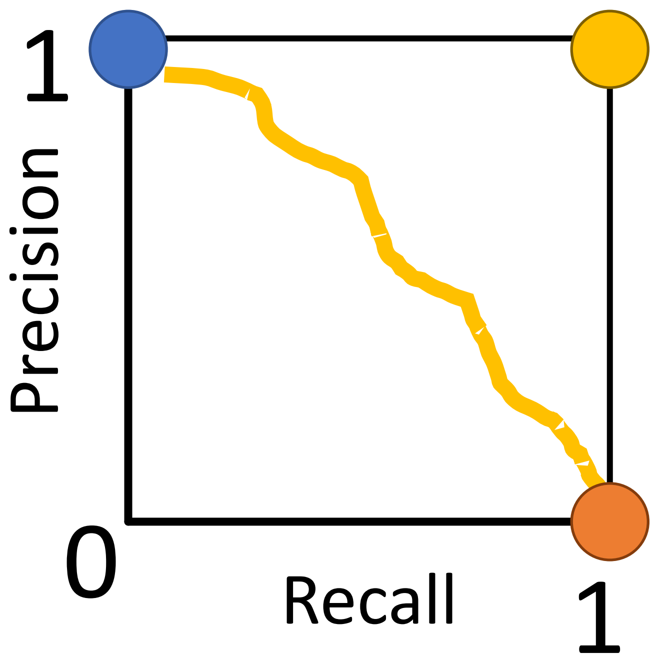
IoU > 0.5 is “decent”,
IoU > 0.7 is “pretty good”,
IoU > 0.9 is “almost perfect”



Precision & Recall

- True detection: high intersection over union based on a threshold
- Precision: #true detections / #detections
- Recall: #true detections / #true positives

Reject everything: no mistakes



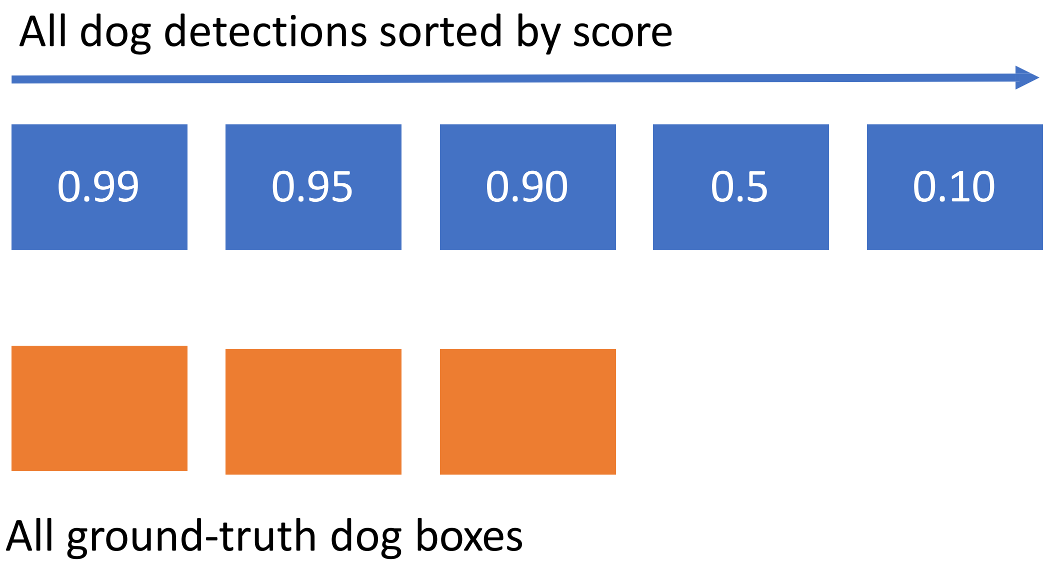
Ideal!

Summarize by area under curve (avg. precision)

Accept everything: Miss nothing

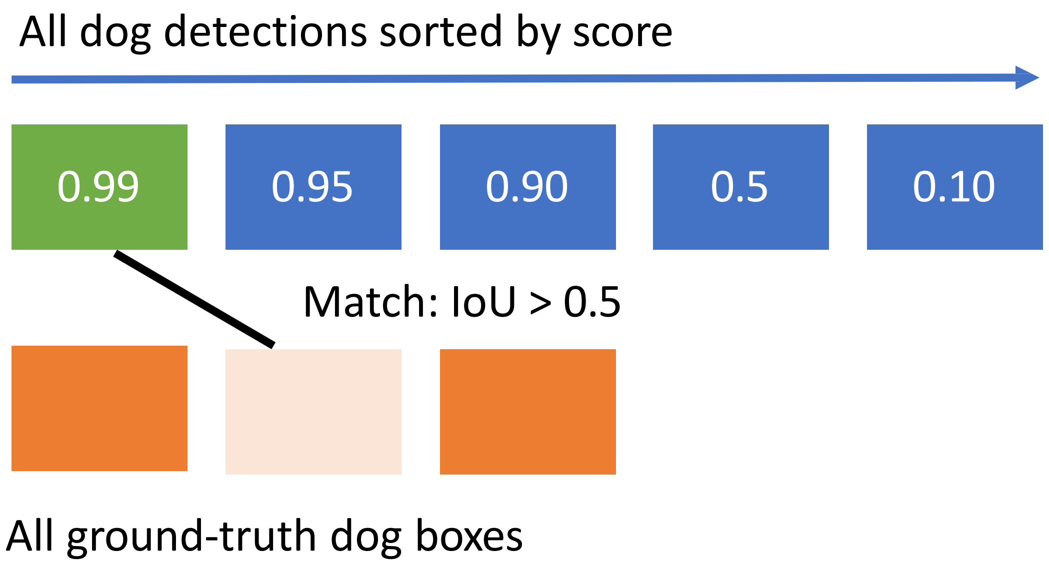
Evaluating Object Detectors: Mean Average Precision (mAP)

1. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve



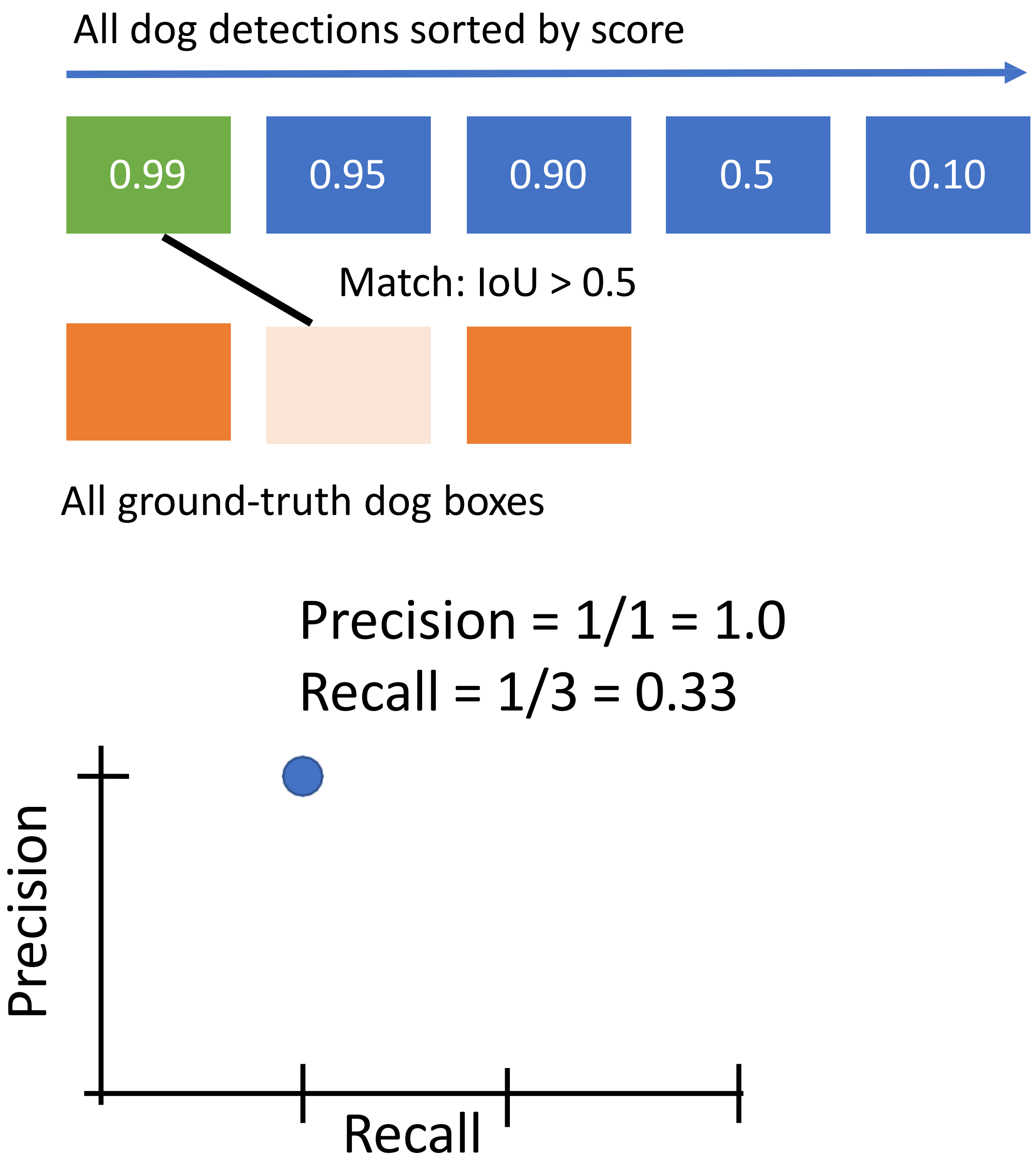
Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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



Evaluating Object Detectors: Mean Average Precision (mAP)

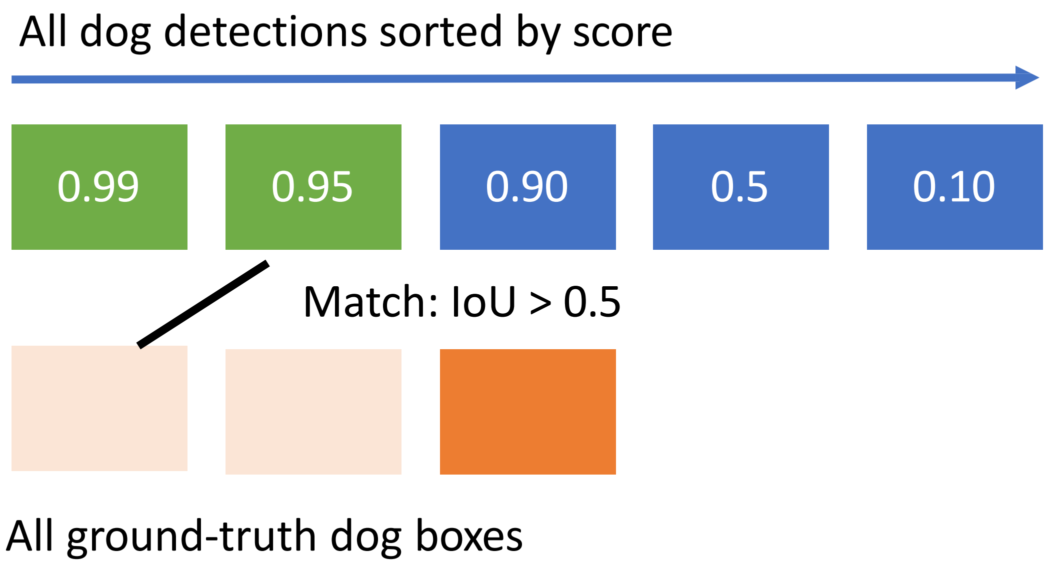
1. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
   1. For each detection (highest score to lowest score)
      1. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      2. Otherwise mark it as negative
      3. Plot a point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



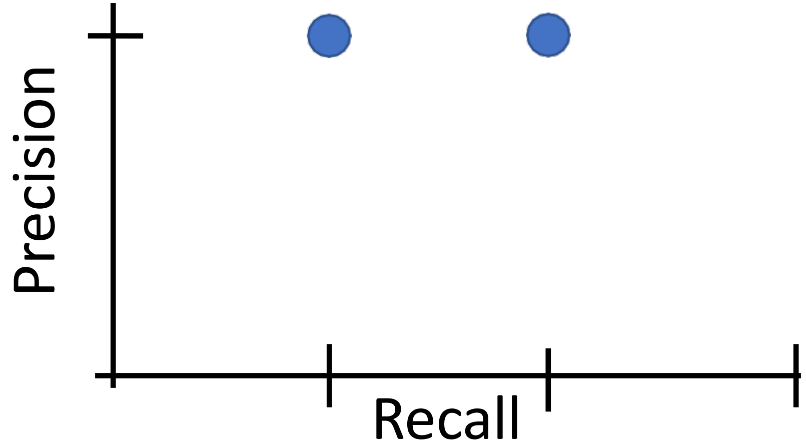
Precision: #true detections / #detections
Recall: #true detections / #true positives

Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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



Precision = 2/2 = 1.0
Recall = 2/3 = 0.67

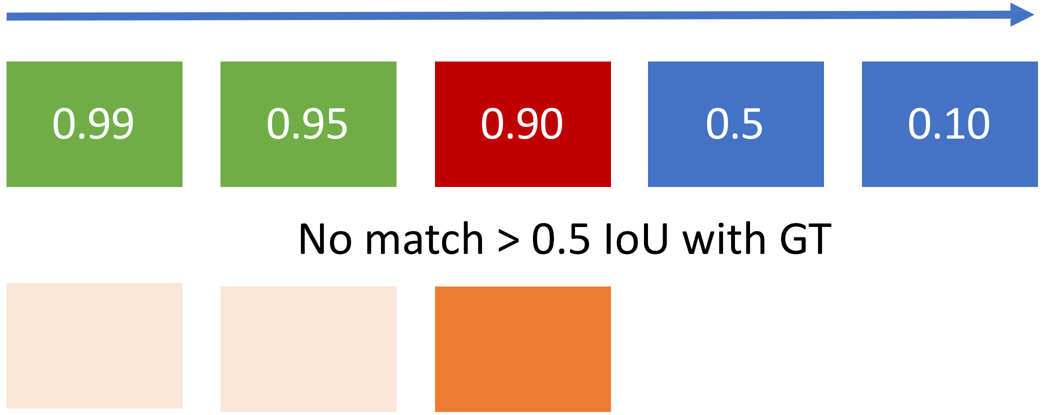


Precision: #true detections / #detections
Recall: #true detections / #true positives

Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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

All dog detections sorted by score

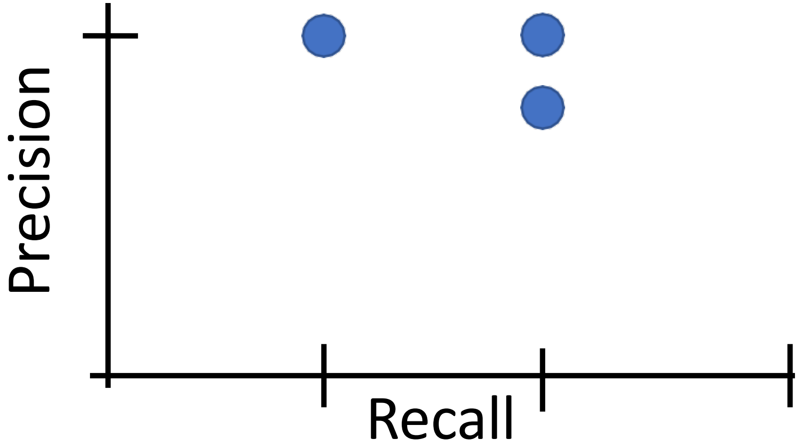


No match > 0.5 IoU with GT

All ground-truth dog boxes

Precision = 2/3 = 0.67

Recall = 2/3 = 0.67

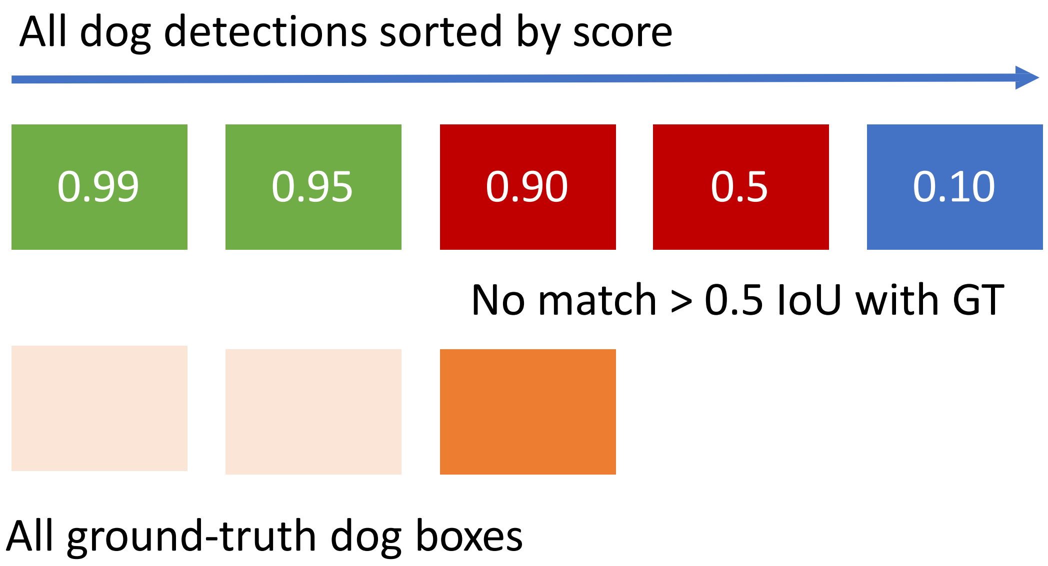


Precision: #true detections / #detections

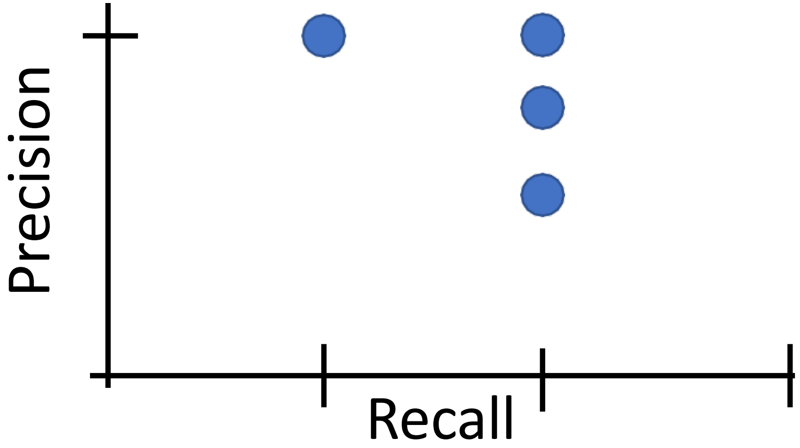
Recall: #true detections / #true positives

Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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



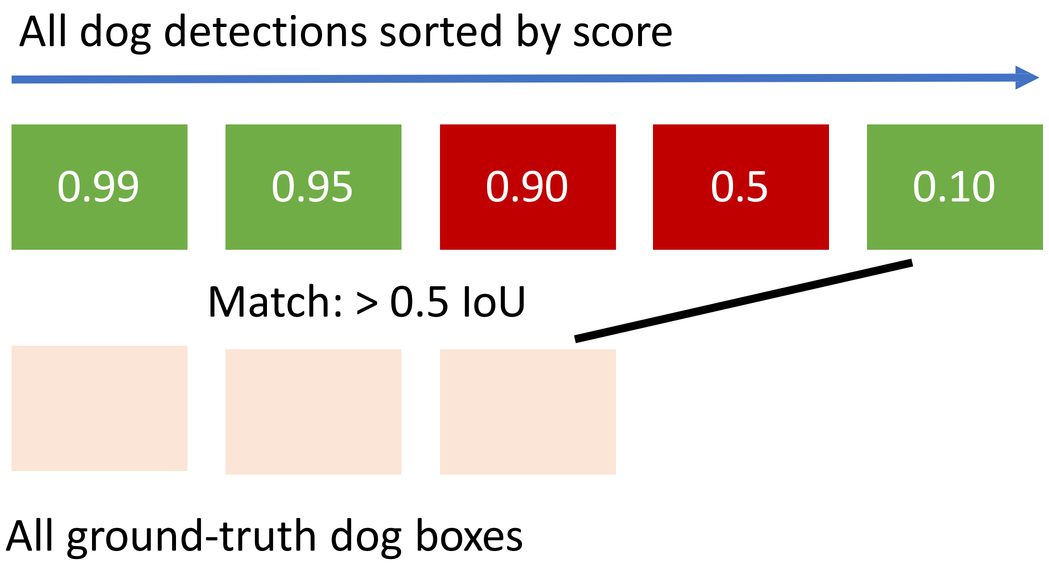
Precision = 2/4 = 0.5
Recall = 2/3 = 0.67



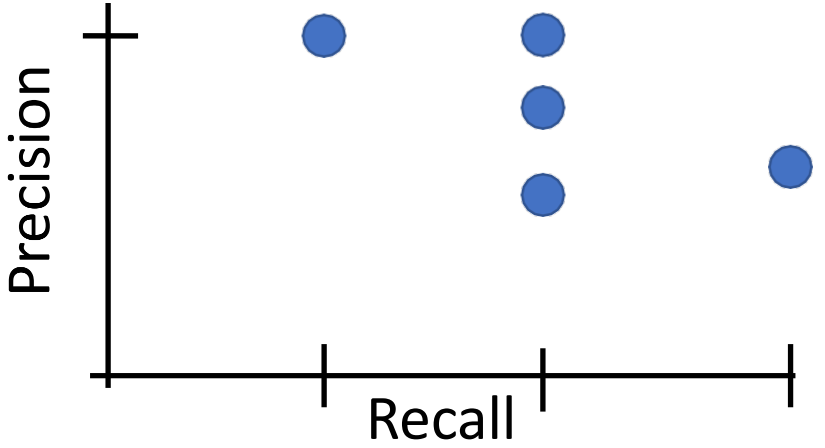
Precision: #true detections / #detections
Recall: #true detections / #true positives

Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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



Precision = 3/5 = 0.6
Recall = 3/3 = 1.0

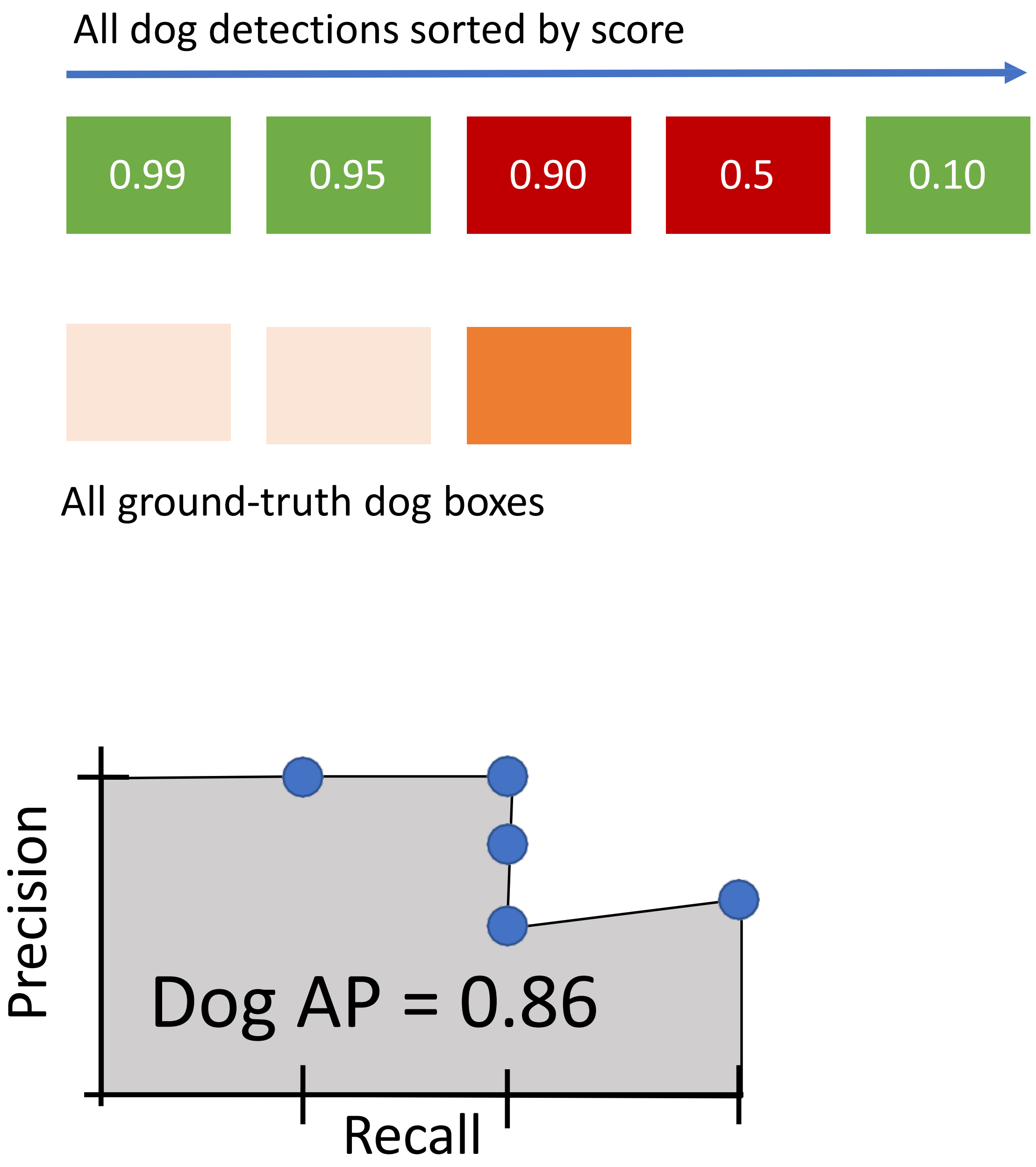


Precision: #true detections / #detections
Recall: #true detections / #true positives

Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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

How to get AP = 1.0: Hit all GT boxes with IoU > 0.5, and have no "false positive" detections ranked above any "true positives"



Precision: #true detections / #detections
Recall: #true detections / #true positives

Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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
2. 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

Evaluating Object Detectors: Mean Average Precision (mAP)

1. 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
2. Mean Average Precision (mAP) = average of AP for each category
3. 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

…

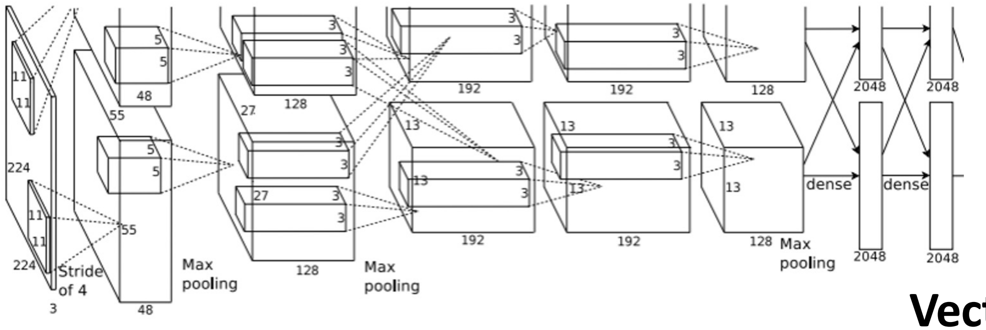
mAP@0.95 = 0.2

COCO mAP = 0.4

Today’s class

- How do we measure Object Detection accuracy?
- Naïve approaches & R-CNN
- Fast R-CNN
- Region Proposal Network & Faster R-CNN
- Advanced topics:
    - Feature Pyramid Networks to detect at scales
    - Single Shot detection

So far: Image Classification



Vector:
4096

Fully-Connected:
4096 to 1000

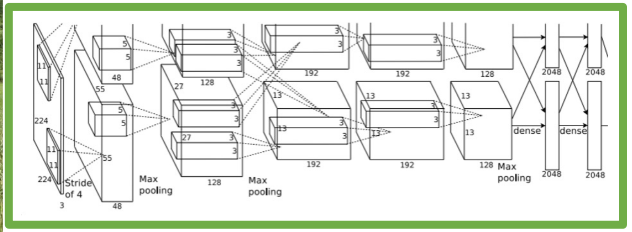
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Detecting a single object

“What”

This image is CC0 public domain

Often pretrained on ImageNet (Transfer learning)



Vector: 4096

Fully Connected: 4096 to 1000

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

Correct label:
Cat

Softmax Loss

Multitask Loss

Weighted Sum

Loss

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

Correct box:
(x’, y’, w’, h’)

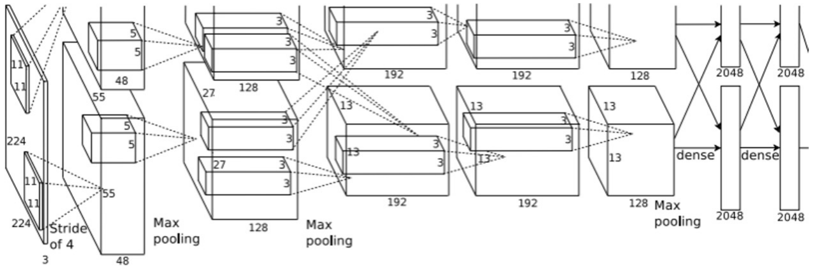
“Where”

Treat localization as a regression problem!

Problem: Images can have more than one object!

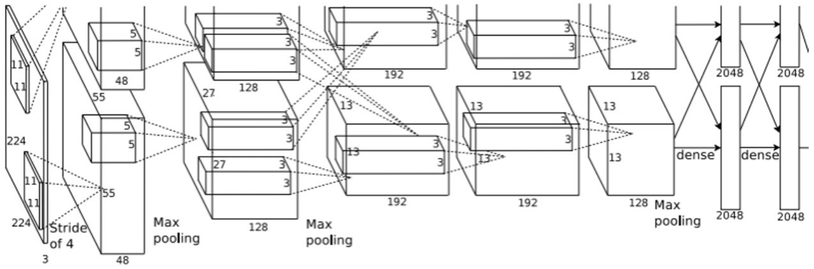
Detecting Multiple Objects

Need different numbers of outputs per image



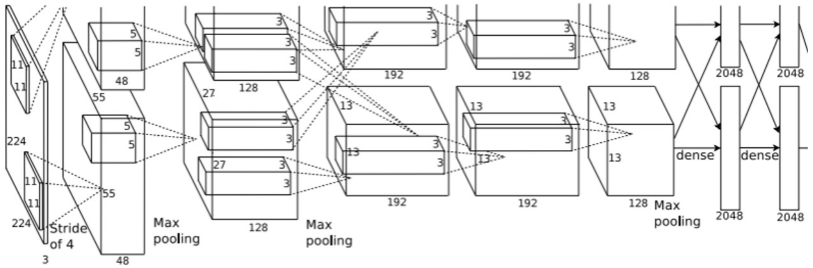
CAT: (x, y, w, h)

4 numbers



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

12 numbers

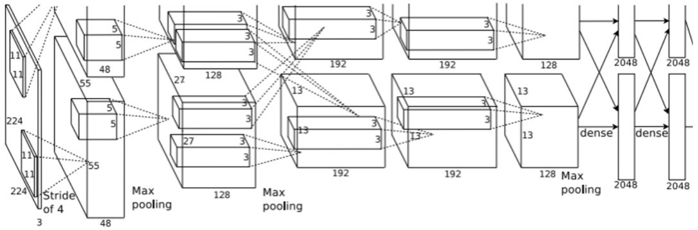


DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

Many numbers!

Multiple Objects: **Sliding Window**

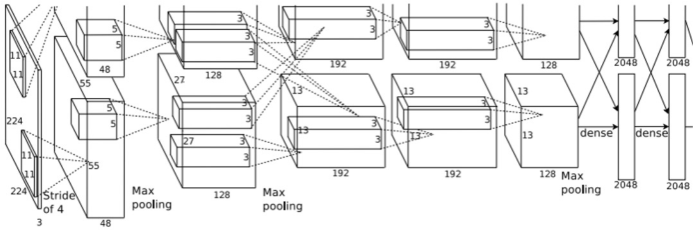
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

Multiple Objects: **Sliding Window**

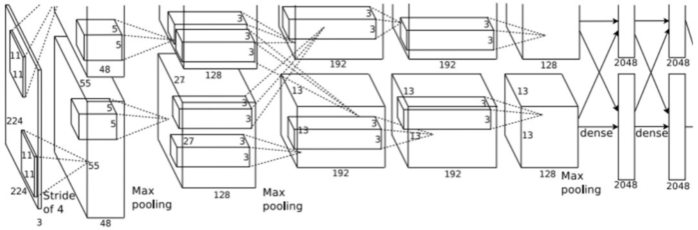
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Multiple Objects: **Sliding Window**

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

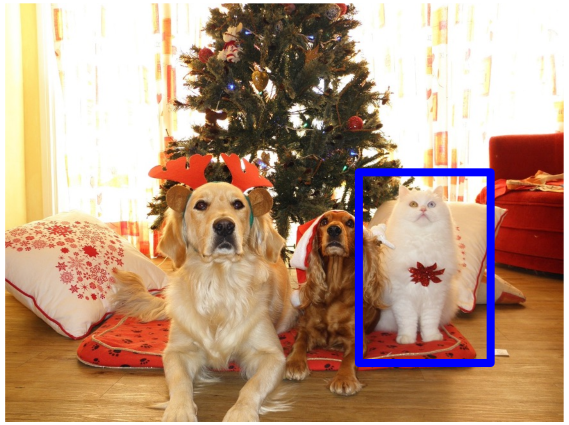


Dog? YES

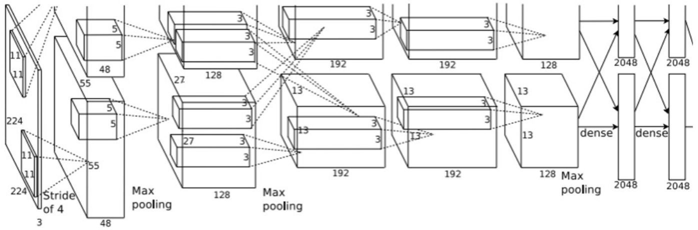
Cat? NO

Background? NO

Multiple Objects: **Sliding Window**



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

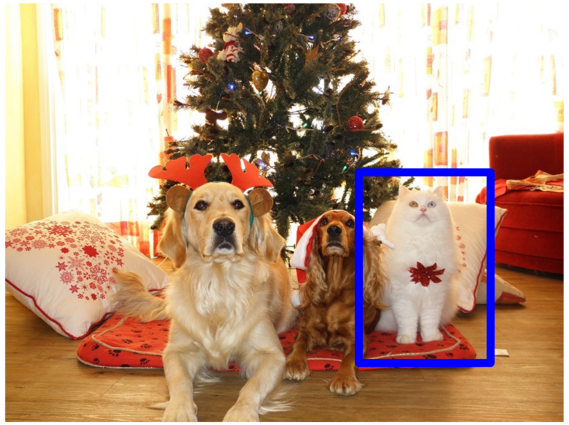


Dog? NO
Cat? YES
Background? NO

Multiple Objects: **Sliding Window**

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

800 x 600 image has ~58M boxes! No way we can evaluate them all



Question: How many possible boxes are there in an image of size H x W?

Consider a box of size h x w:
Possible x positions: W – w + 1
Possible y positions: H – h + 1
Possible positions:
(W – w + 1) * (H – h + 1)

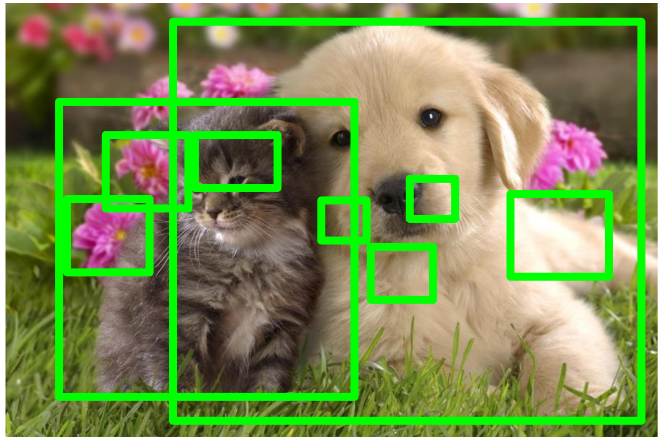
Total possible boxes of different size h x w:

$$\sum_{h=1}^{H}\sum_{w=1}^{W}(W-w+1)(H-h+1)$$

$$=\frac{H(H+1)}{2}\frac{W(W+1)}{2}$$

Region Proposals

- Find a small set of boxes that are likely to cover all objects
- Often based on heuristics: e.g. look for "blob-like" image regions
- Relatively fast to run; e.g. **Selective Search algorithm** 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: Region-Based CNN

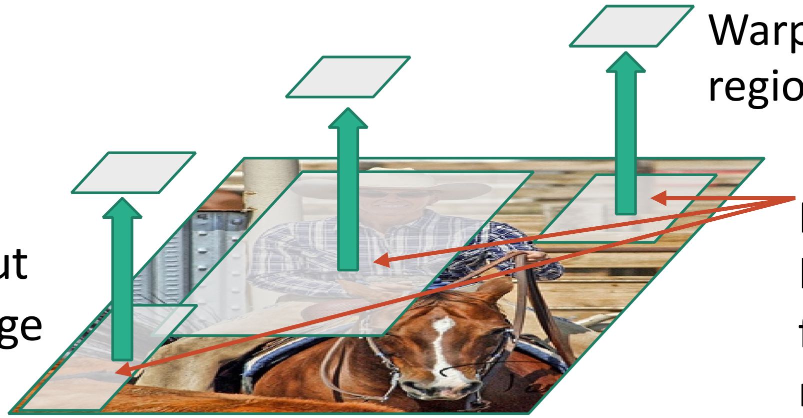
Input image

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: Region-Based CNN

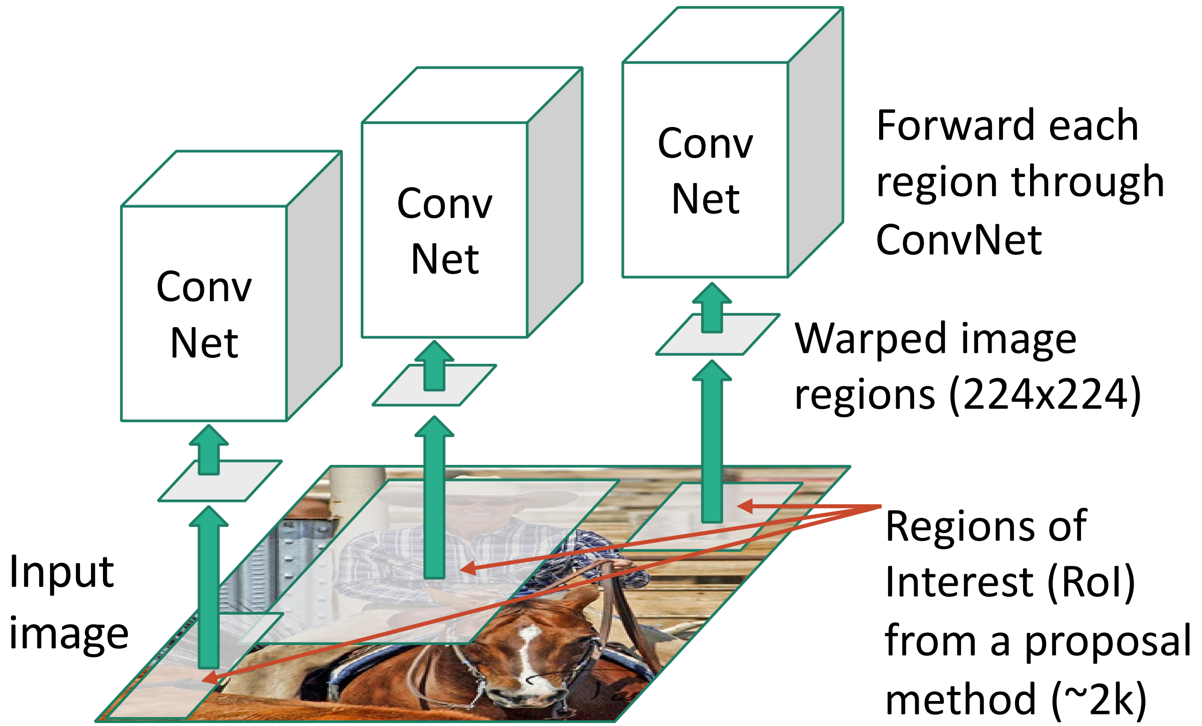
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.

R-CNN: Region-Based CNN



Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

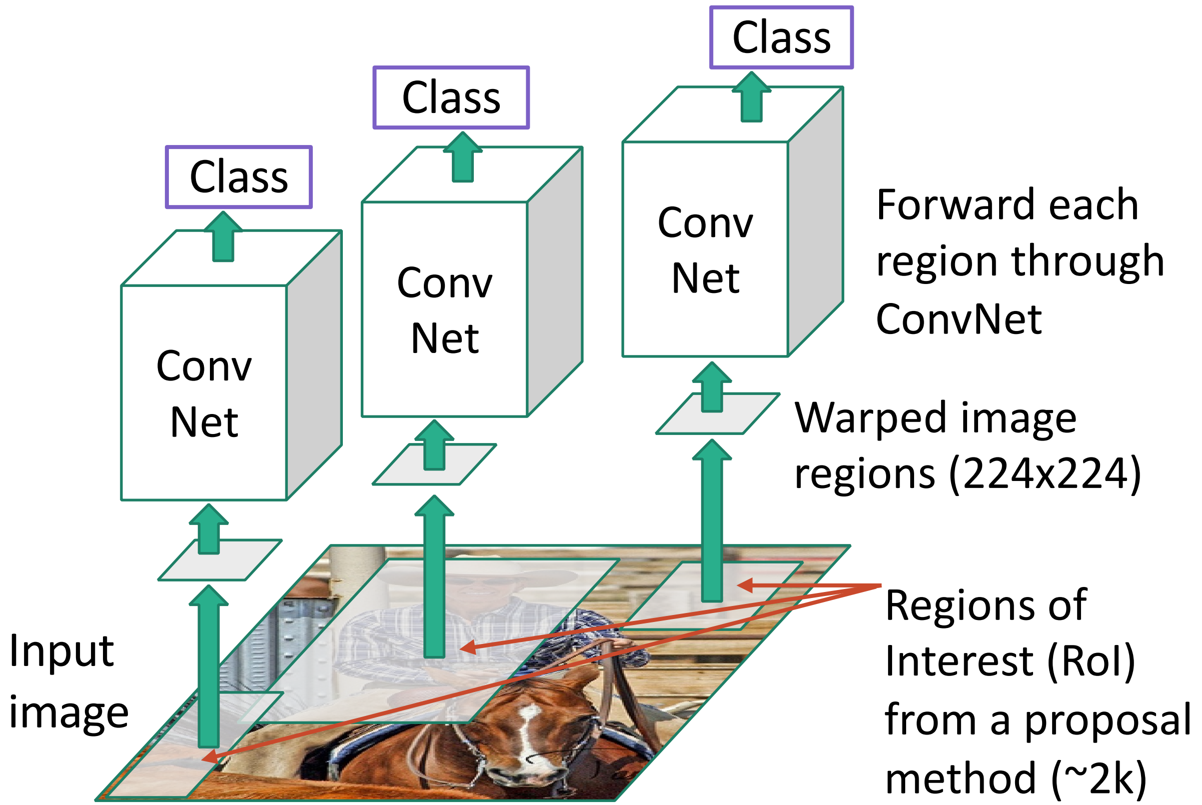
R-CNN: Region-Based CNN



Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

R-CNN: Region-Based CNN

Classify each region

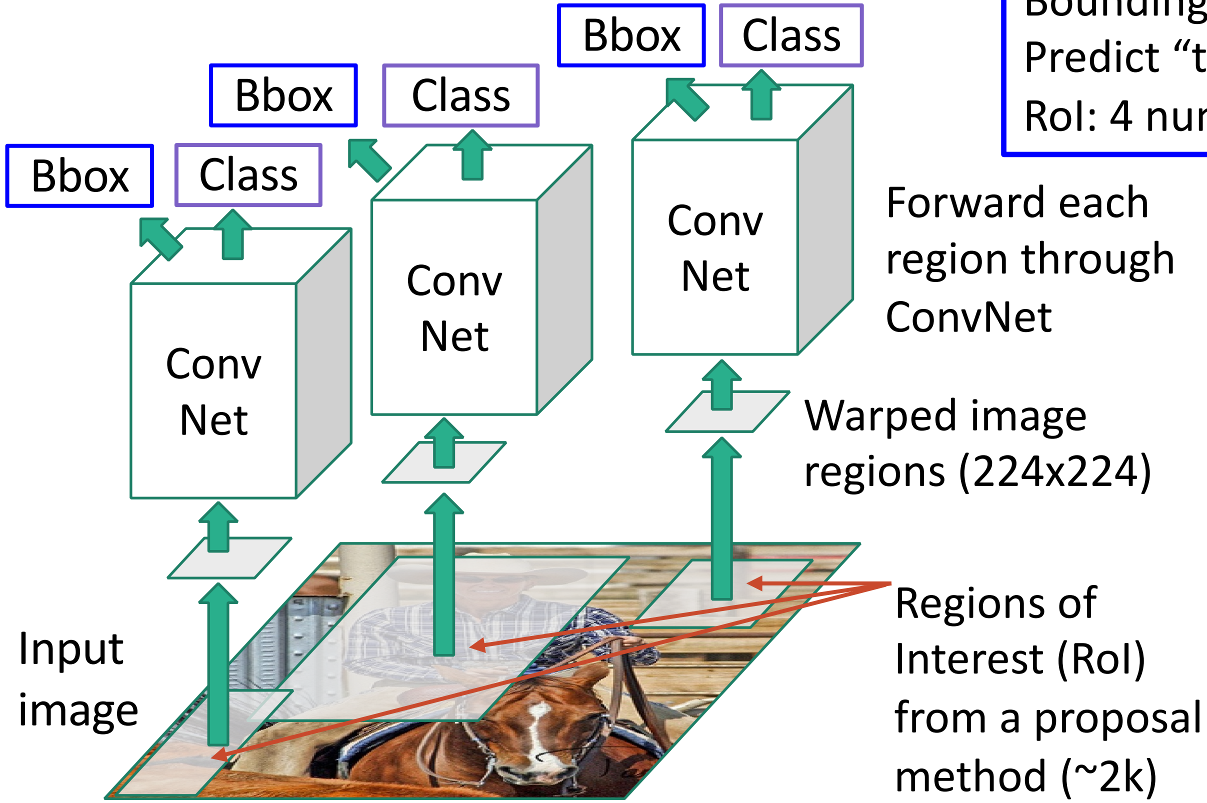


Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

R-CNN: Region-Based CNN

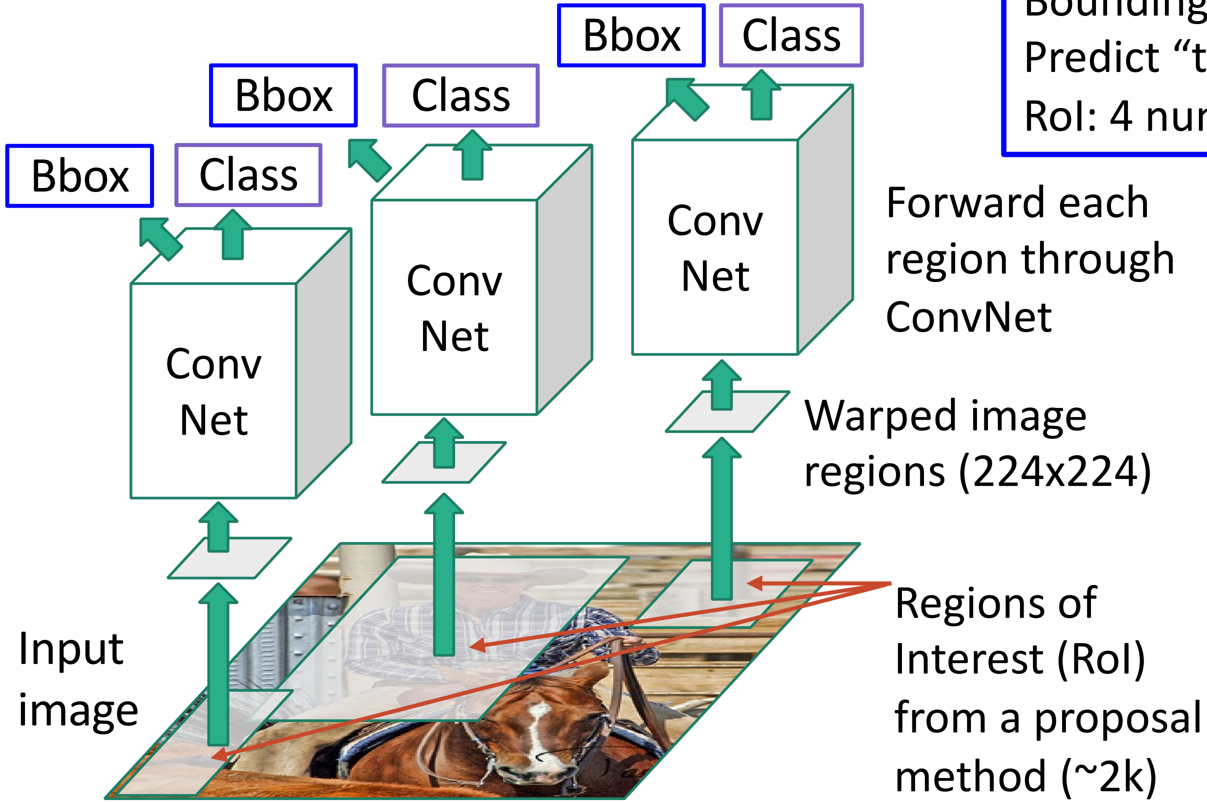
Classify each region

Bounding box regression:
Predict "transform" to correct the RoI: 4 numbers (t_x, t_y, t_h, t_w)



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: Region-Based CNN



Classify each region

Bounding box regression:
Predict "transform" to correct the RoI: 4 numbers (t_x, t_y, t_h, t_w)

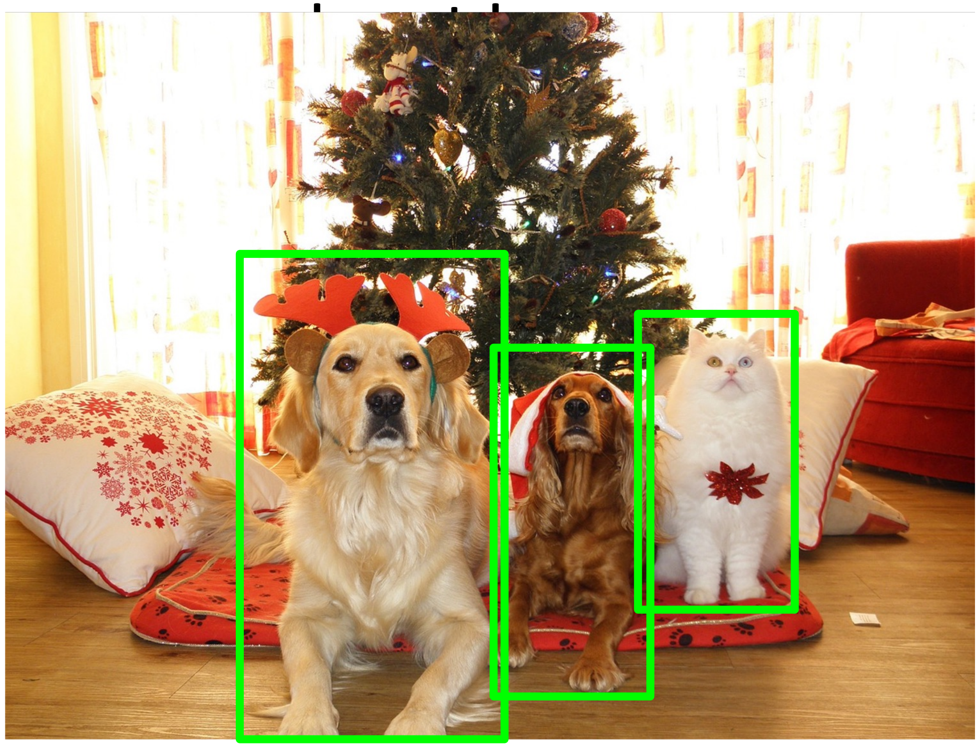
Region proposal: (p_x, p_y, p_h, p_w)
Transform: (t_x, t_y, t_h, t_w)
Output box: (b_x, b_y, b_h, b_w)

Translate relative to box size:
$b_x = p_x + p_w t_x$ $\quad$ $b_y = p_y + p_h t_y$

Log-space scale transform:
$b_w = p_w \exp(t_w)$ $\quad$ $b_h = p_h \exp(t_h)$

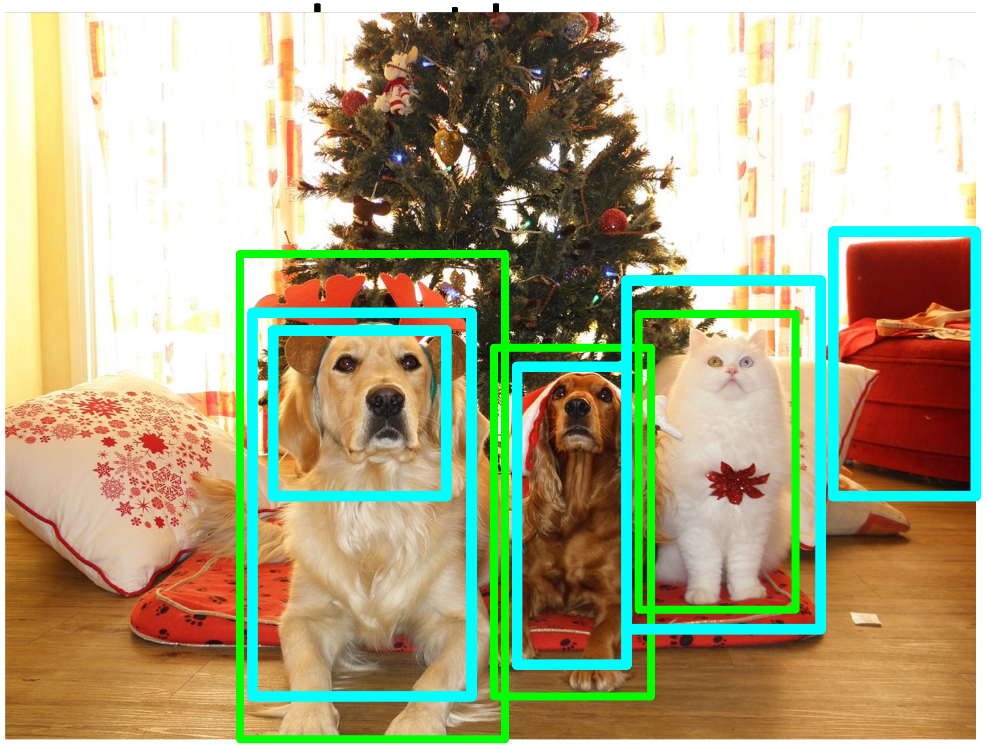
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

R-CNN Training



Ground-Truth boxes

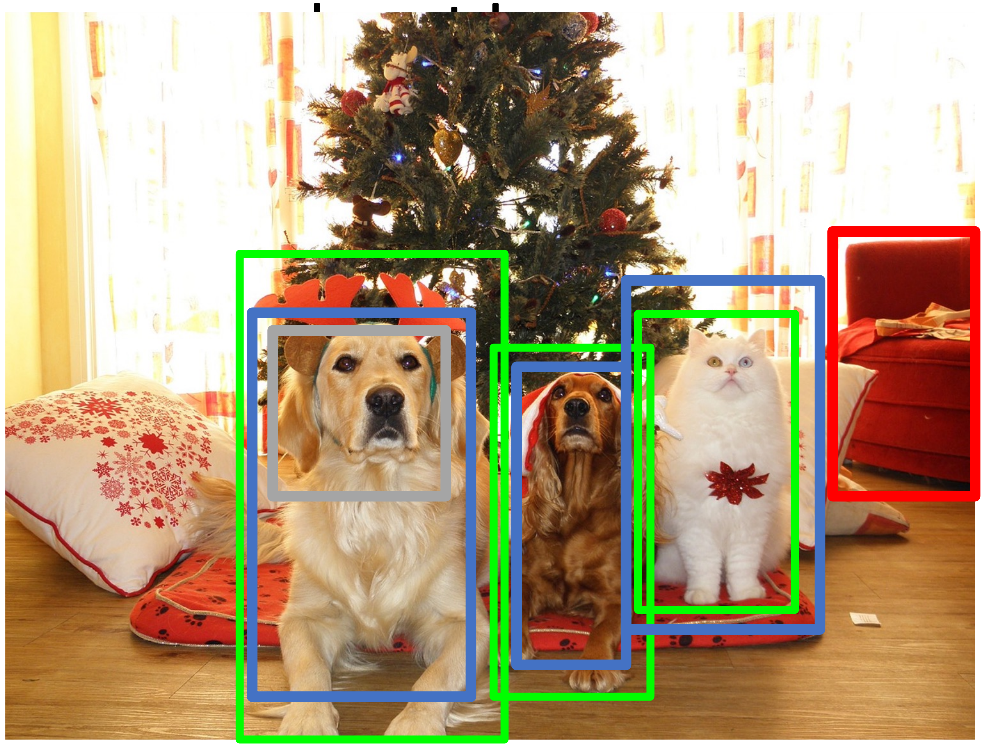
R-CNN Training



Ground-Truth boxes

Region Proposals

R-CNN Training

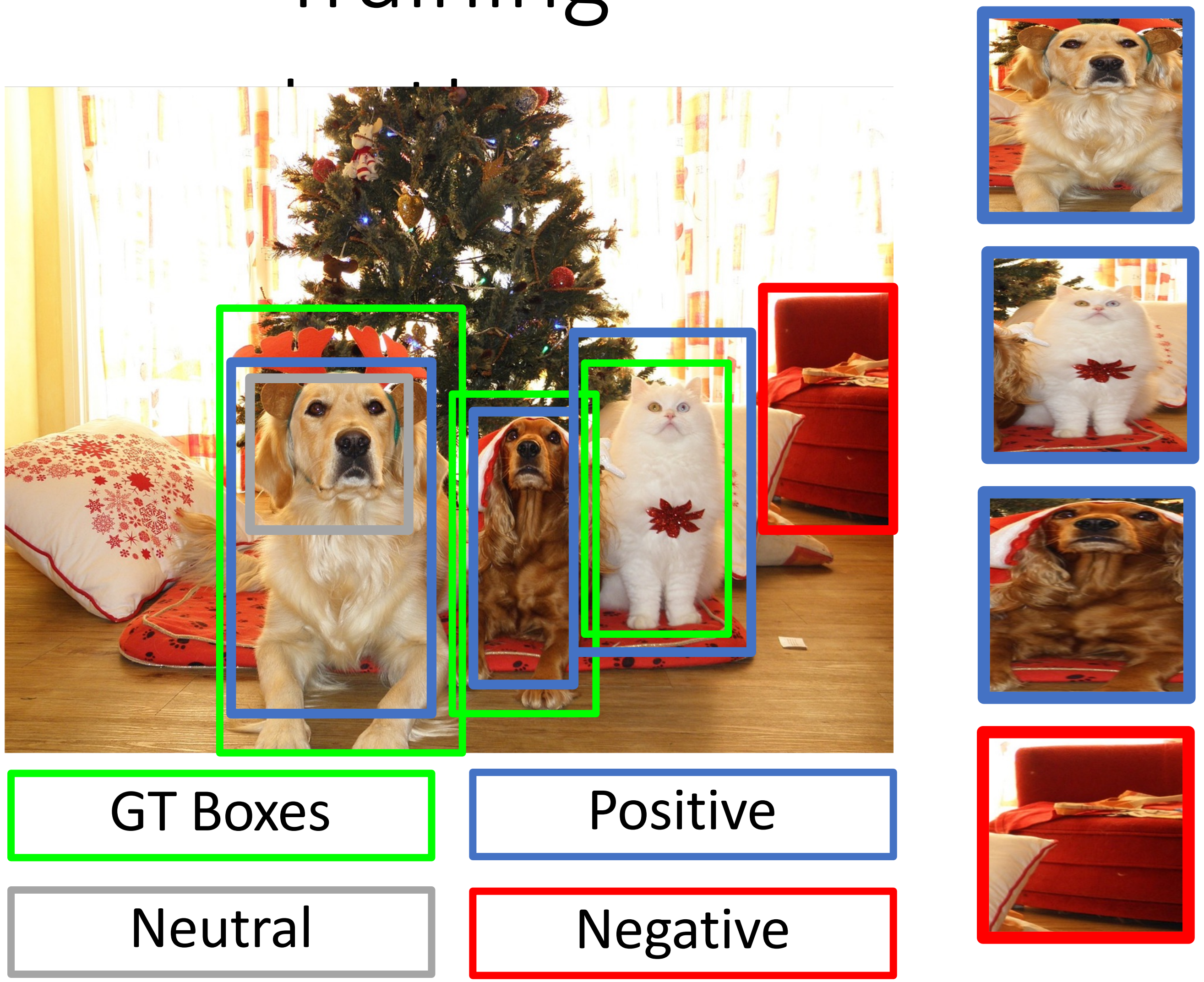


GT Boxes | Positive | Neutral | Negative

Categorize each region proposal as positive, negative, or neutral based on overlap with ground-truth boxes:

Positive: > 0.5 IoU with a GT box
Negative: < 0.3 IoU with all GT boxes
Neutral: between 0.3 and 0.5 IoU with GT boxes

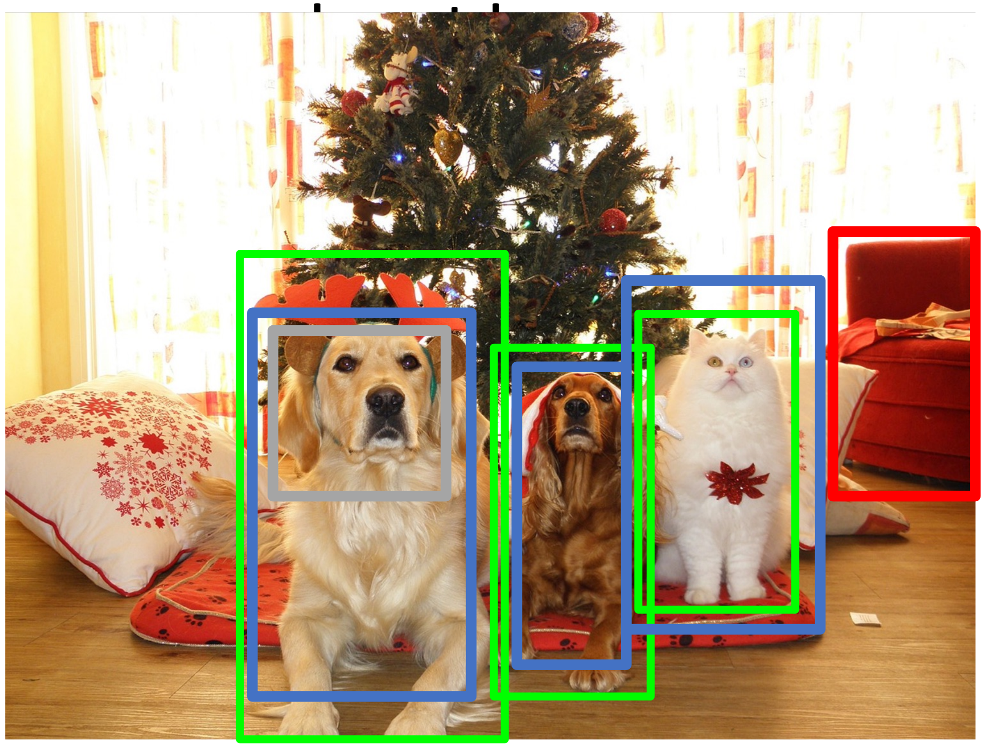
R-CNN
Training



Crop pixels from each positive and negative proposal, resize to 224 x 224

R-CNN Training

Run each region through CNN
Positive regions: predict class and transform
Negative regions: just predict class

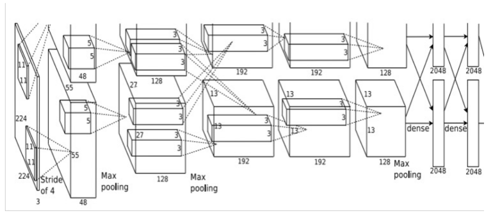


GT Boxes

Positive

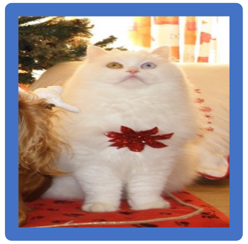
Neutral

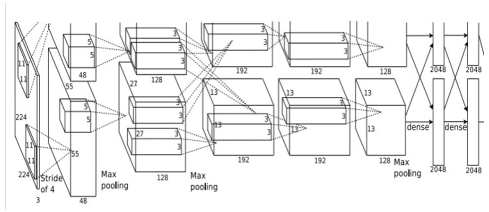
Negative



Class target: Dog

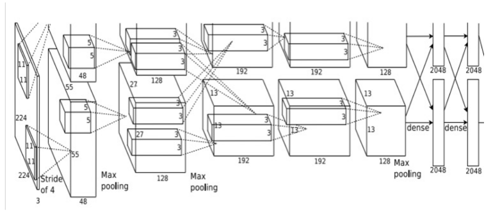
Box target: →





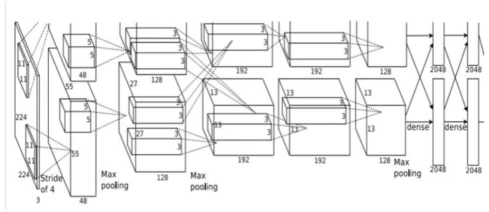
Class target: Cat

Box target: →



Class target: Dog

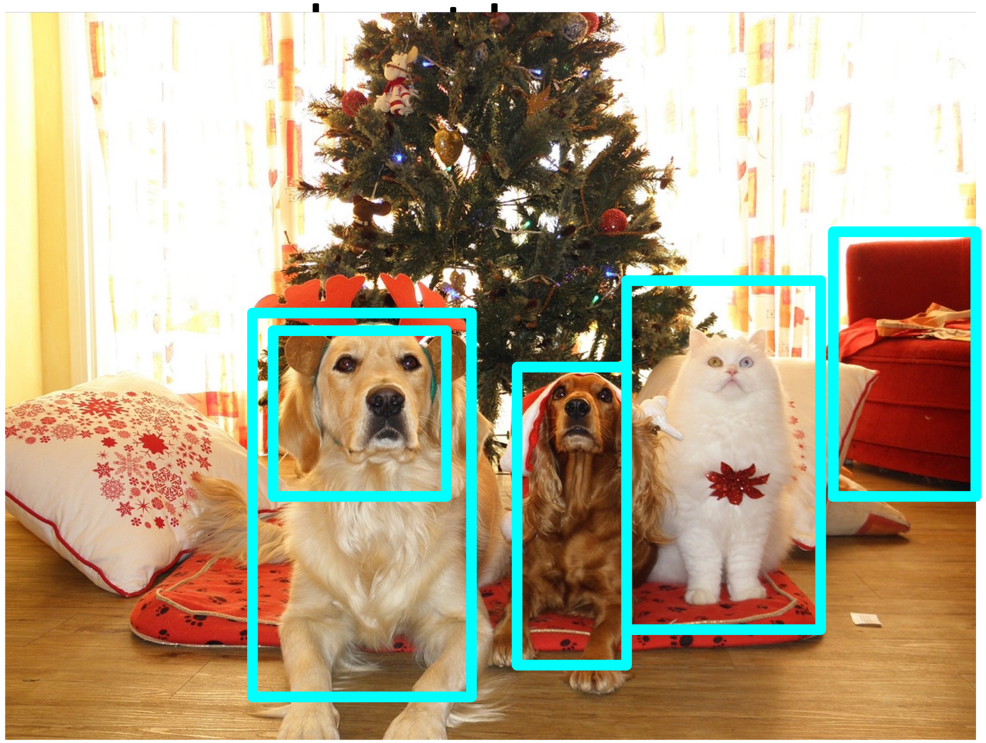
Box target: →



Class target: Background

Box target: None

R-CNN Test-Time



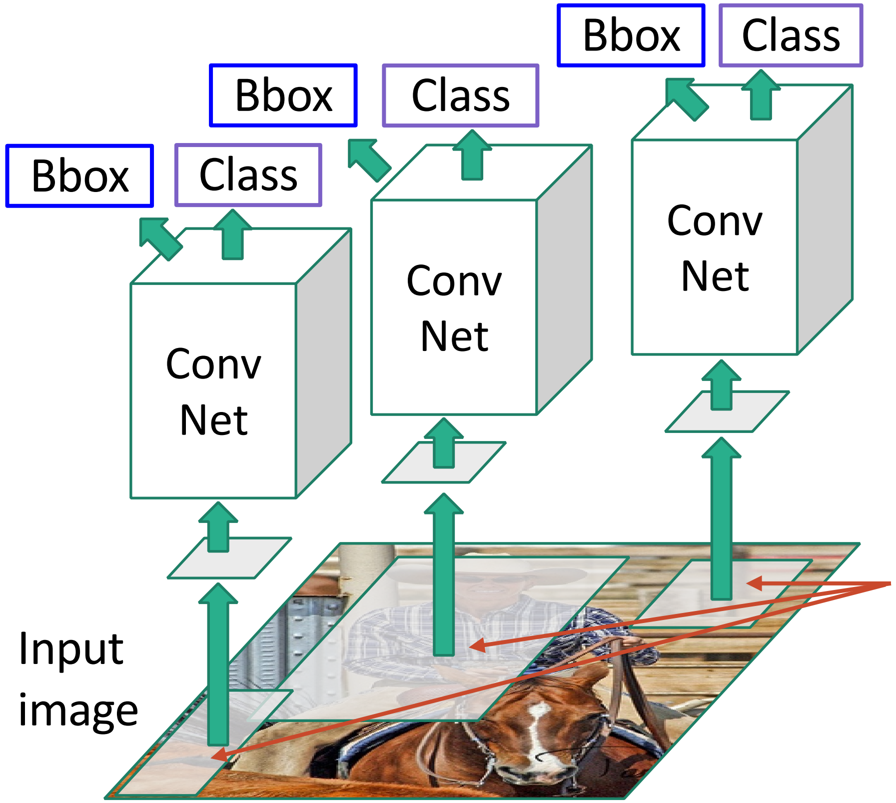
Region Proposals

1. Run proposal method
2. Run CNN on each proposal to get class scores, transforms
3. Threshold class scores to get a set of detections

2 problems:

- CNN often outputs overlapping boxes
  - Non-maximal suppression
- How to set thresholds?
  - Hyper-parameter

R-CNN: Region-Based CNN



Problem: Very slow! Need to do ~2k forward passes for each image!

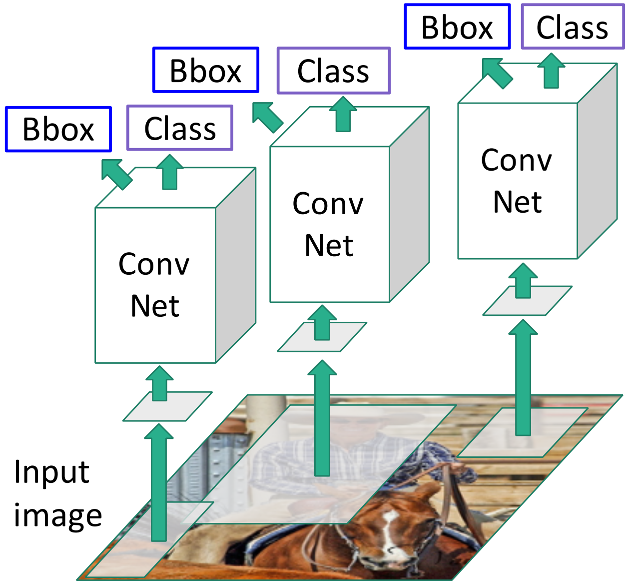
Solution: Run CNN *before* cropping!

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

Today's class

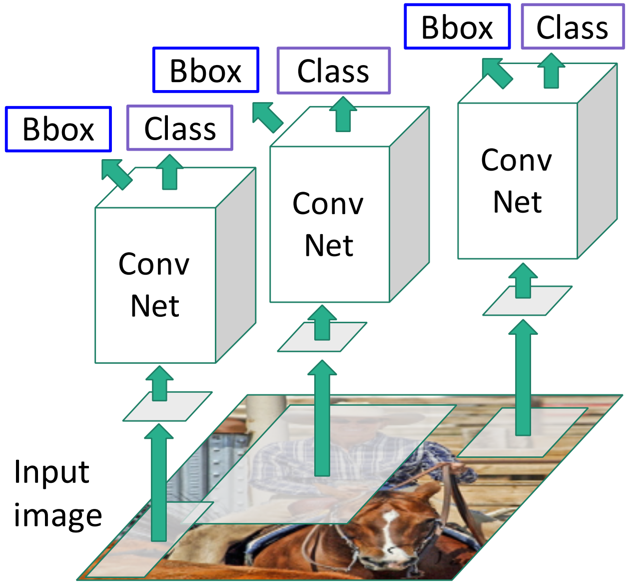
- How do we measure Object Detection accuracy?
- Naïve approaches & R-CNN
- Fast R-CNN
- Region Proposal Network & Faster R-CNN
- Advanced topics:
  - Feature Pyramid Networks to detect at scales
  - Single Shot detection

<u>“Slow” R-CNN</u>
Process each region independently



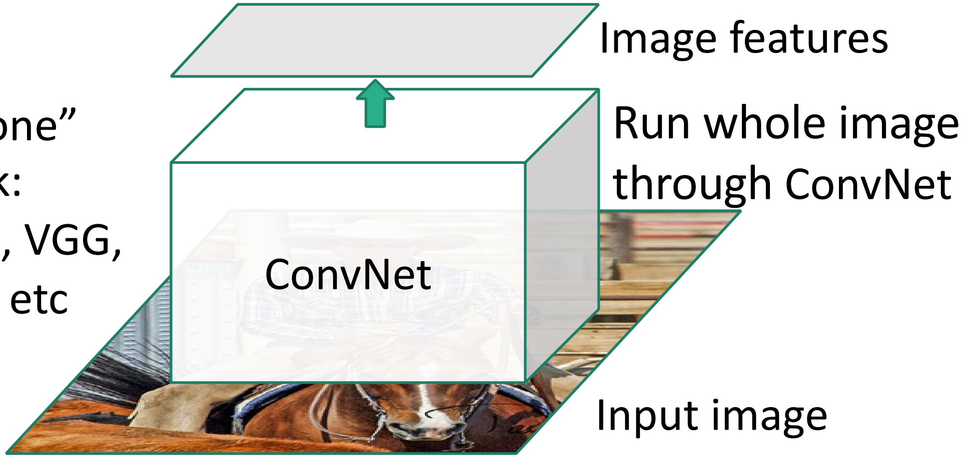
Fast R-CNN

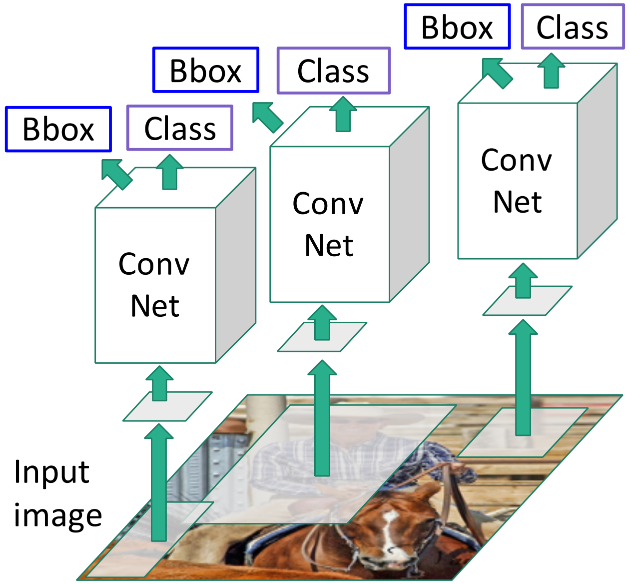
<u>“Slow” R-CNN</u>
Process each region independently



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

Regions of Interest (RoIs) from a proposal method

"Backbone" network: AlexNet, VGG, ResNet, etc

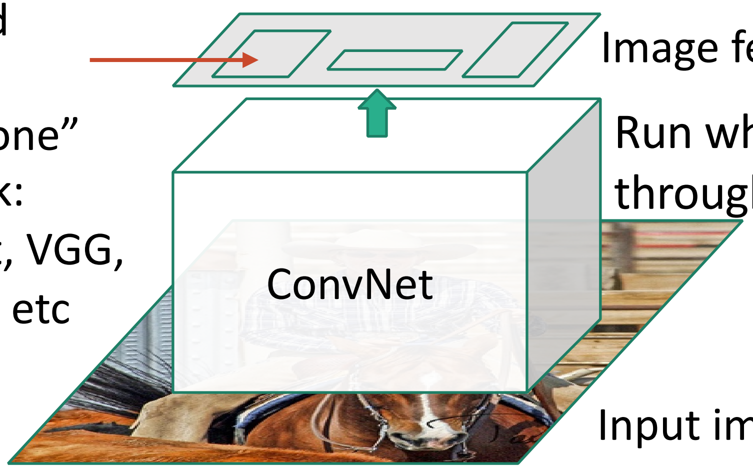
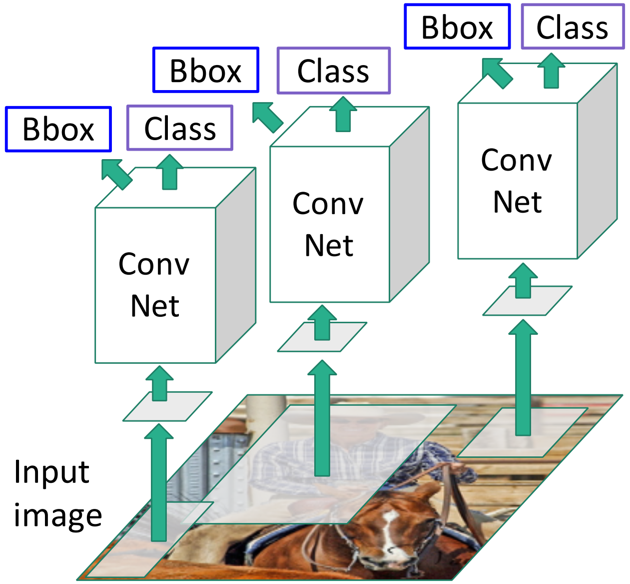


Image features

Run whole image through ConvNet

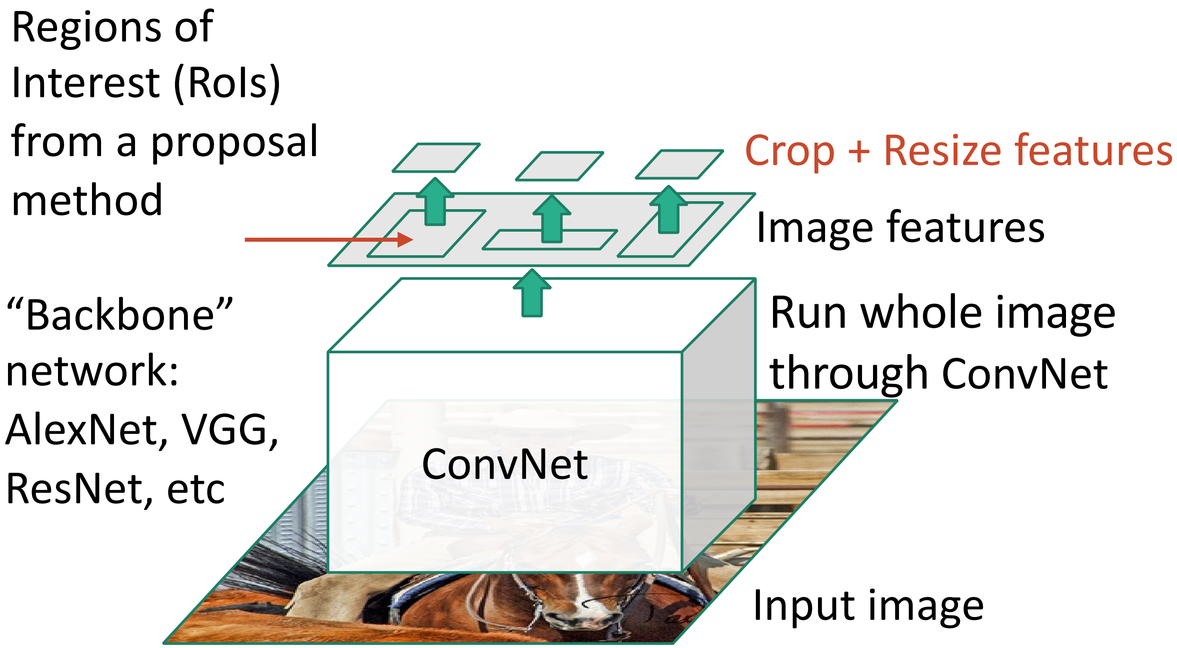
Input image

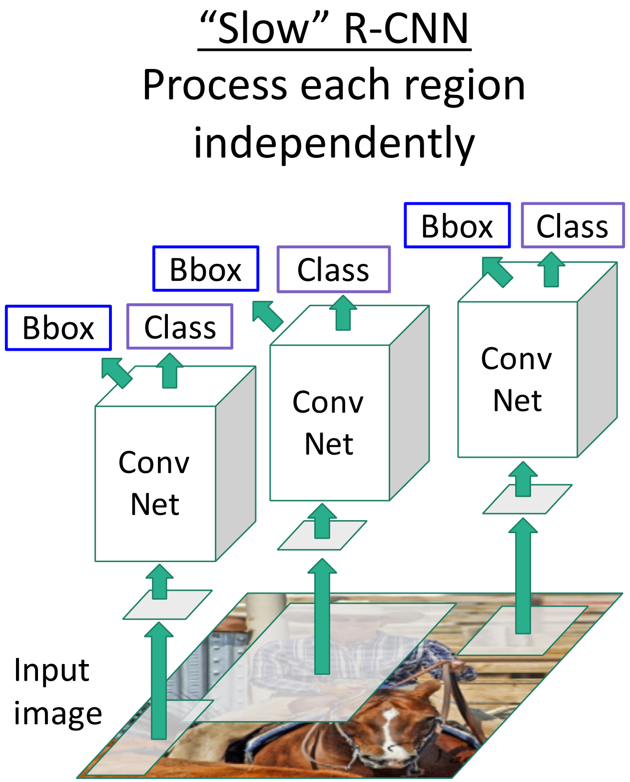
"Slow" R-CNN
Process each region independently



Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

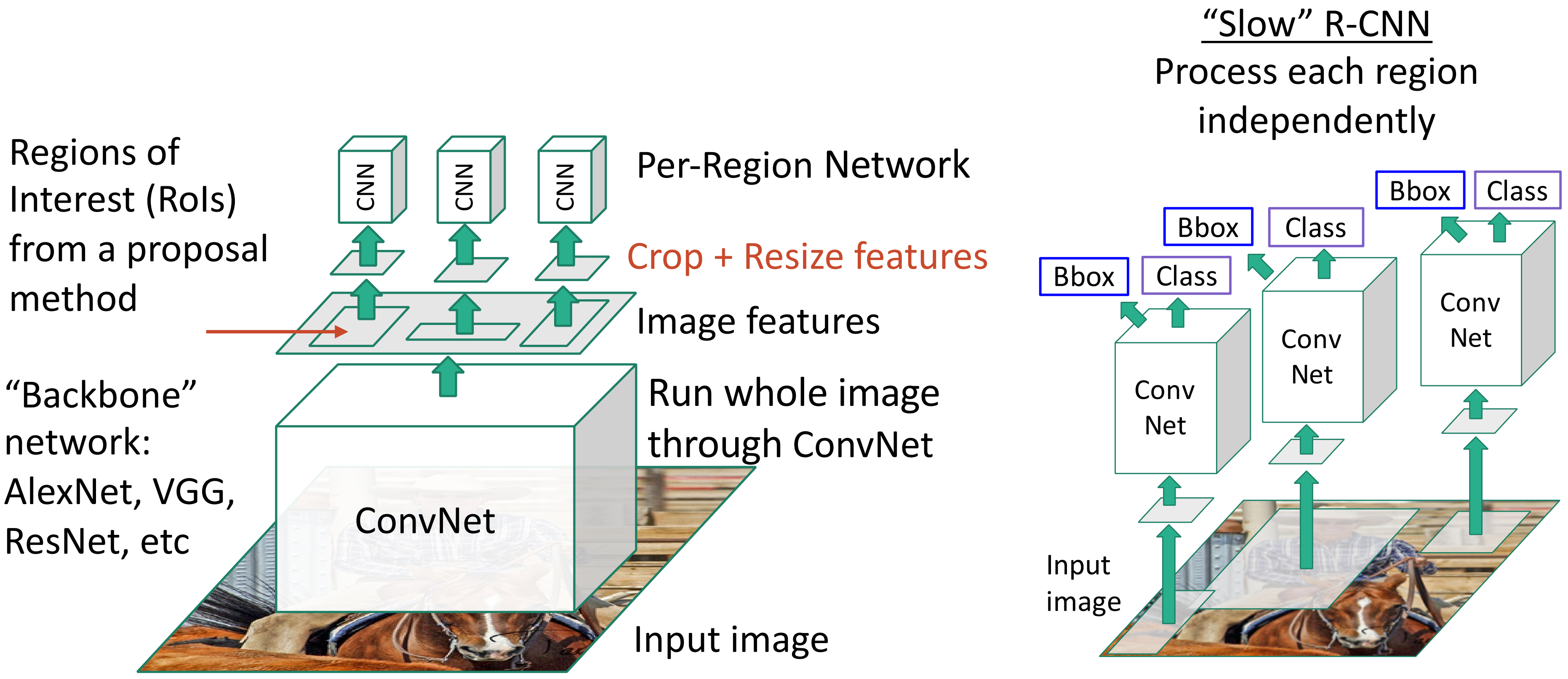
Fast R-CNN





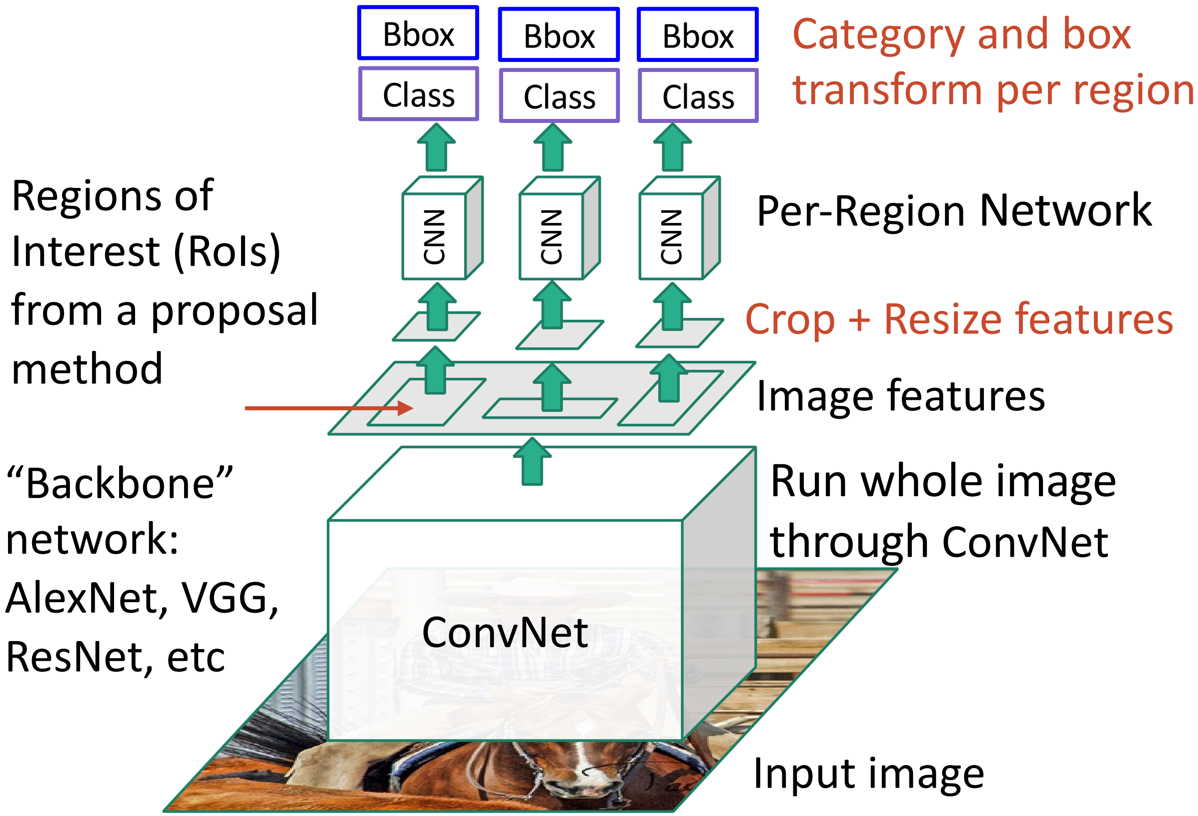
Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

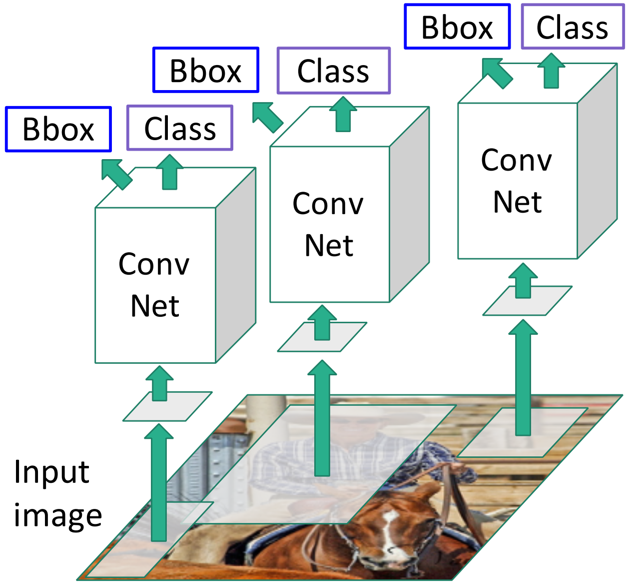


Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

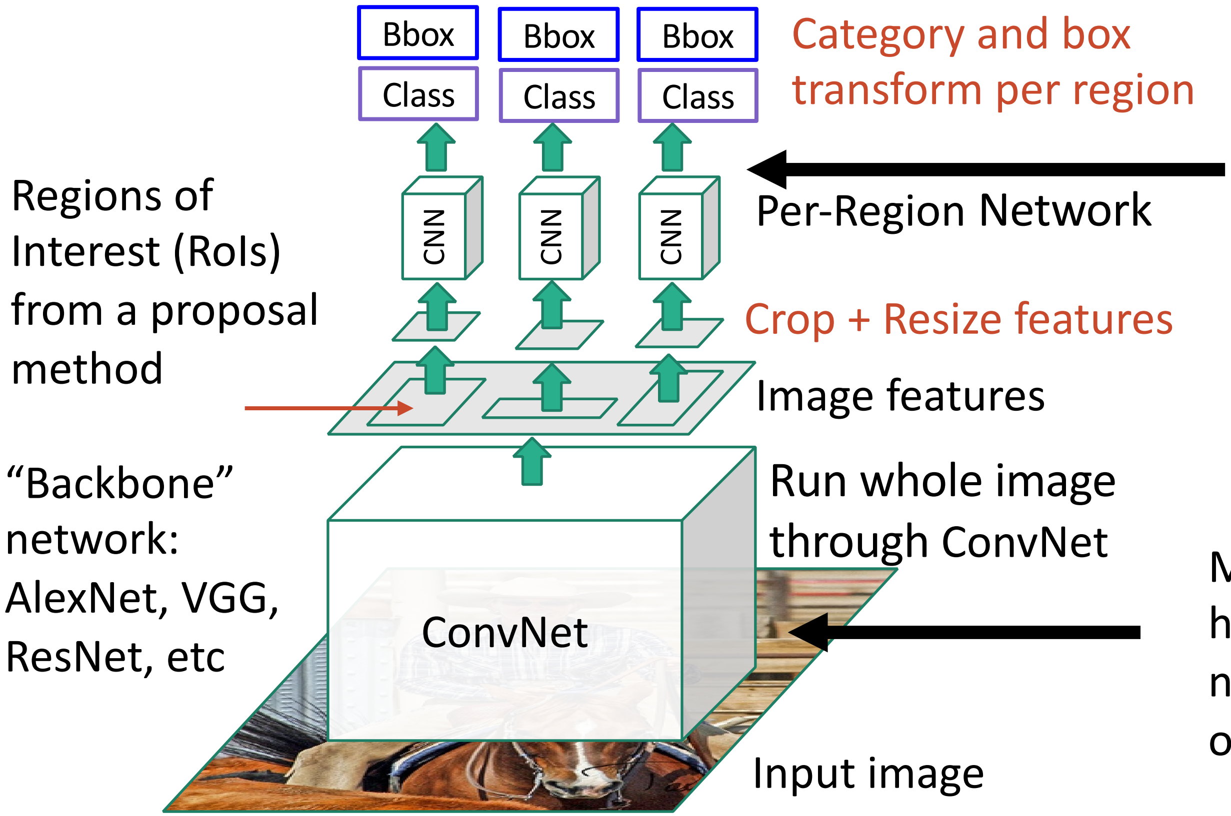
Fast R-CNN



<u>“Slow” R-CNN</u>
Process each region independently



Fast R-CNN

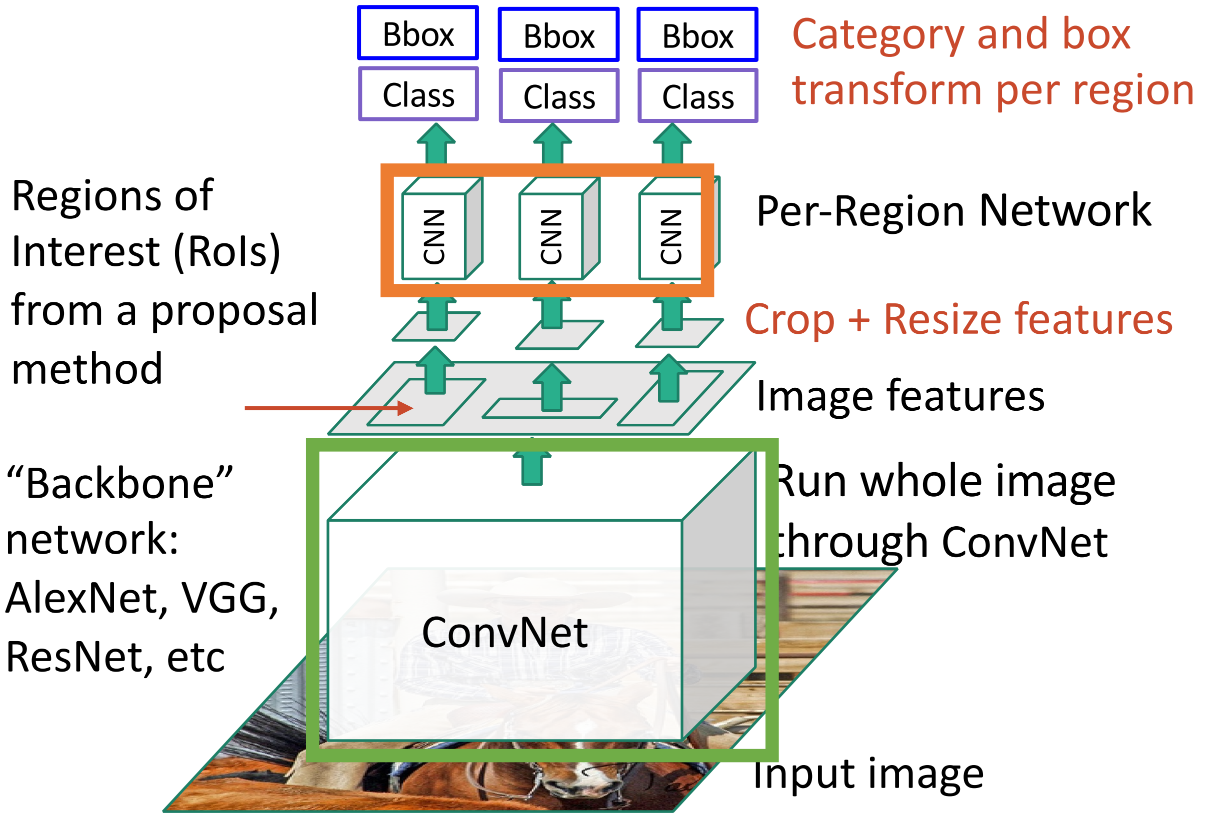


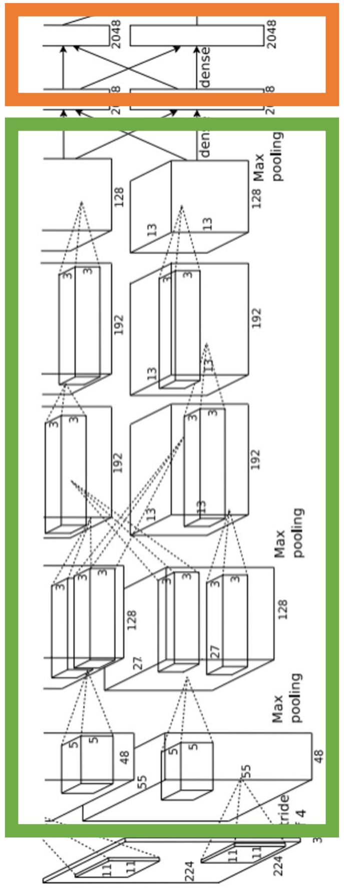
Per-Region network is relatively lightweight

Most of the computation happens in backbone network; this saves work for overlapping region proposals

Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN

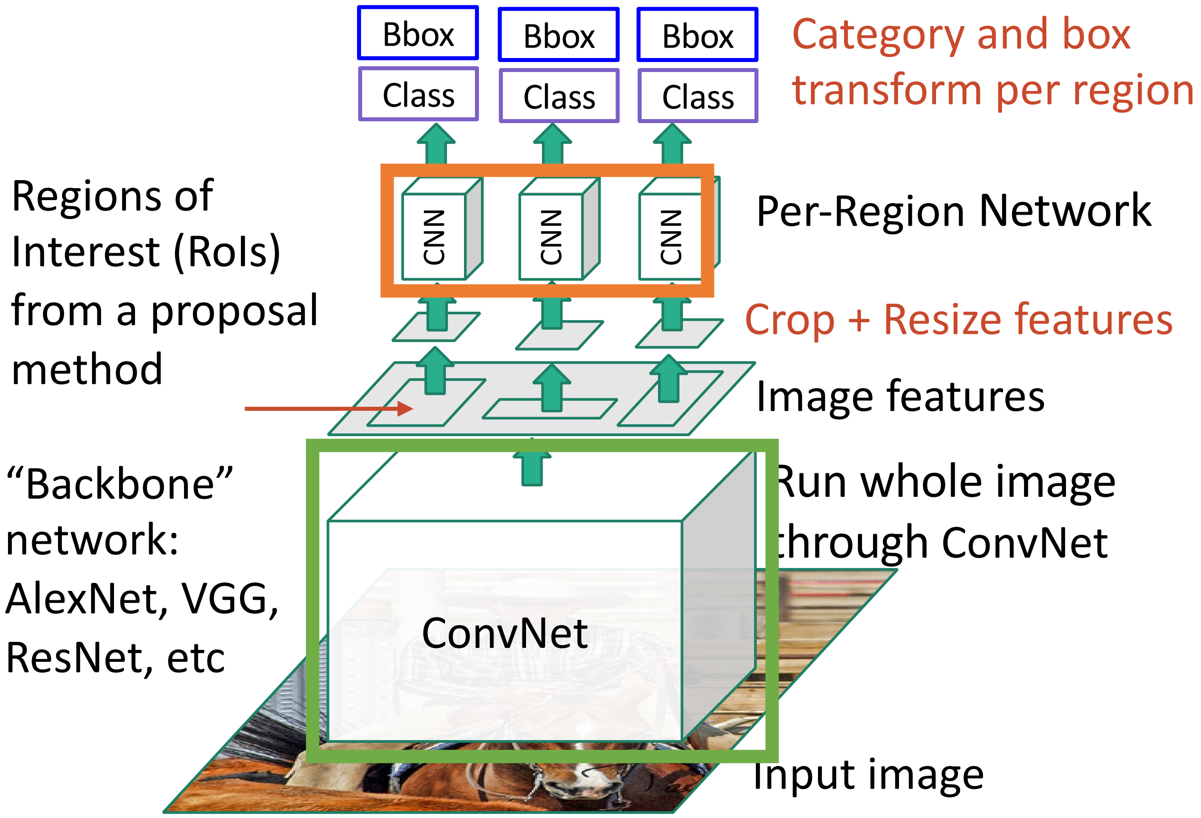


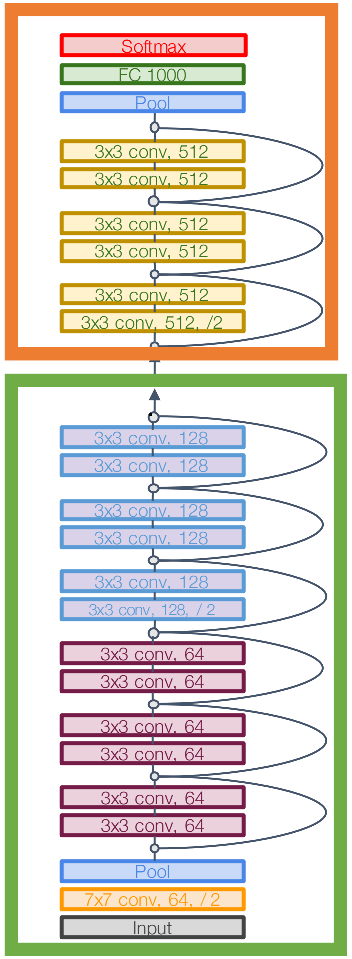


Example:
When using AlexNet for detection, five conv layers are used for backbone and two FC layers are used for per-region network

Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

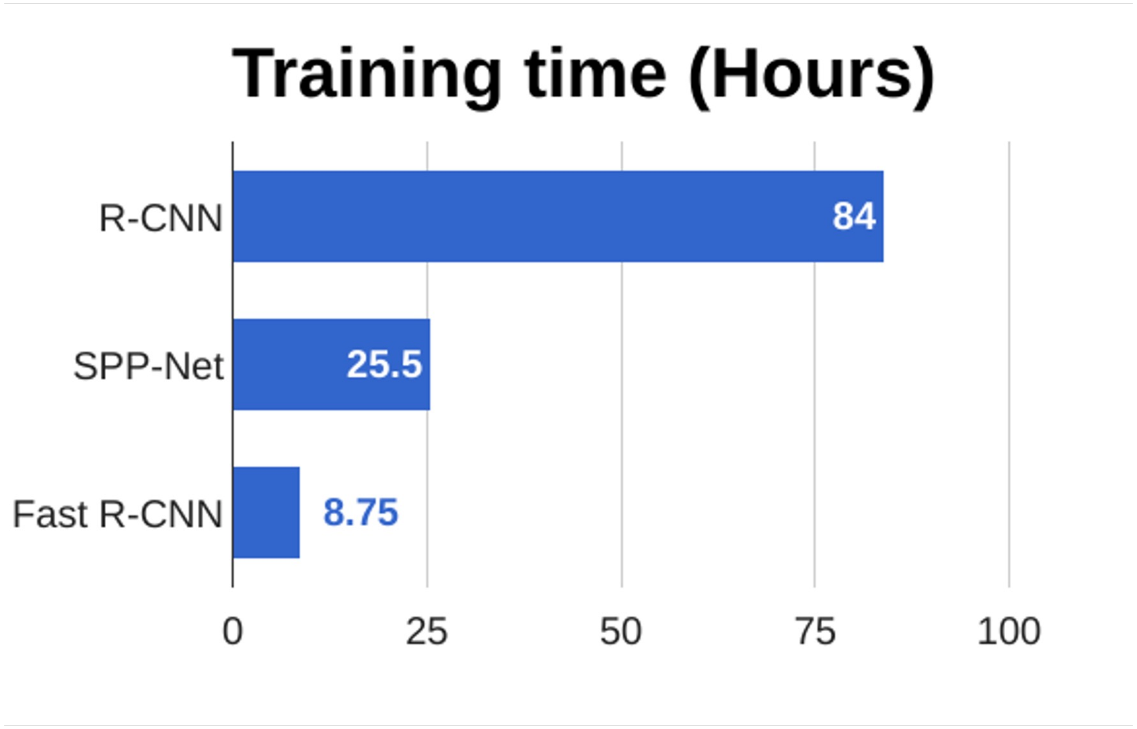
Fast R-CNN

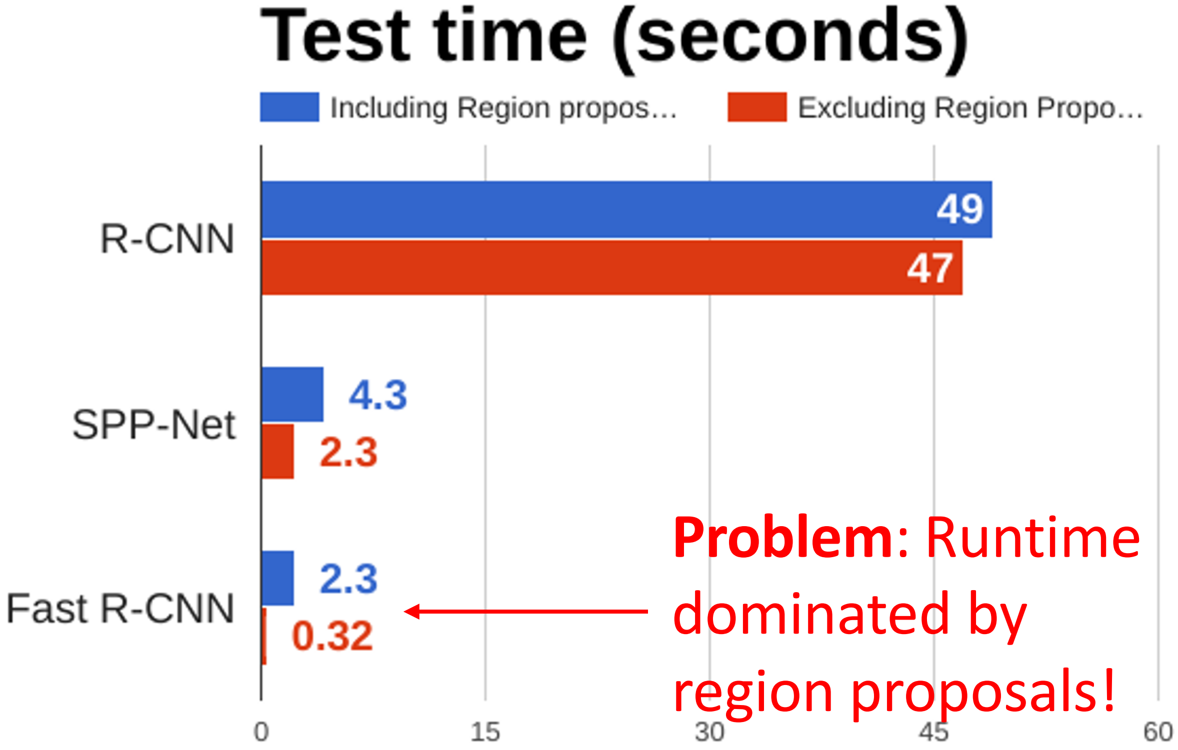




Example:
For ResNet, last stage is used as per-region network; the rest of the network is used as backbone

Fast R-CNN vs "Slow" R-CNN





Recall: Region proposals computed by heuristic "Selective Search" algorithm on CPU -- let's learn them with a CNN instead!

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

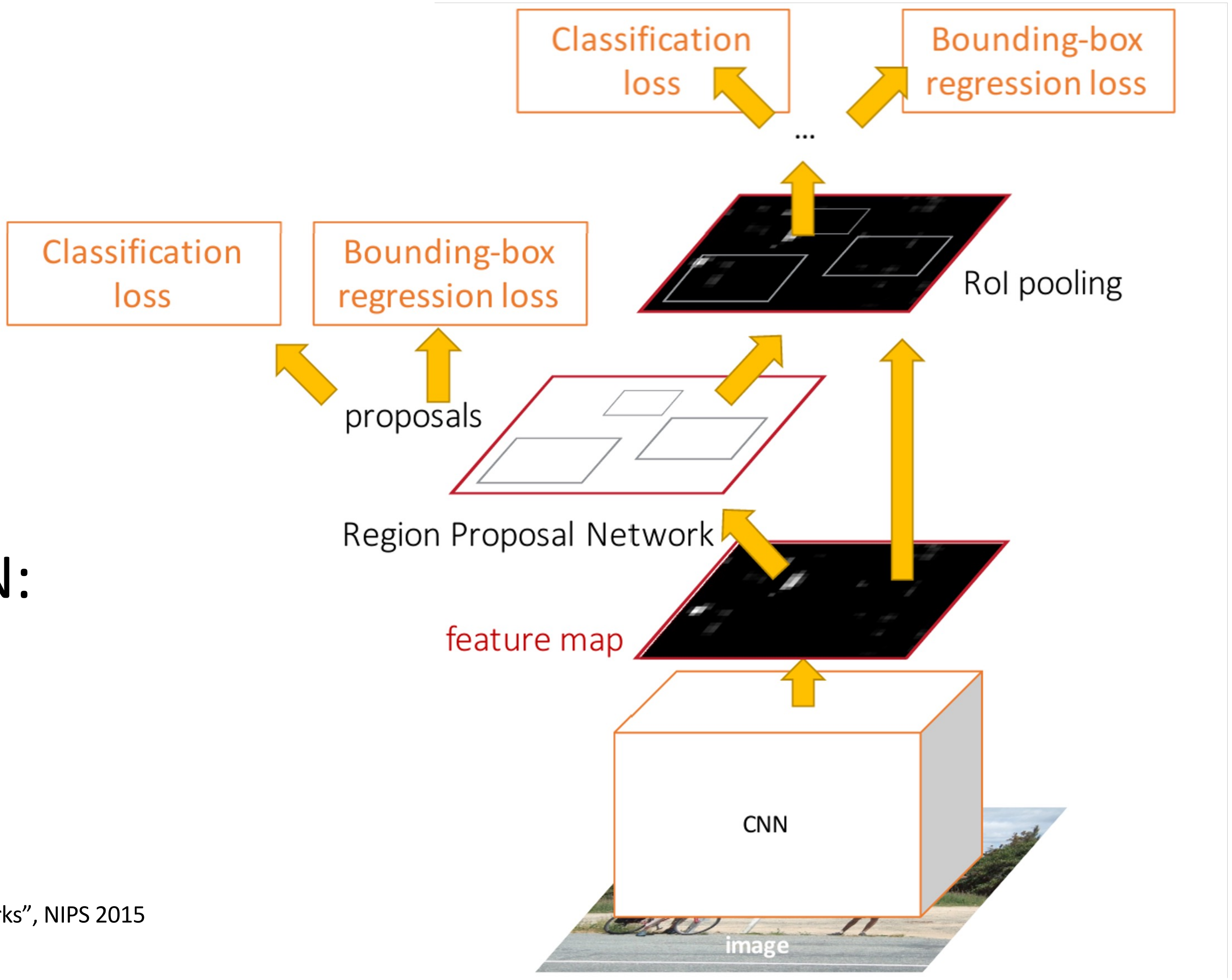
Today's class

- How do we measure Object Detection accuracy?
- Naïve approaches & R-CNN
- Fast R-CNN
- Region Proposal Network & Faster R-CNN
- Advanced topics:
  - Feature Pyramid Networks to detect at scales
  - Single Shot detection

Faster R-CNN: Learnable Region Proposals

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

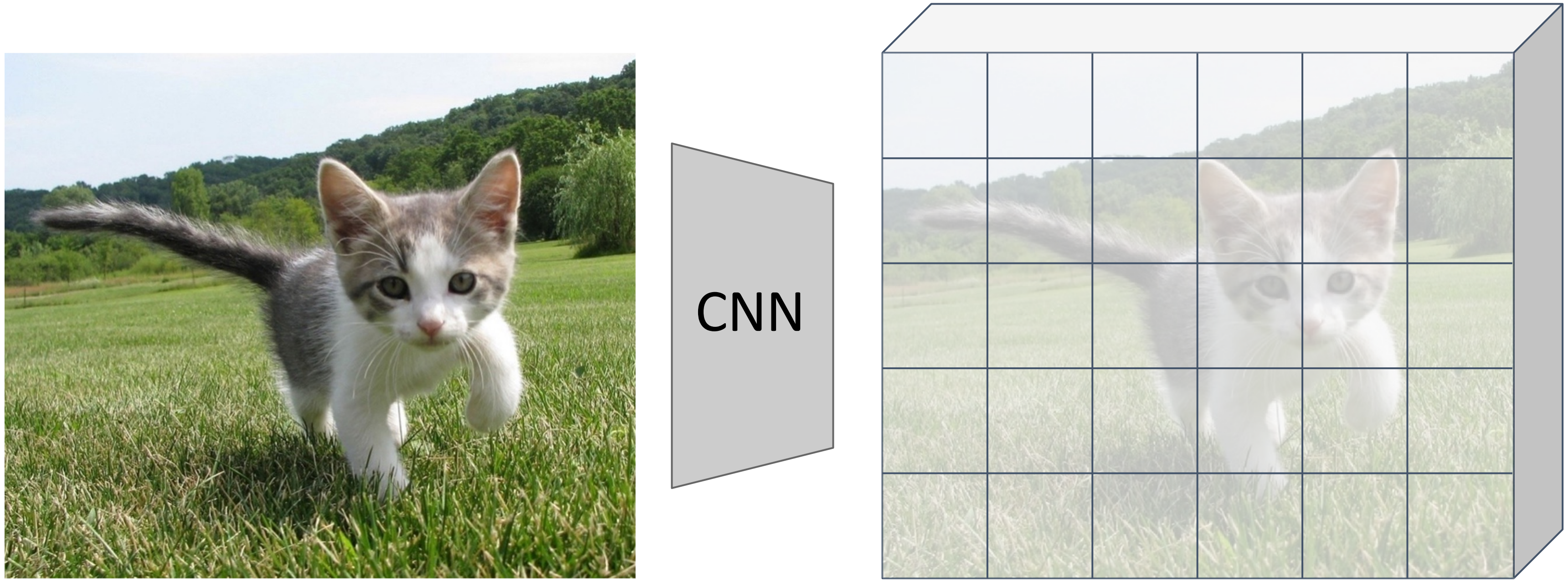
Otherwise same as Fast R-CNN: Crop features for each proposal, classify each one



Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015

Region Proposal Network (RPN)

Run backbone CNN to get
features aligned to input image



Input Image
(e.g. 3 x 640 x 480)

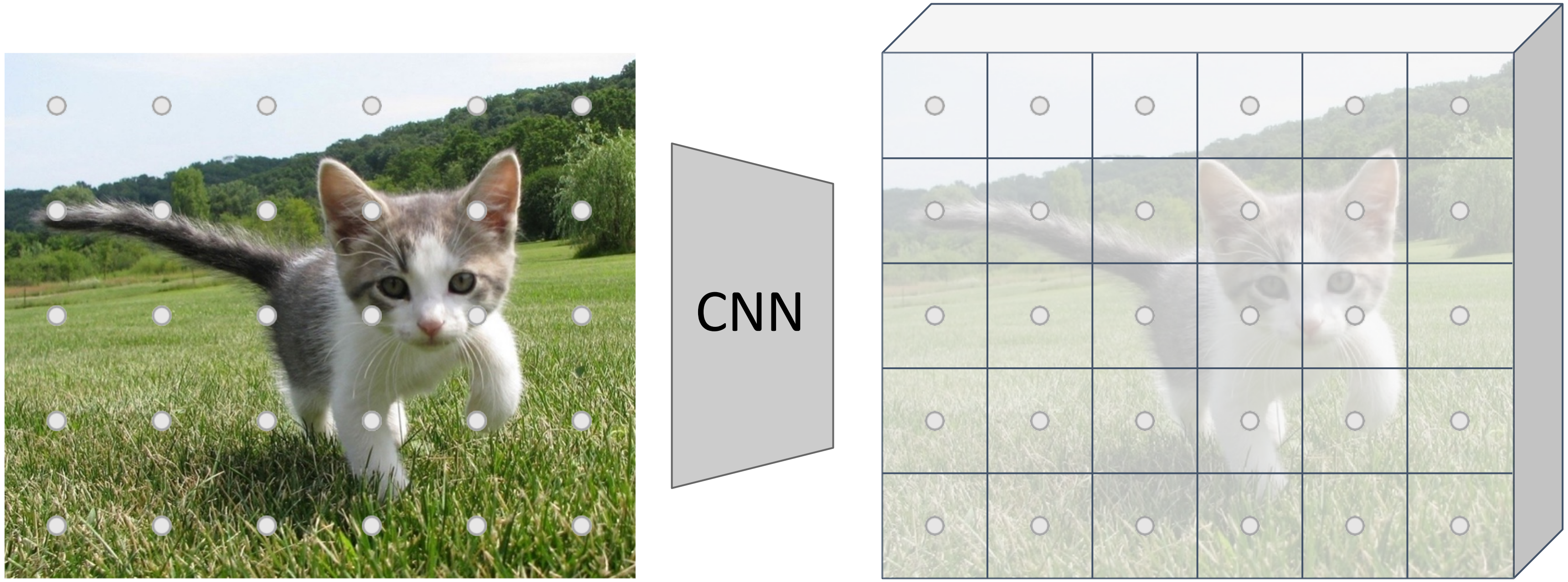
Image features
(e.g. 512 x 5 x 6)

Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015

Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

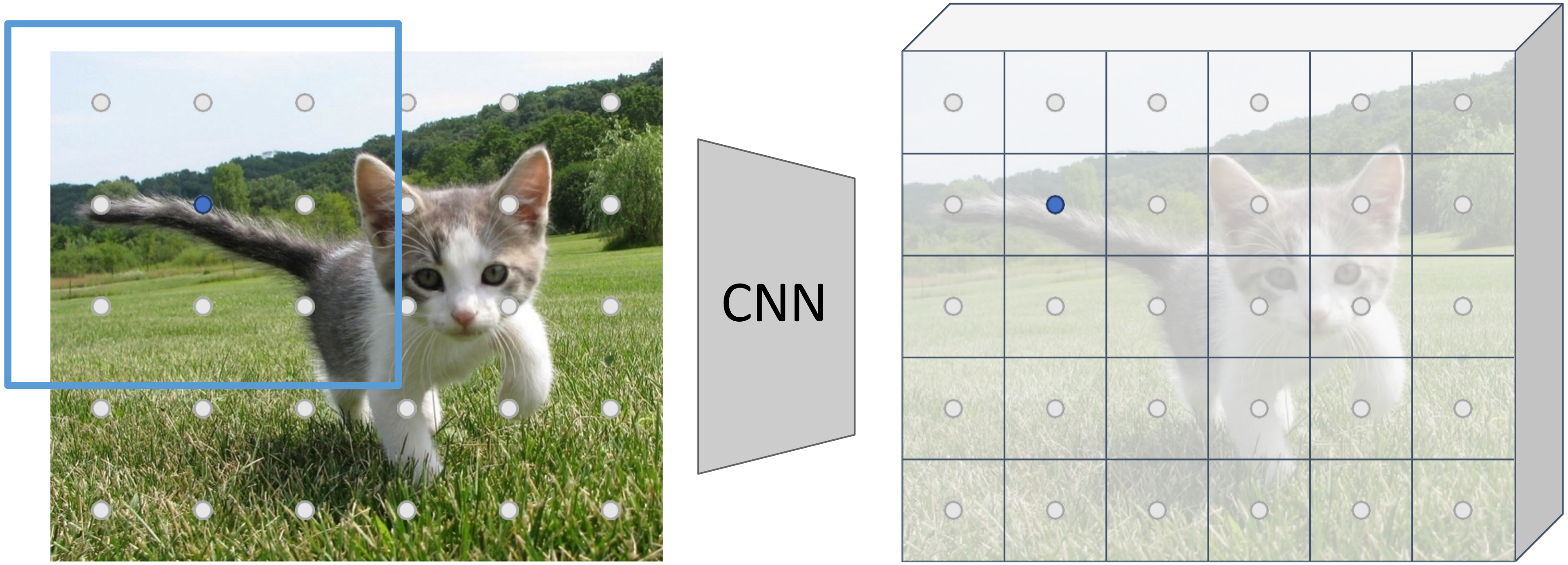
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Region Proposal Network (RPN)

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

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

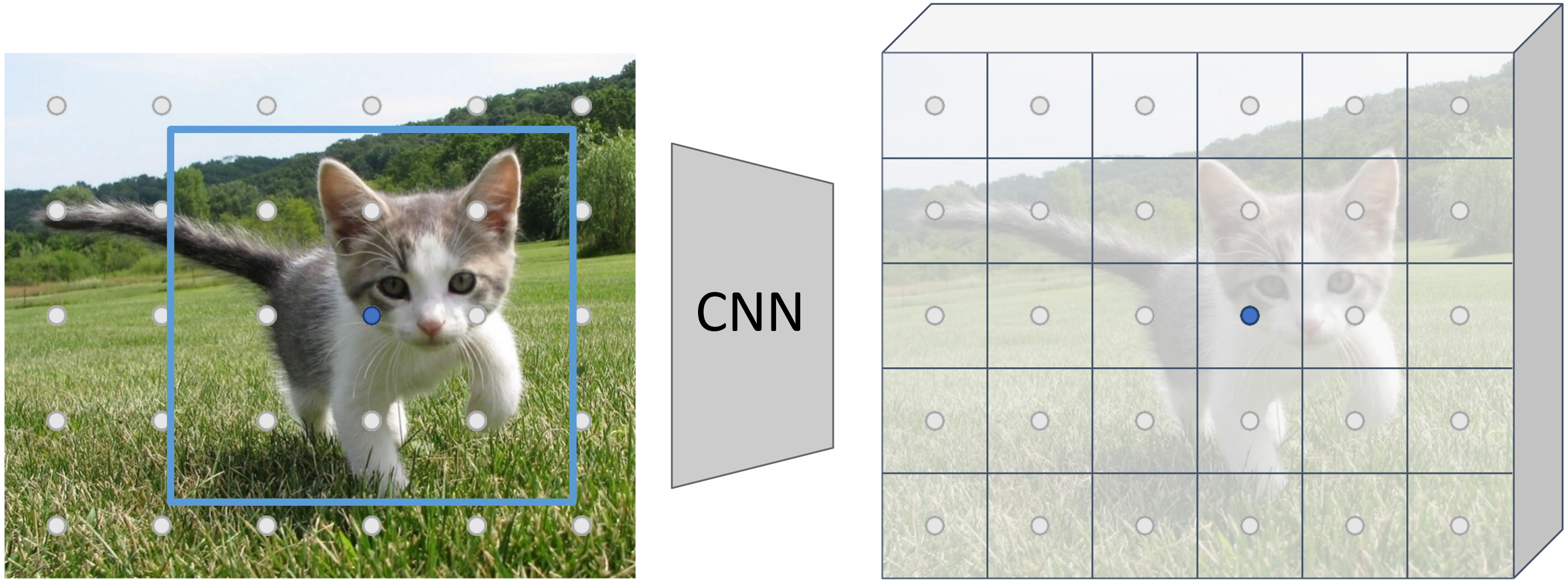
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Region Proposal Network (RPN)

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

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

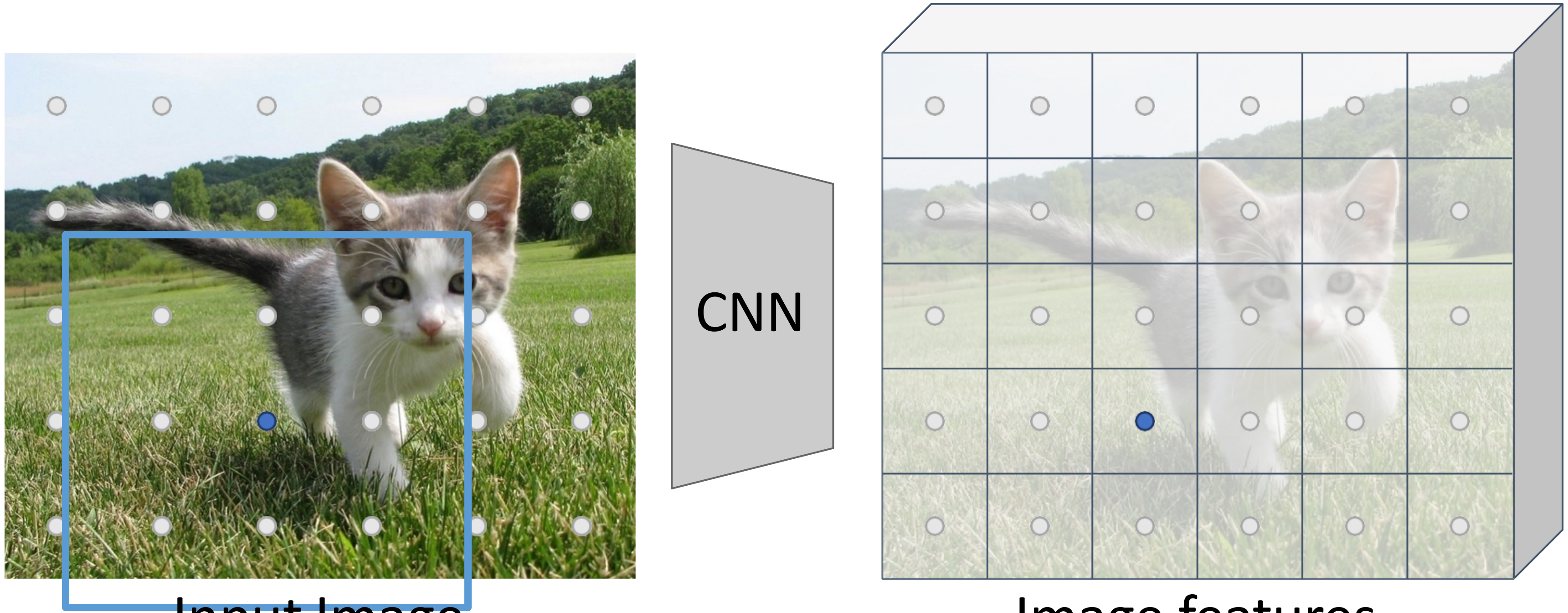
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Region Proposal Network (RPN)

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

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

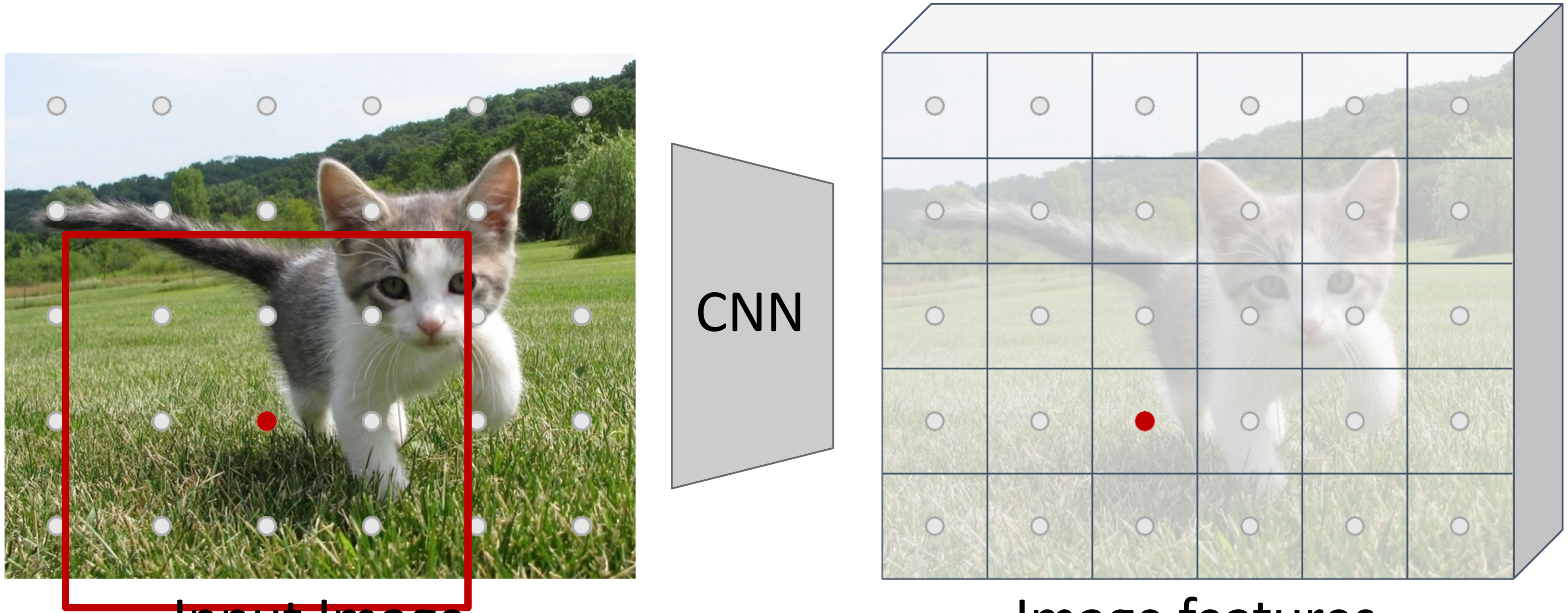
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input

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



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

Classify each anchor as positive (object) or negative (no object)

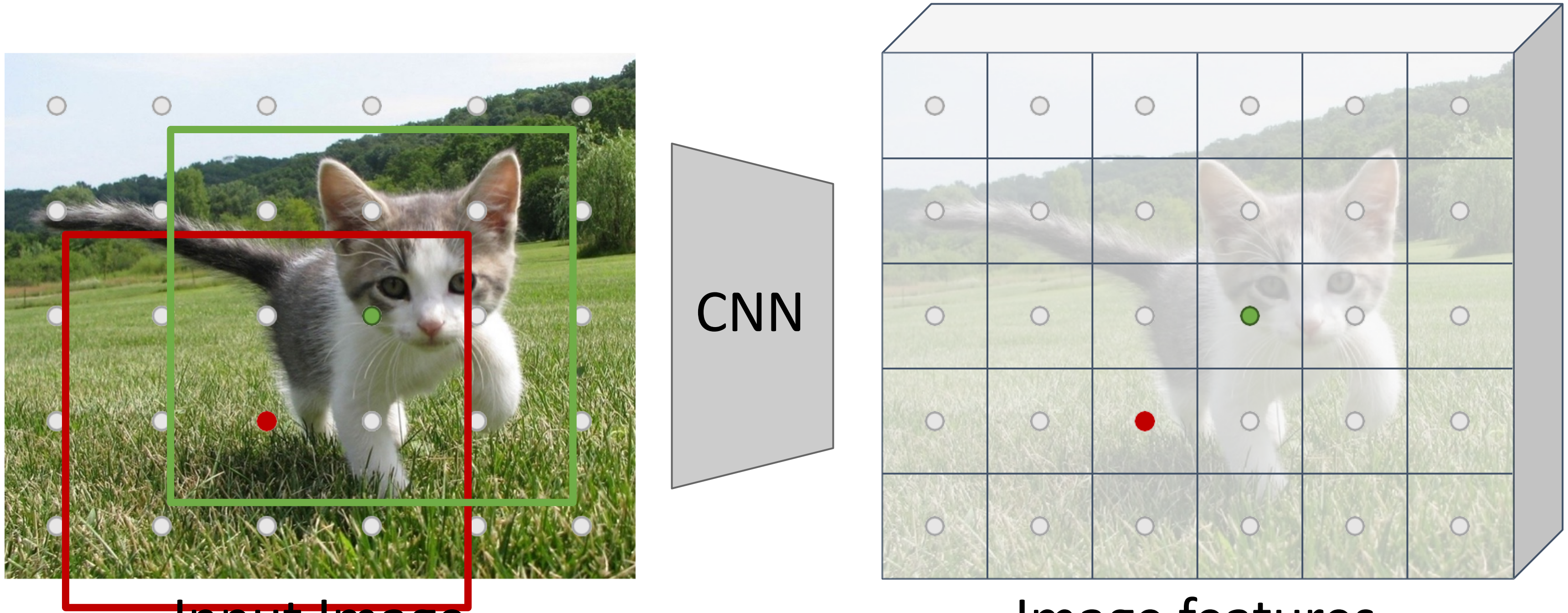
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Region Proposal Network (RPN)

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

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

Classify each anchor as positive (object) or negative (no object)

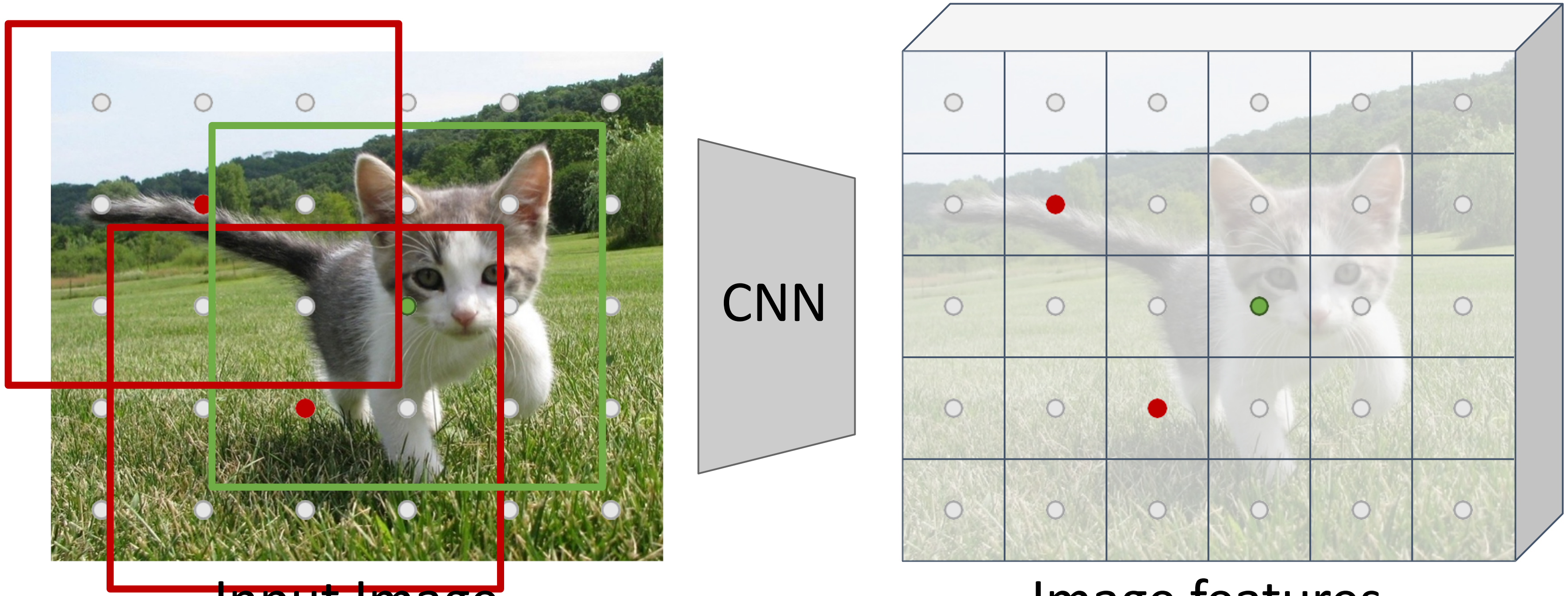
Ren et al, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, NIPS 2015

Region Proposal Network (RPN)

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

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

Classify each anchor as positive (object) or negative (no object)

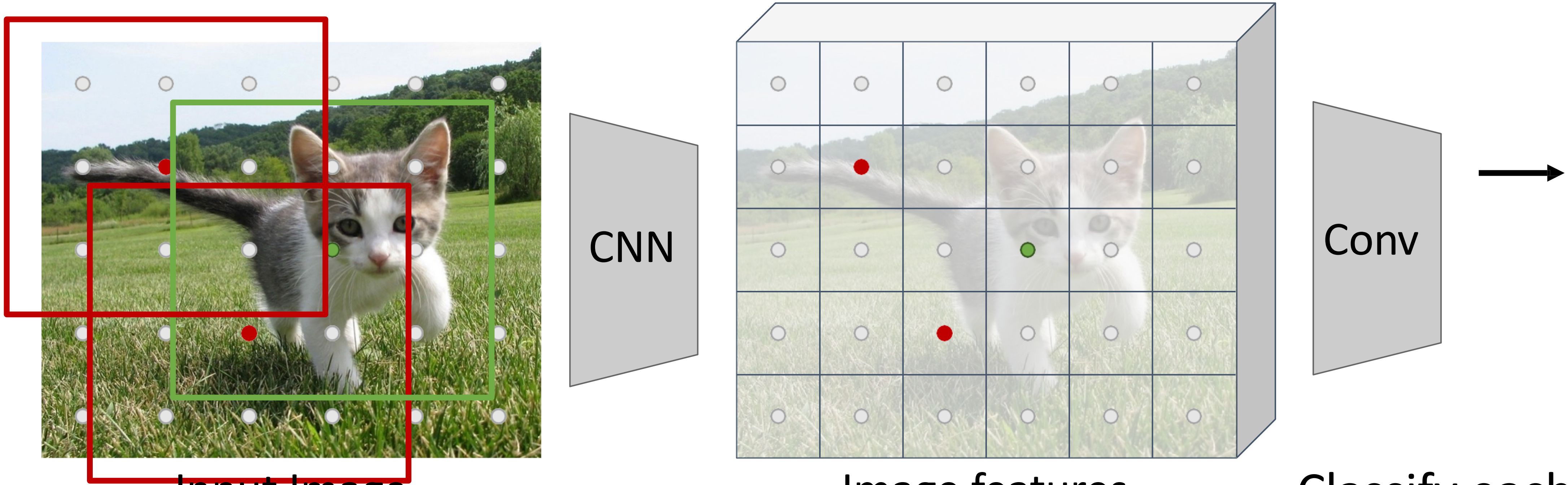
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Region Proposal Network (RPN)

Predict object vs not object scores for all anchors with a conv layer (512 input filters, 2 output filters)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Anchor is object?
2 x 5 x 6

Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

Classify each anchor as positive (object) or negative (no object)

Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

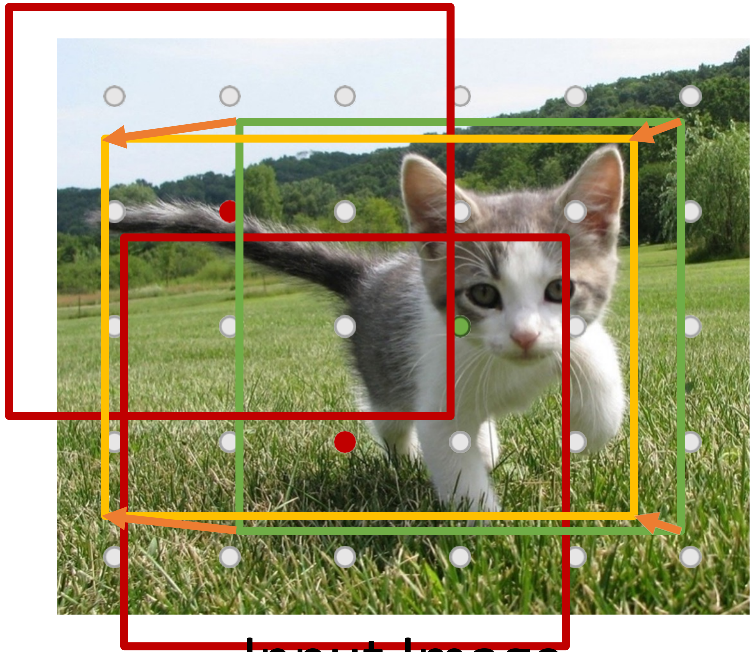
Region Proposal Network (RPN)

For positive anchors, also predict a transform that converting the anchor to the GT box (like R-CNN)

Predict transforms with conv

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

CNN

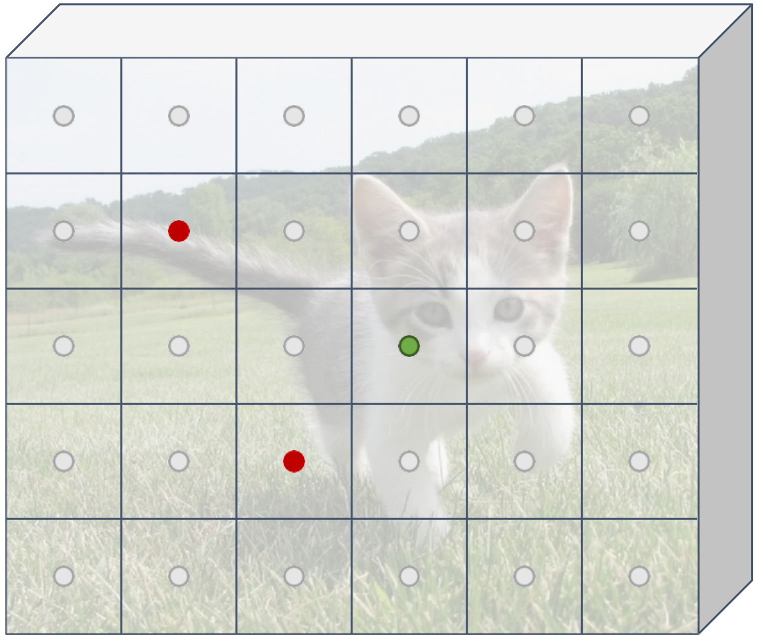


Image features
(e.g. 512 x 5 x 6)

Conv

Anchor is object?
2 x 5 x 6

Anchor transforms
4 x 5 x 6

Classify each anchor as positive (object) or negative (no object)

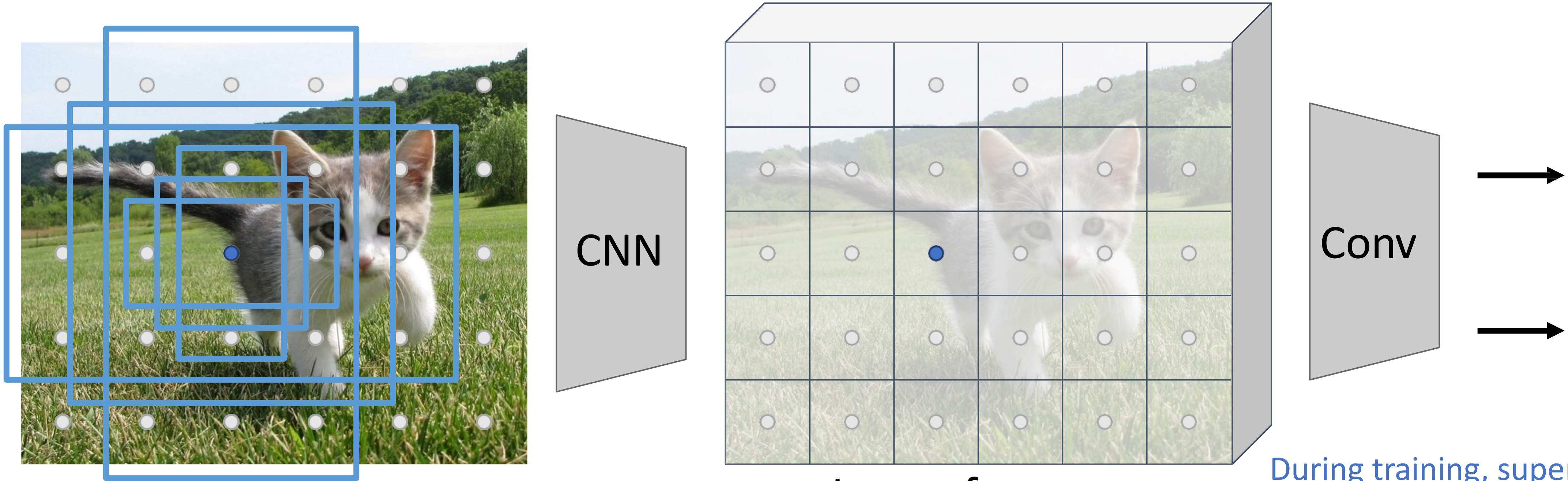
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Region Proposal Network (RPN)

In practice: Rather than using one anchor per point, instead consider K different anchors with different size and scale (here K = 6)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Anchor is object?
2K x 5 x 6

Anchor transforms
4K x 5 x 6

Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

During training, supervised positive / negative anchors and box transforms like R-CNN

Positive anchors: >= 0.7 IoU with some GT box (plus highest IoU to each GT)

Negative anchors: < 0.3 IoU with all GT boxes. Don't supervised transforms for negative boxes.

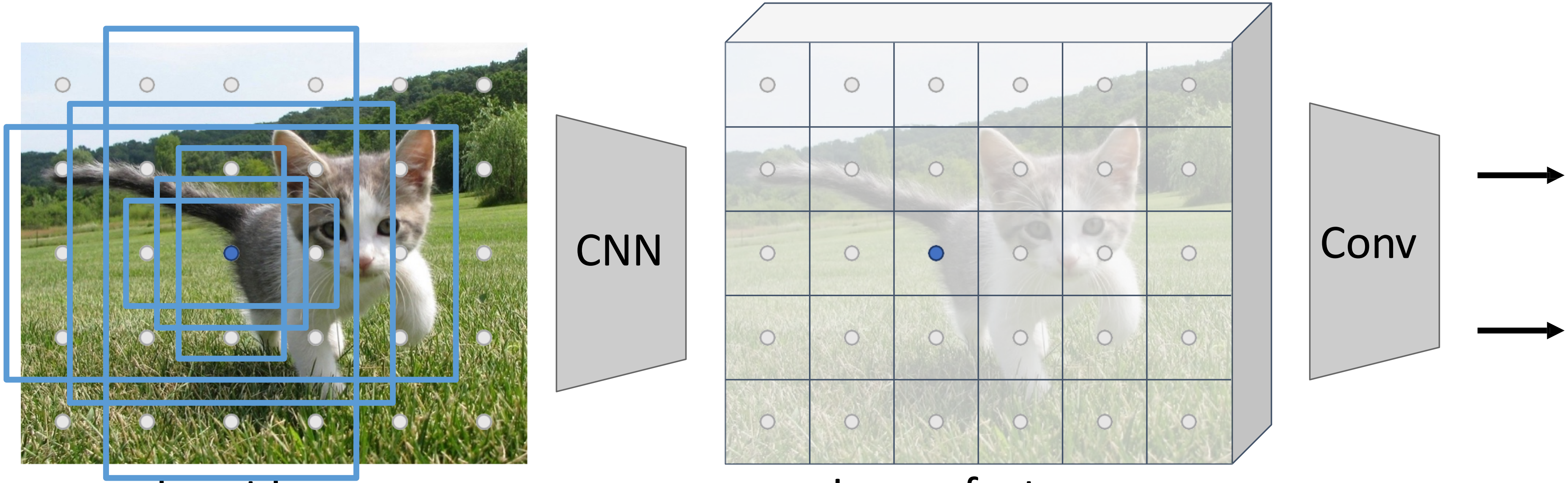
Neutral anchors: between 0.3 and 0.7 IoU with all GT boxes; ignored during training

Region Proposal Network (RPN)

In practice: Rather than using one anchor per point, instead consider K different anchors with different size and scale (here K = 6)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Anchor is object?
2K x 5 x 6

Anchor transforms
4K x 5 x 6

Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

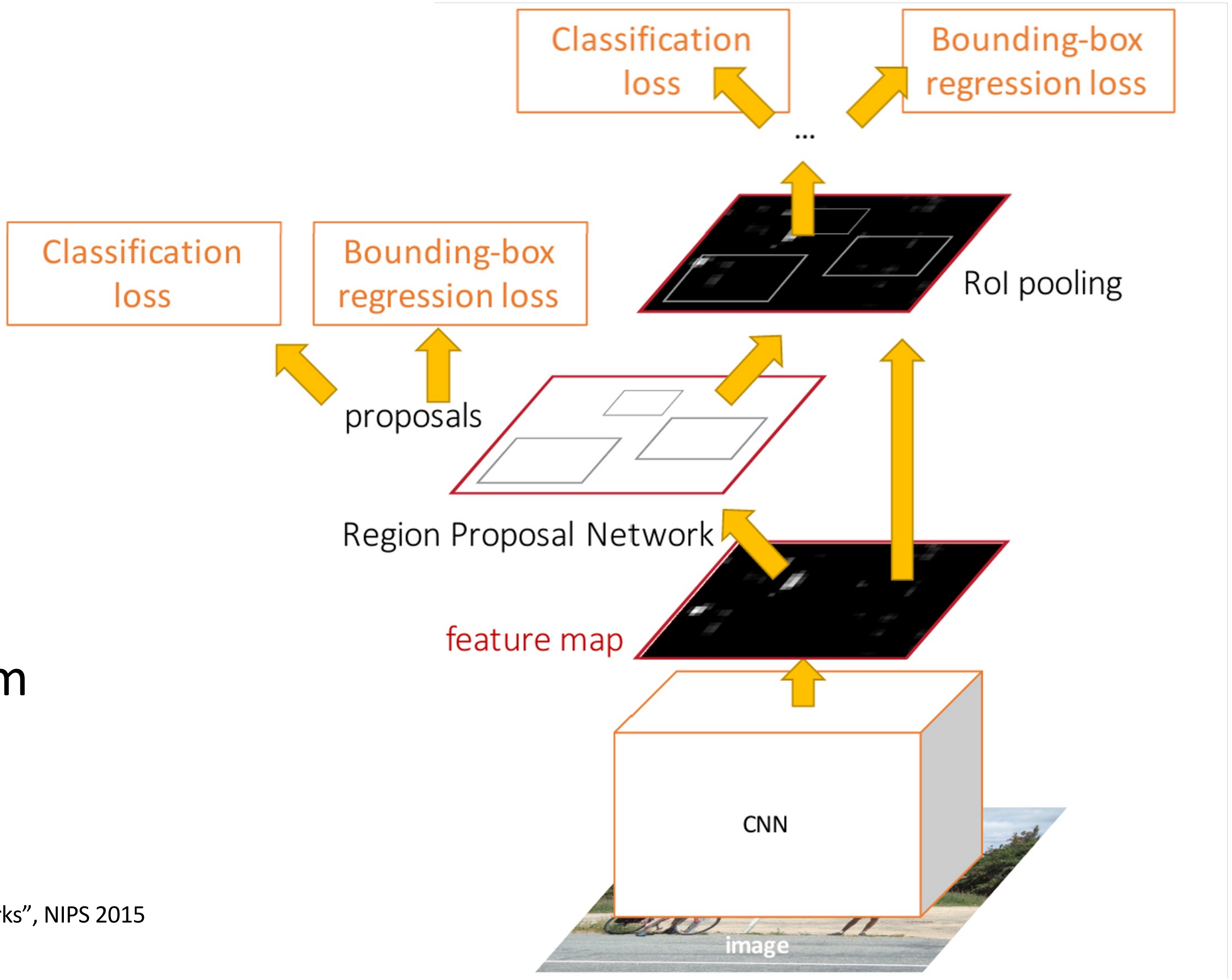
At test-time, sort all K*5*6 boxes by their positive score, take top 300 as our region proposals

Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Fast<u>er</u> R-CNN: Learnable Region Proposals

Jointly train with 4 losses:

1. **RPN classification**: anchor box is object / not an object
2. **RPN regression**: predict transform from anchor box to proposal box
3. **Object classification**: classify proposals as background / object class
4. **Object regression**: predict transform from proposal box to object box



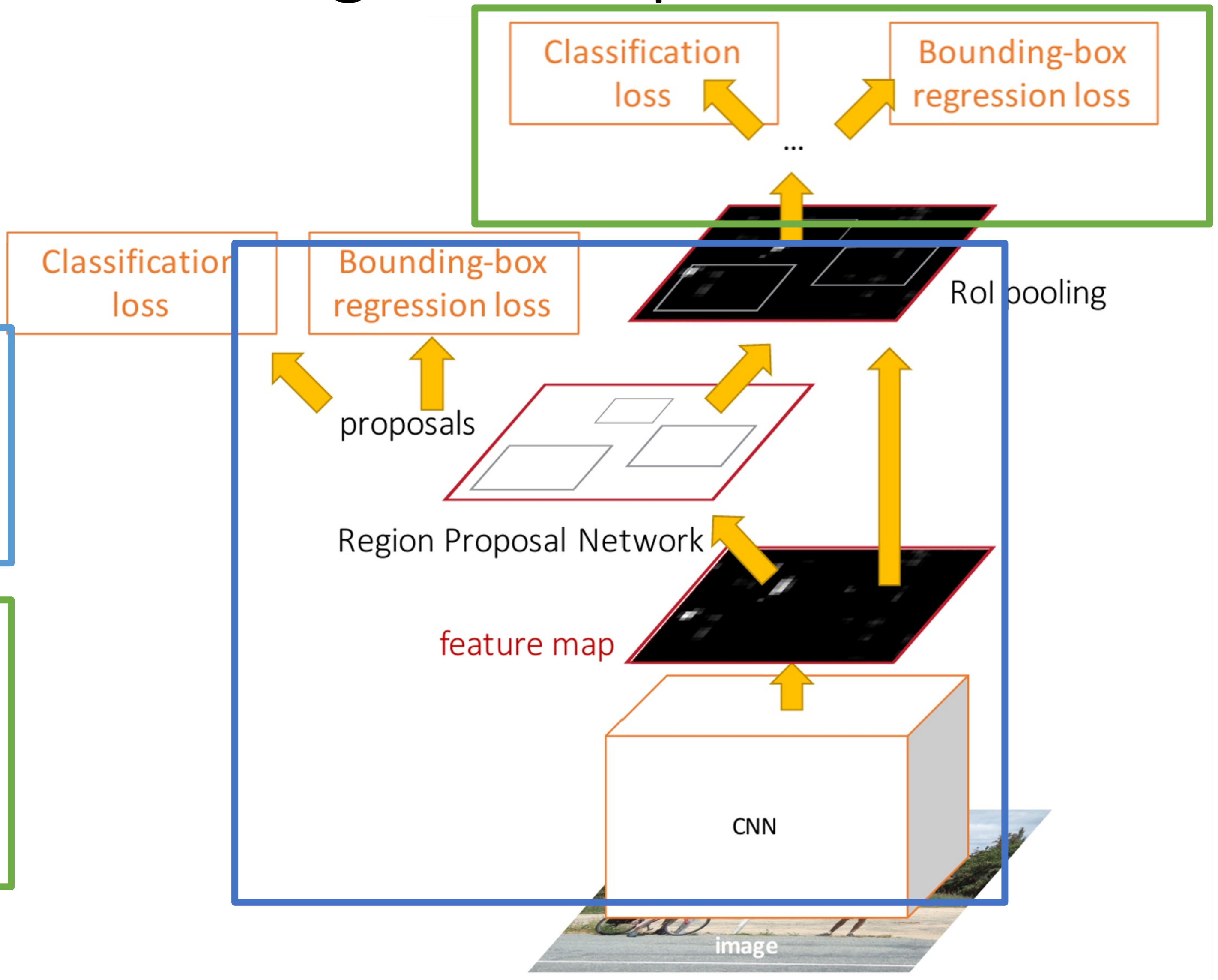
Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015

Faster R-CNN: Learnable Region Proposals

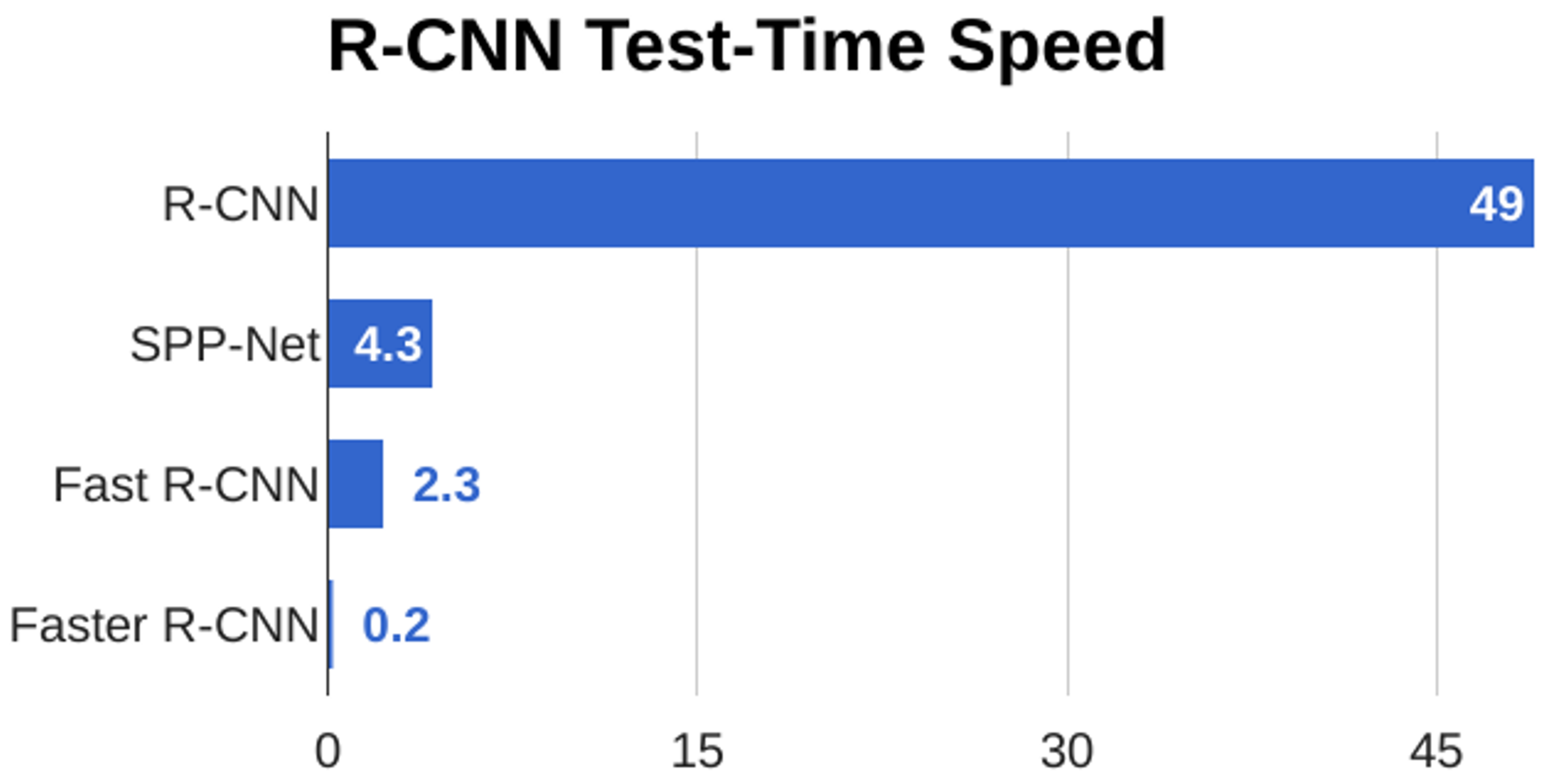
Faster R-CNN is a
Two-stage object detector

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: Learnable Region Proposals



Today's class

- How do we measure Object Detection accuracy?
- Naïve approaches & R-CNN
- Fast R-CNN
- Region Proposal Network & Faster R-CNN
- Advanced topics:
  - Feature Pyramid Networks to detect at scales
  - Single Shot detection

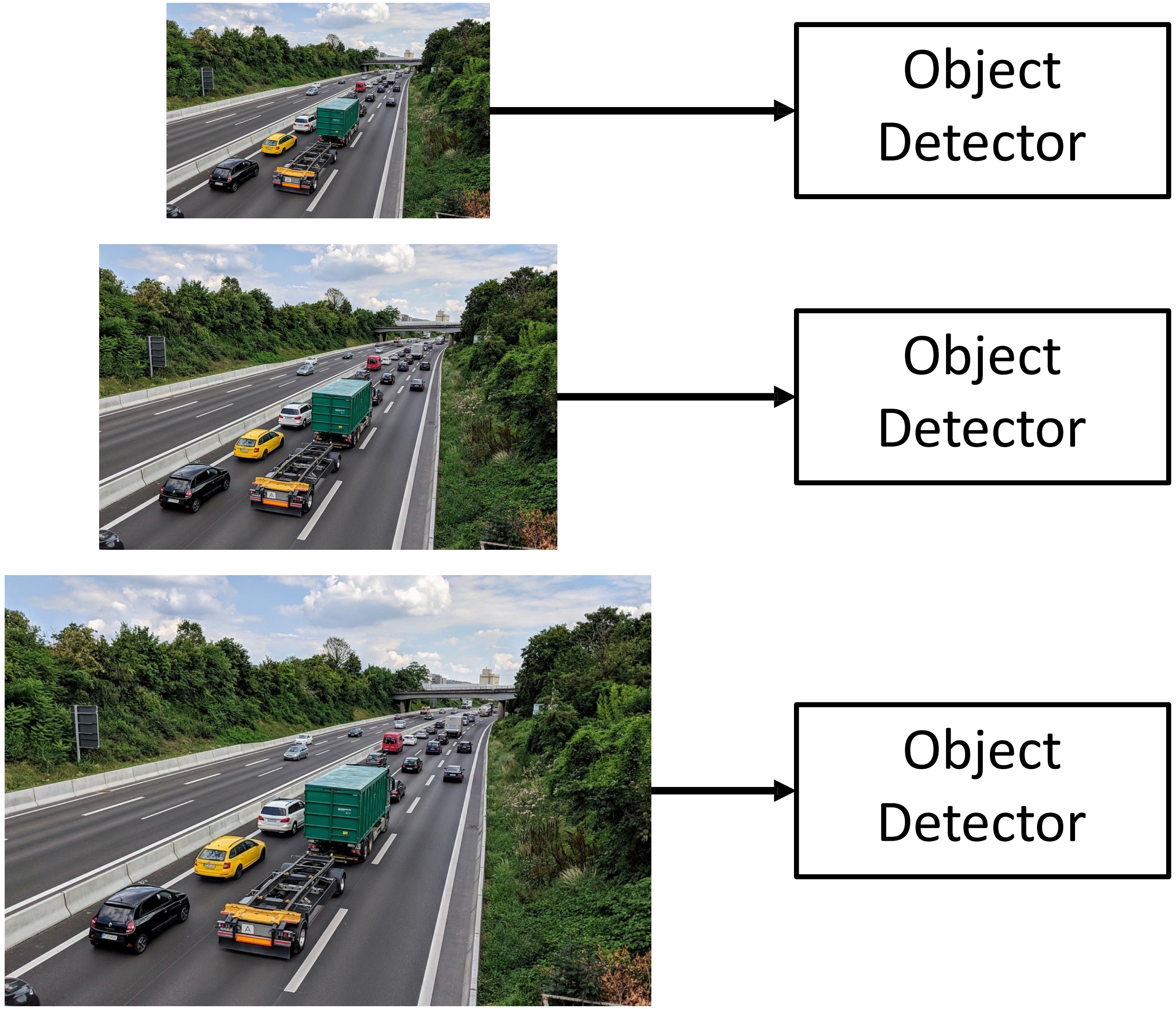
Dealing with Scale

We need to detect objects of many different scales.
How to improve ***scale invariance*** of the detector?

Dealing with Scale: Image Pyramid

Classic idea: build an *image pyramid* by resizing the image to different scales, then process each image scale independently.

Problem: Expensive! Don't share any computation between scales

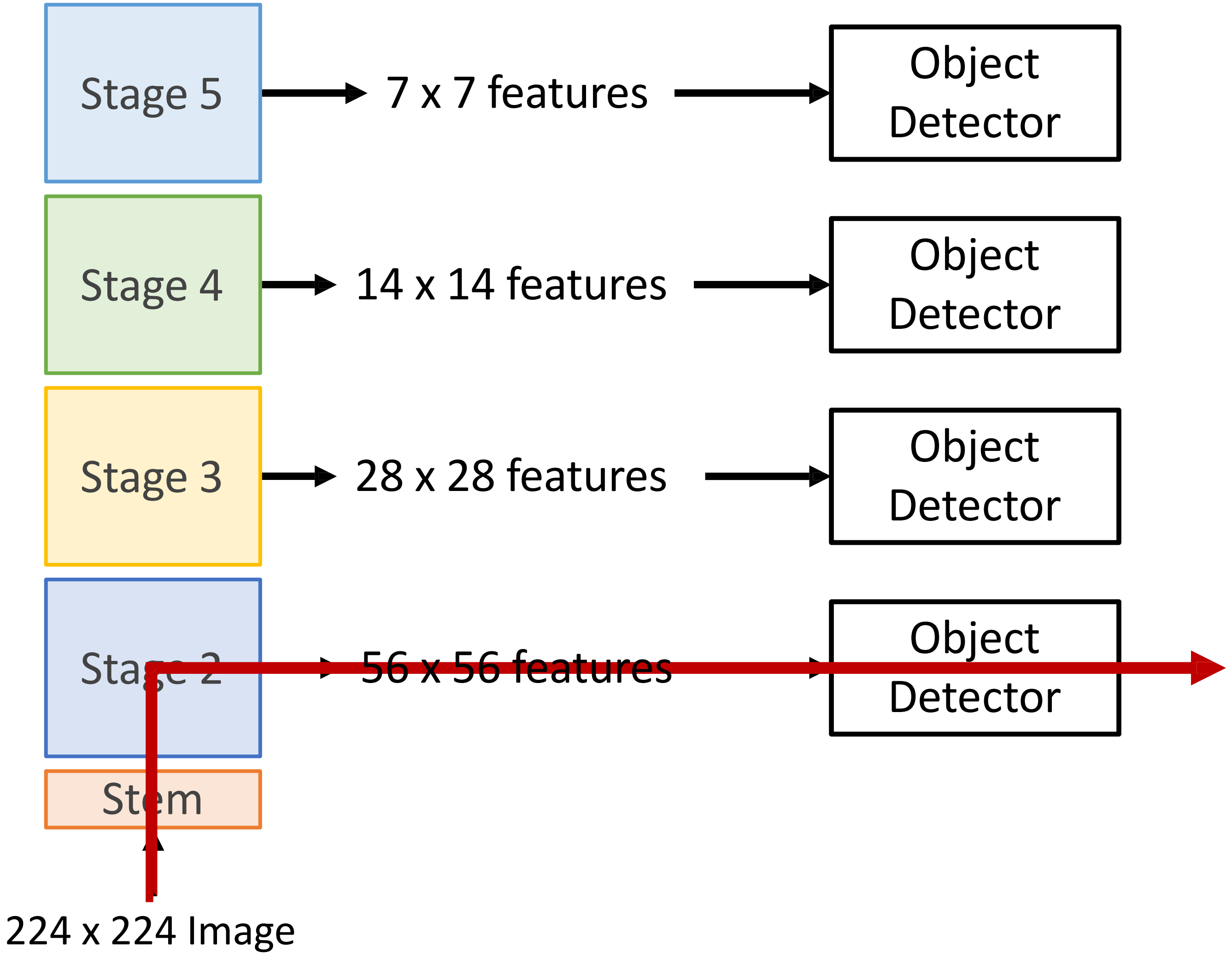


Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

Dealing with Scale: Multiscale Features

CNNs have multiple *stages* that operate at different resolutions. Attach an independent detector to the features at each level

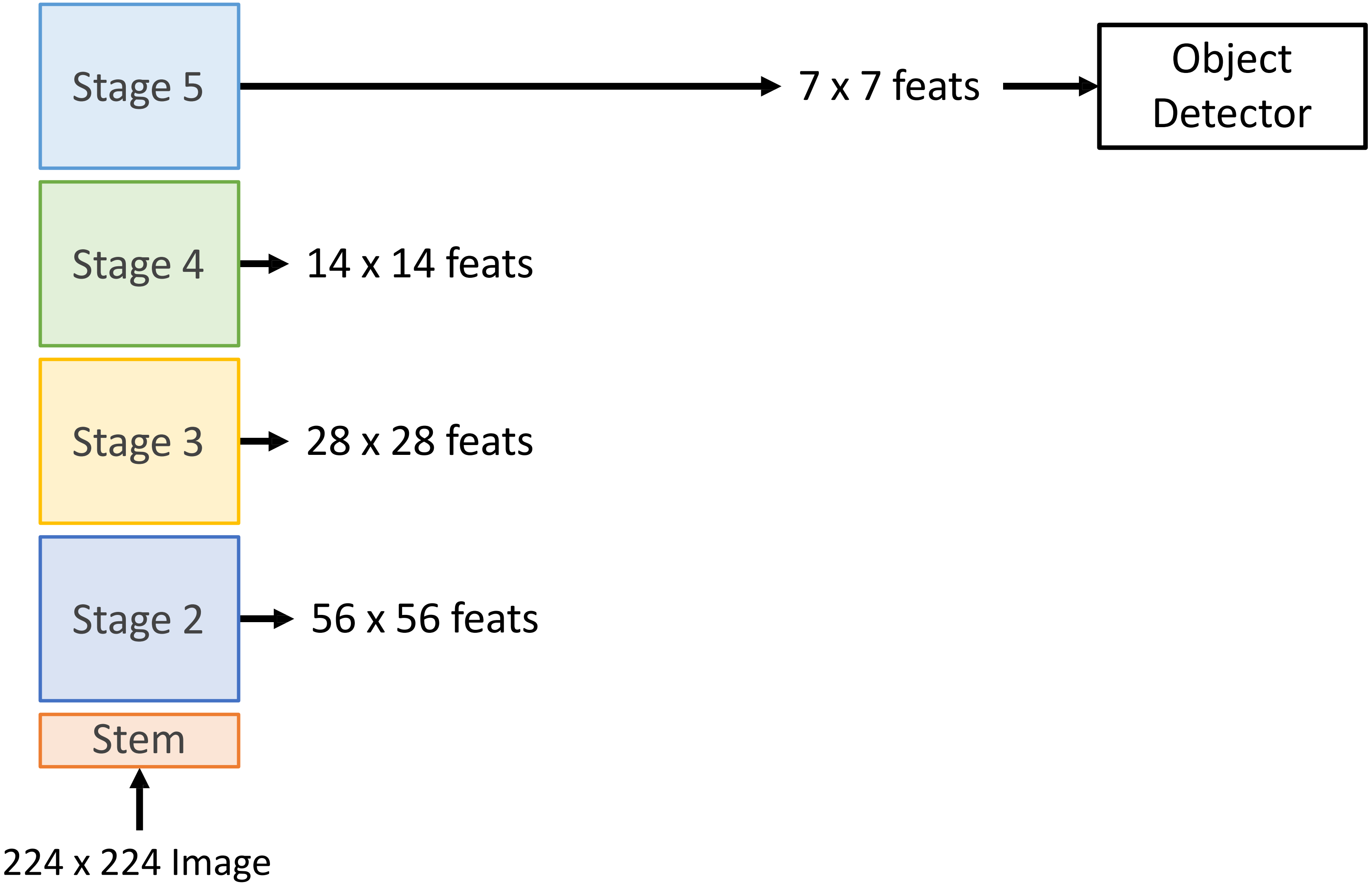
Problem: detector on early features doesn't make use of the entire backbone; doesn't get access to high-level features



Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

Dealing with Scale: Feature Pyramid Network

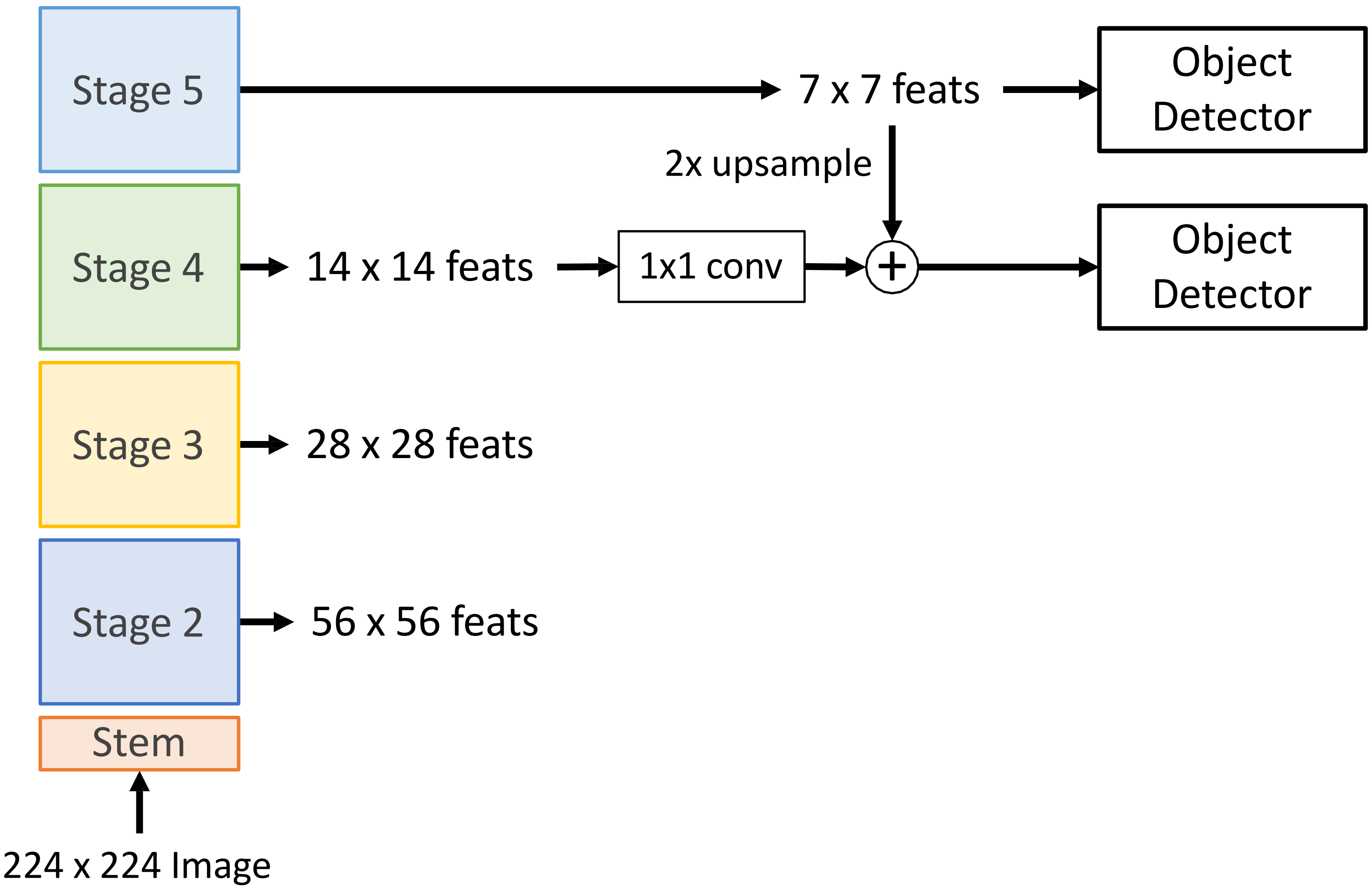
Add *top down connections* that feed information from high level features back down to lower level features



Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

Dealing with Scale: Feature Pyramid Network

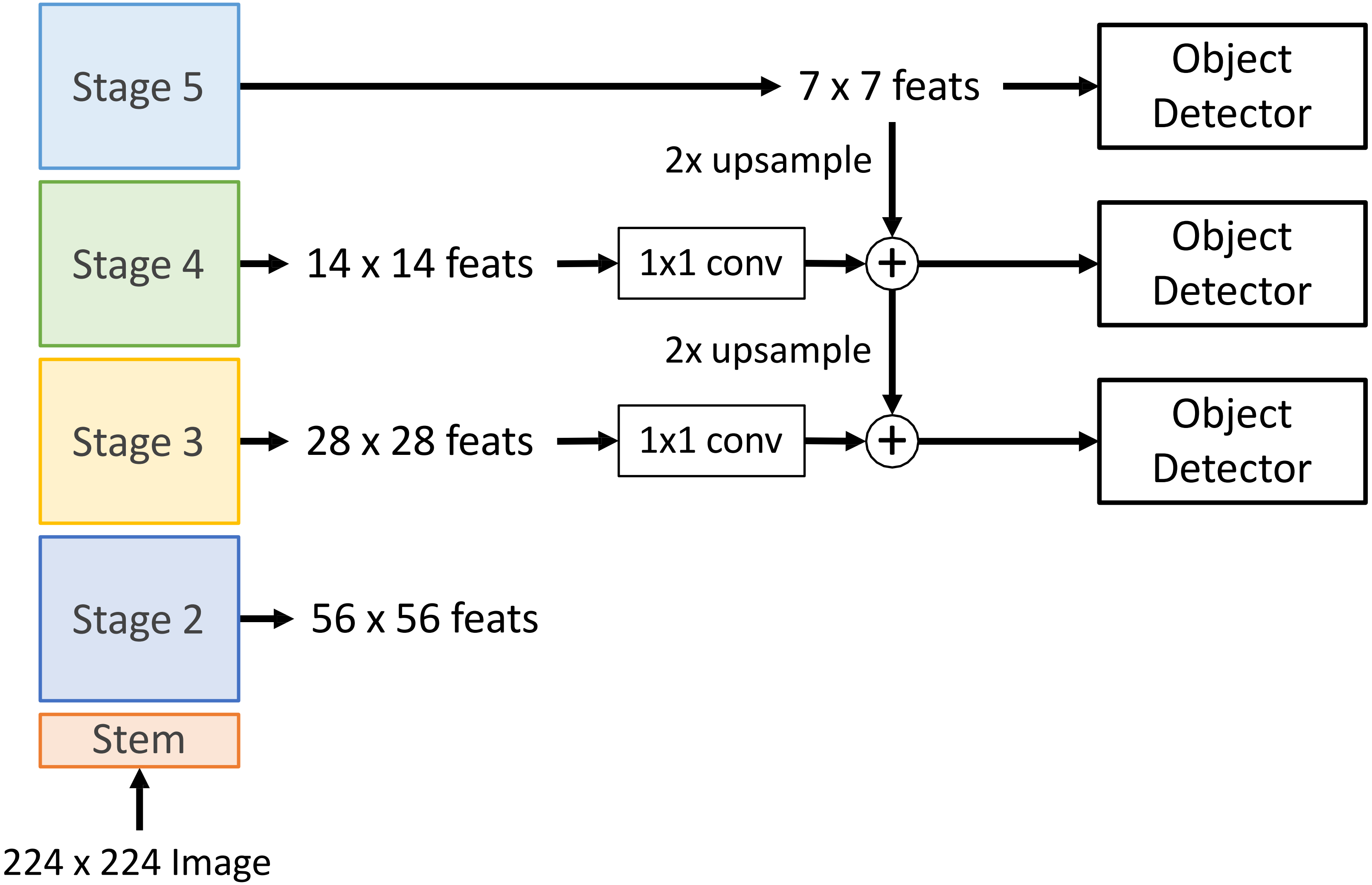
Add *top down connections* that feed information from high level features back down to lower level features



Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

Dealing with Scale: Feature Pyramid Network

Add *top down connections* that feed information from high level features back down to lower level features

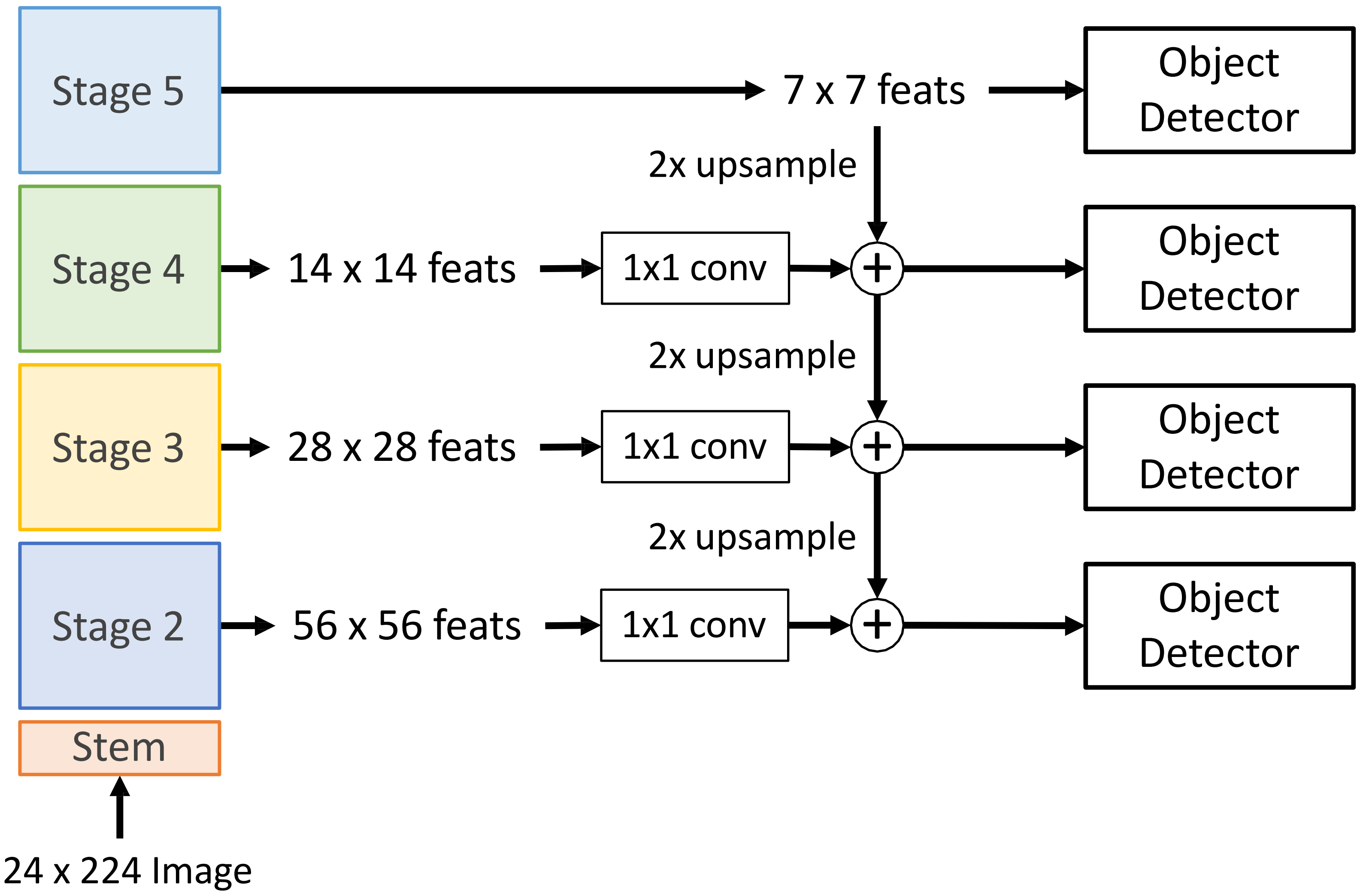


Lin et al, “Feature Pyramid Networks for Object Detection”, ICCV 2017

Dealing with Scale: Feature Pyramid Network

Add *top down connections* that feed information from high level features back down to lower level features

Efficient multiscale features where all levels benefit from the whole backbone! Widely used in practice



Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

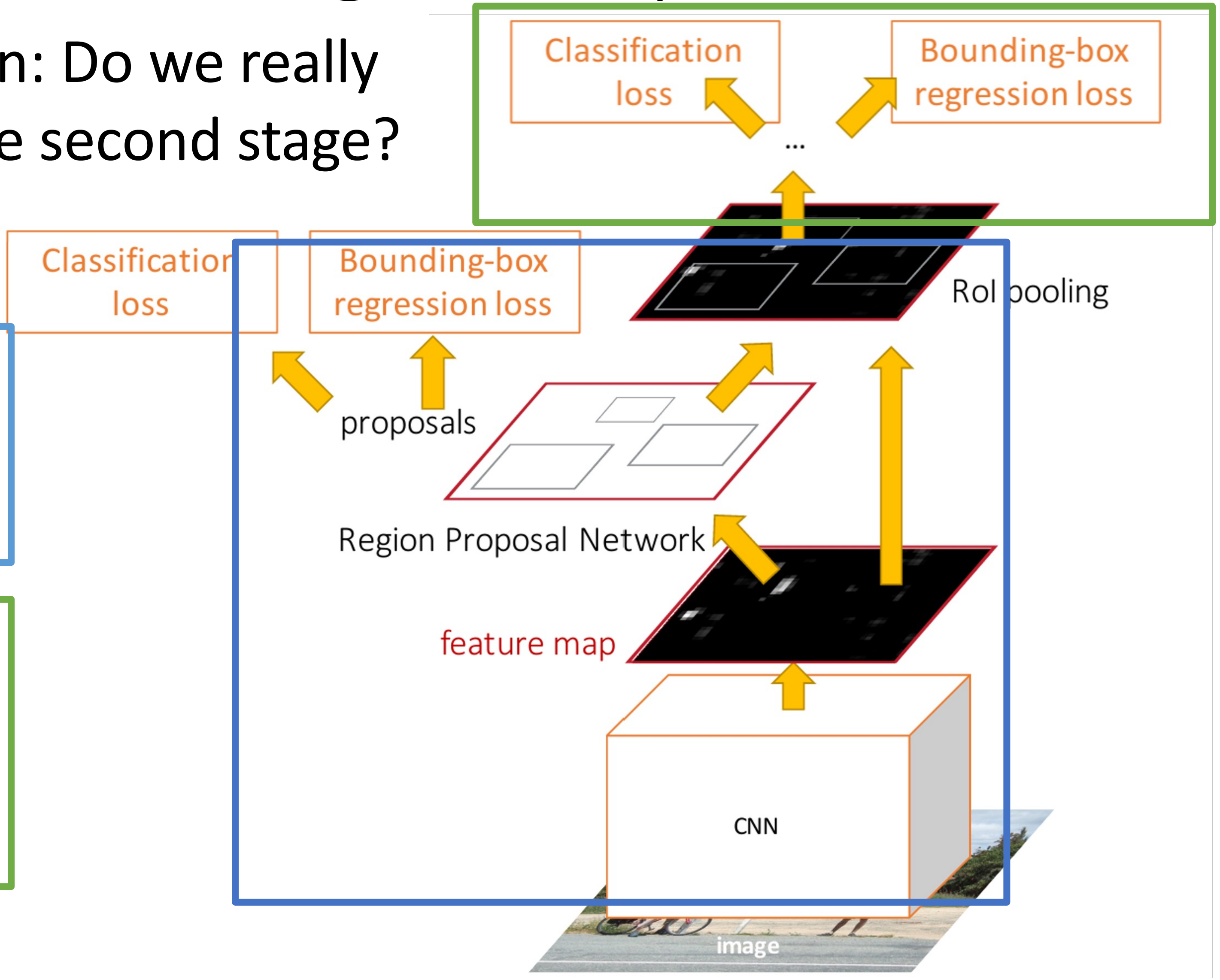
Faster R-CNN: Learnable Region Proposals

Question: Do we really need the second stage?

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

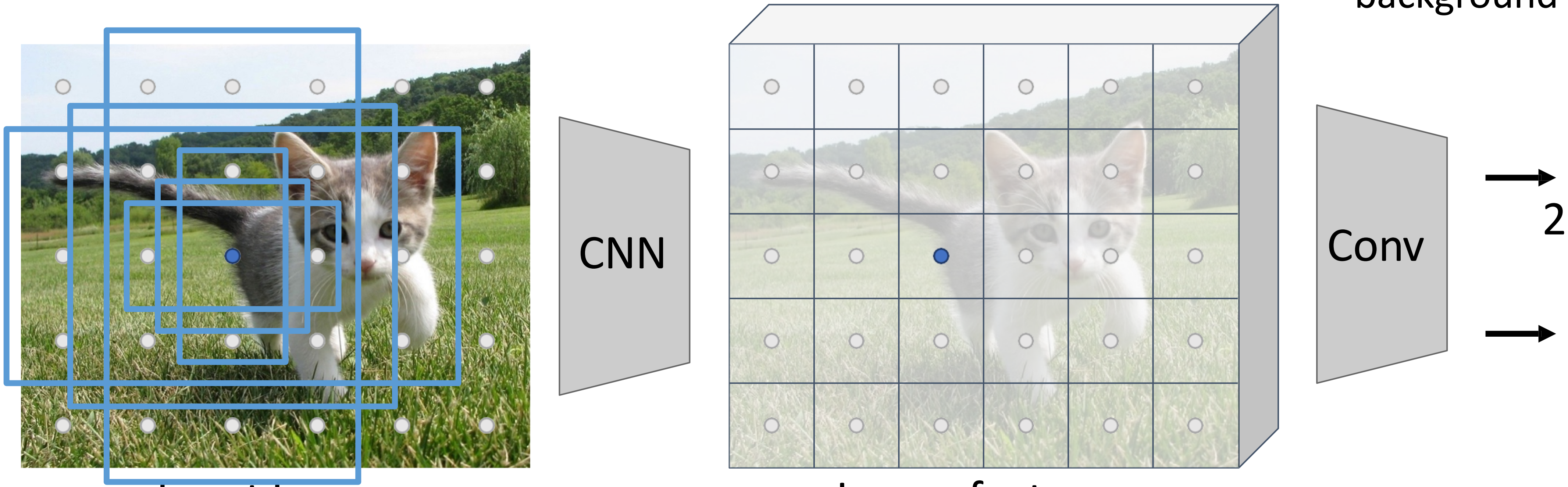


Single-Stage Detectors: RetinaNet

Similar to RPN – but rather than classify anchors as object/no object, directly predict object category (among C categories) or background

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Anchor classification
2K*(C+1) x 5 x 6

Anchor transforms
4K x 5 x 6

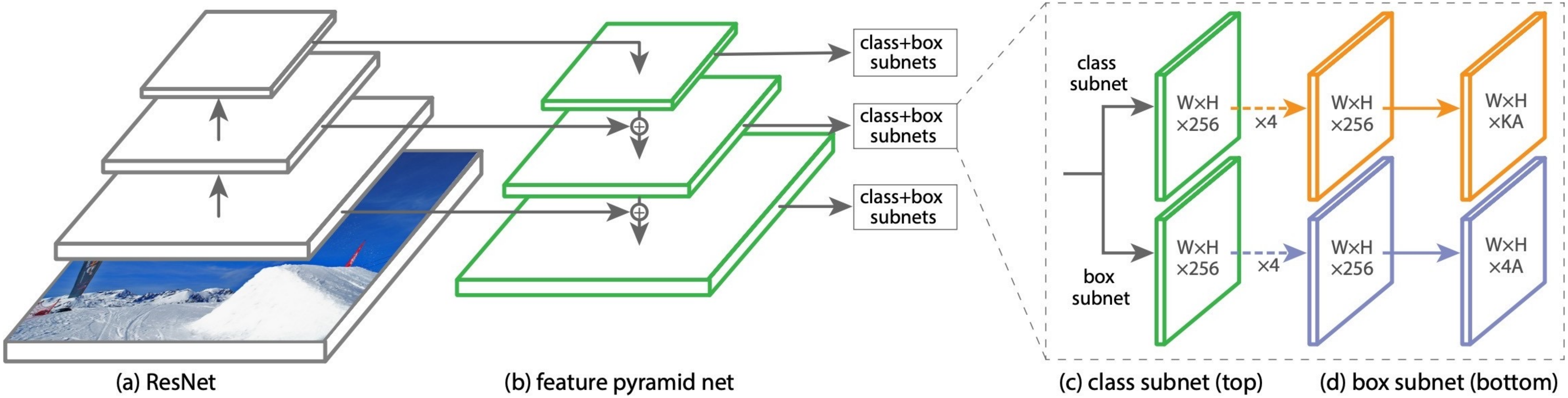
Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

Lin et al, "Focal Loss for Dense Object Detection", ICCV 2017

Single-Stage Detectors: RetinaNet

In practice, RetinaNet also uses Feature Pyramid Network to handle multiscale

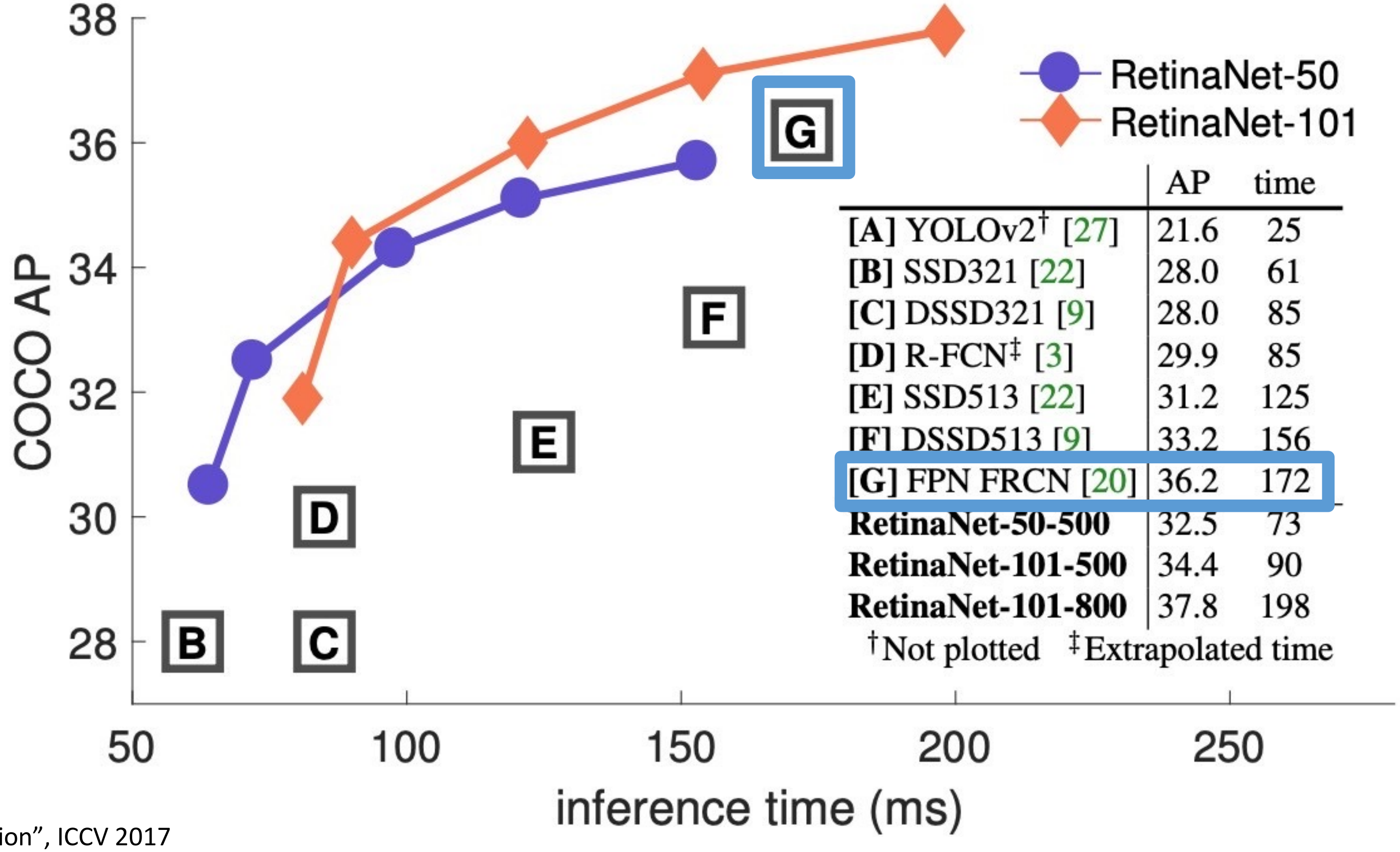


Lin et al, "Focal Loss for Dense Object Detection", ICCV 2017

Figure credit: Lin et al, ICCV 2017

Single-Stage Detectors: RetinaNet

Single-Stage detectors can be much faster than two-stage detectors

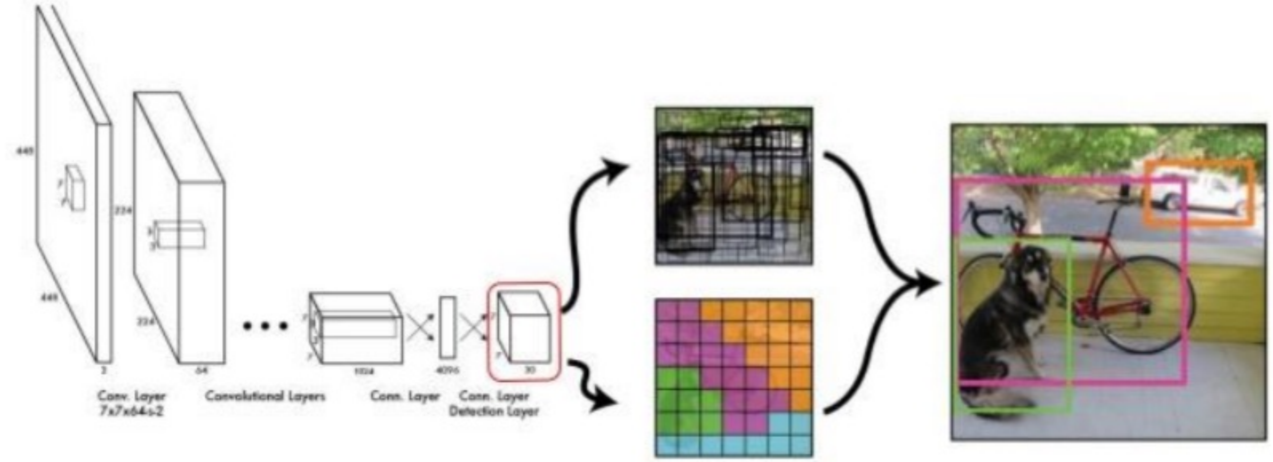


Faster R-CNN with Feature Pyramid Network

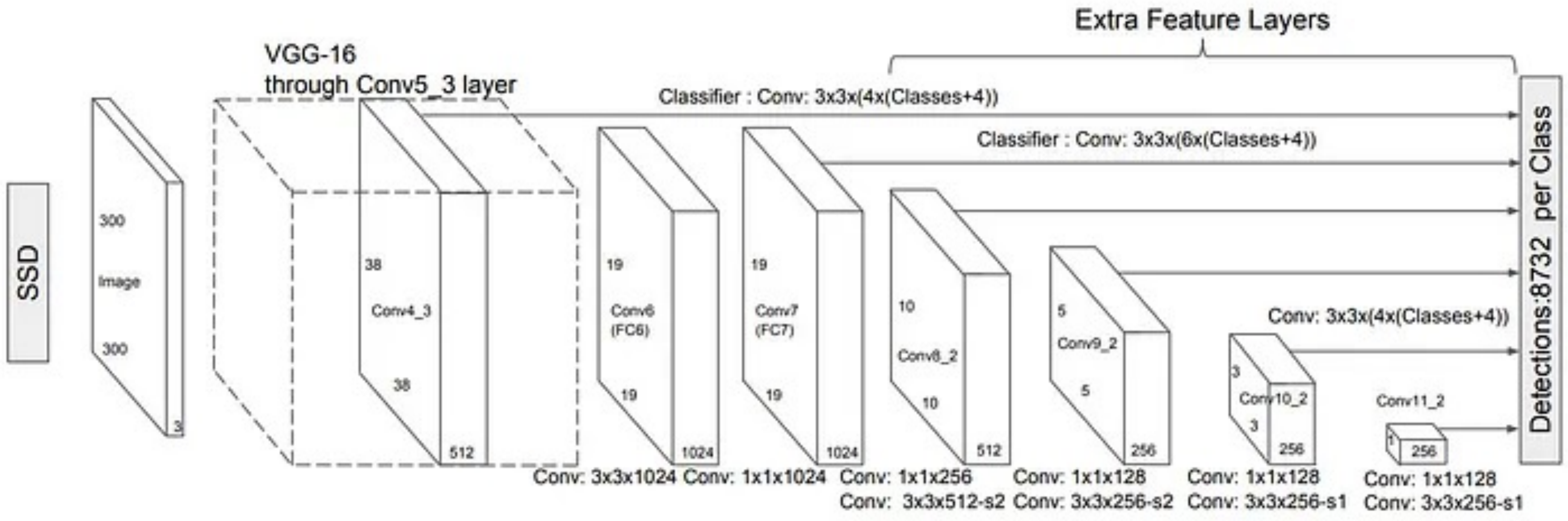
Lin et al, "Focal Loss for Dense Object Detection", ICCV 2017

Figure credit: Lin et al, ICCV 2017

Other popular Single-Stage Detectors



YOLO (You Only Look Once) – v1 to v7 versions

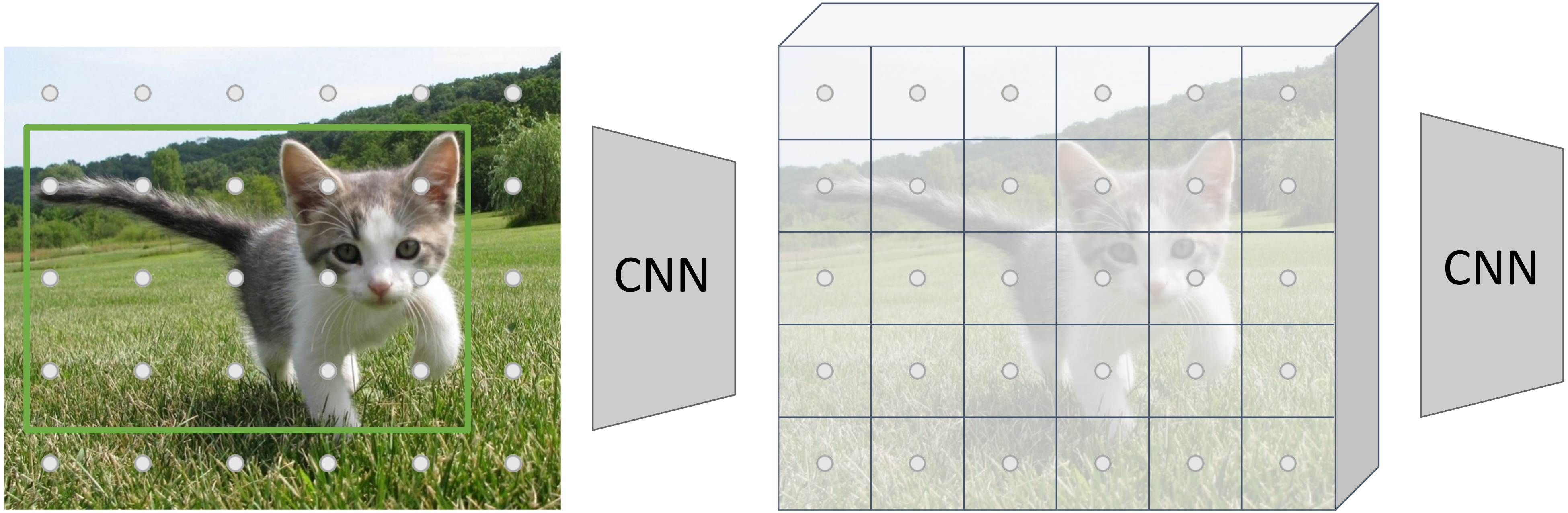


SSD Multibox detector, developed at UNC

Single-Stage Detectors: FCOS ("Anchor-free" detector)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

Tian et al, "FCOS: Fully Convolutional One-Stage Object Detection", ICCV 2019

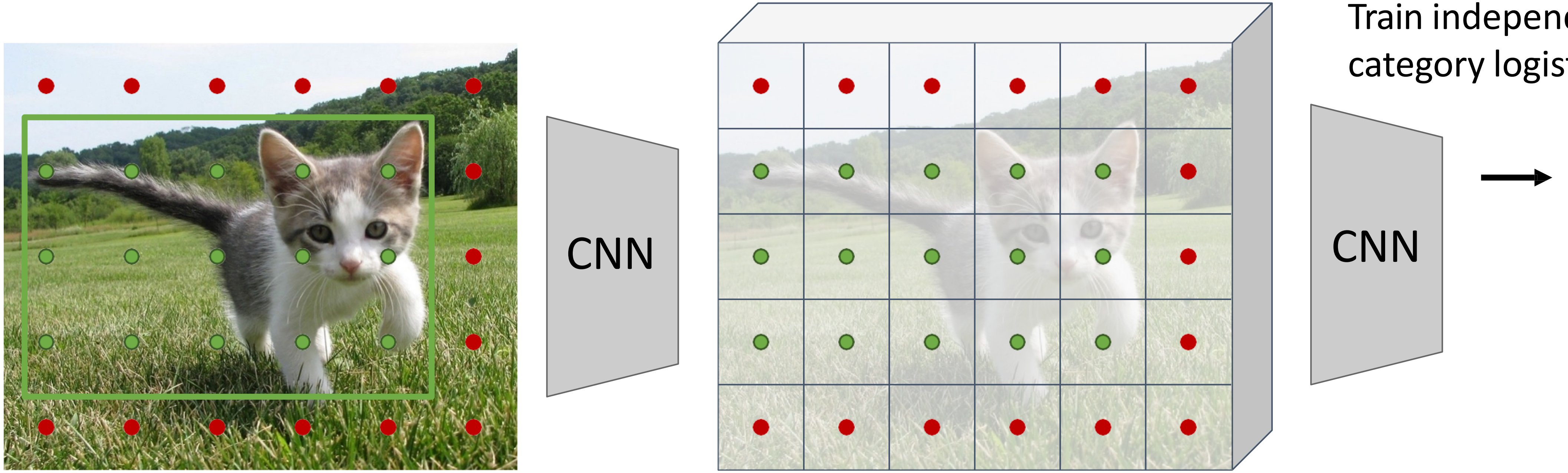
Single-Stage Detectors: FCOS ("Anchor-free" detector)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input

Classify points as positive if they fall into a GT box, or negative if they don't

Train independent per-category logistic regressors



Class scores
C x 5 x 6

Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

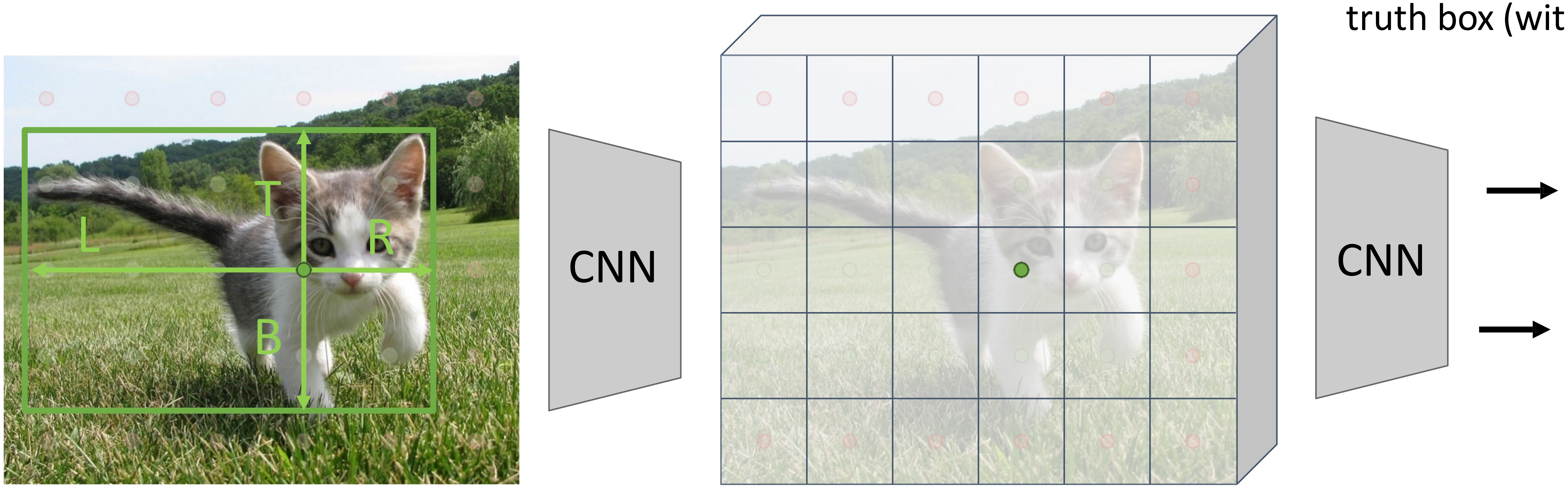
Tian et al, "FCOS: Fully Convolutional One-Stage Object Detection", ICCV 2019

Single-Stage Detectors: FCOS ("Anchor-free" detector)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input

For positive points, also regress distance to left, right, top, and bottom of ground-truth box (with L2 loss)



Class scores
C x 5 x 6

Box edges
4 x 5 x 6

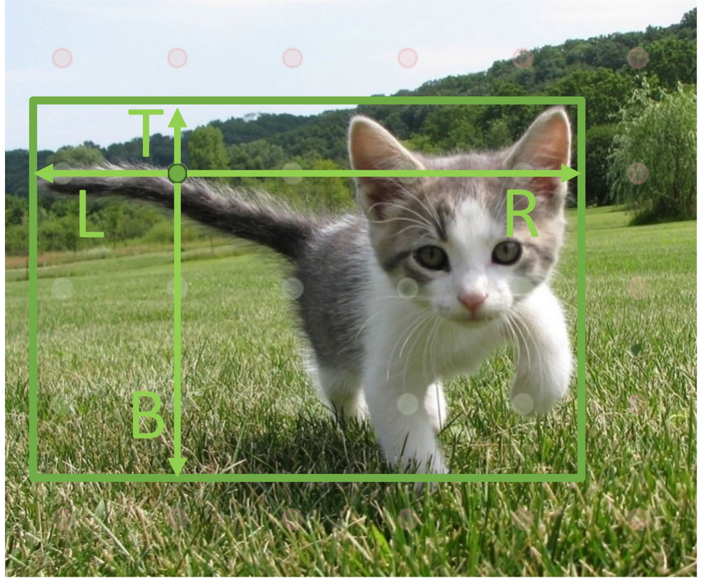
Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 5 x 6)

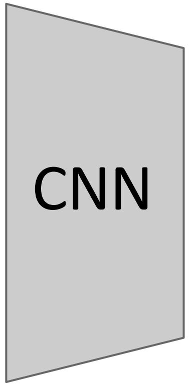
Tian et al, "FCOS: Fully Convolutional One-Stage Object Detection", ICCV 2019

Single-Stage Detectors: FCOS ("Anchor-free" detector)

Run backbone CNN to get features aligned to input image



Input Image
(e.g. 3 x 640 x 480)



Each feature corresponds to a point in the input

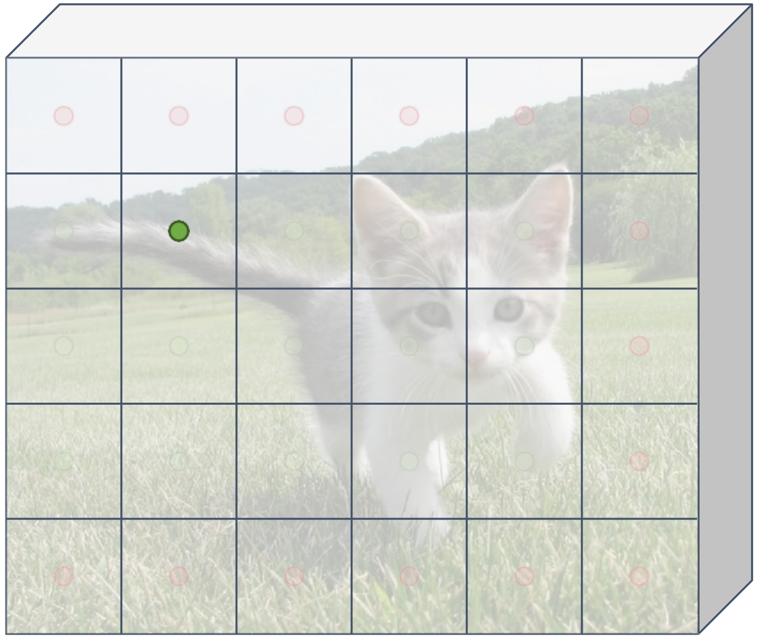
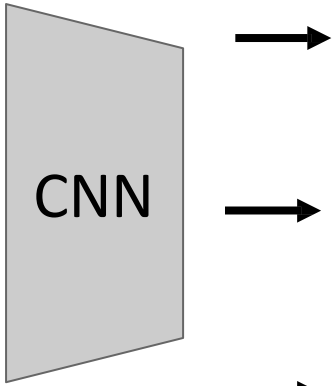


Image features
(e.g. 512 x 5 x 6)

Finally, predict "centerness" for all positive points (using logistic regression loss)



Class scores
C x 5 x 6

Box edges
4 x 5 x 6

Centerness
1 x 5 x 6

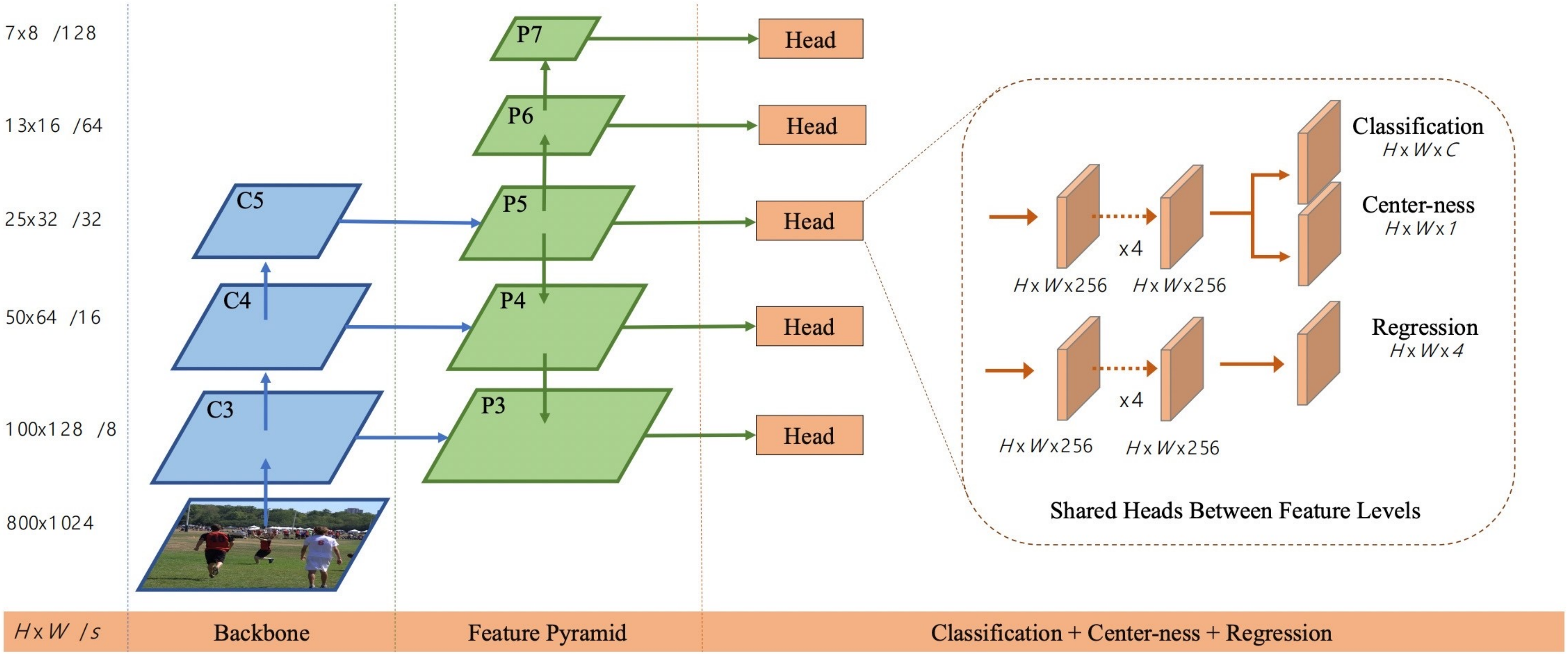
$$centerness = \sqrt{\frac{\min(L,R)}{\max(L,R)} \cdot \frac{\min(T,B)}{\max(T,B)}}$$

Ranges from 1 at box center to 0 at box edge

Test-time: predicted "confidence" for the box from each point is product of its class score and centerness.

Single-Stage Detectors: FCOS ("Anchor-free" detector)

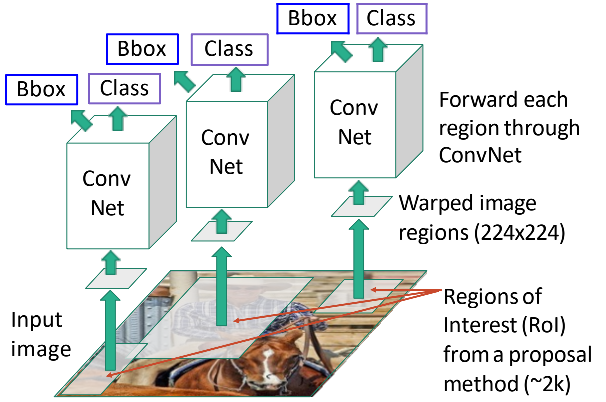
FCOS also uses a Feature Pyramid Network with heads shared across stages



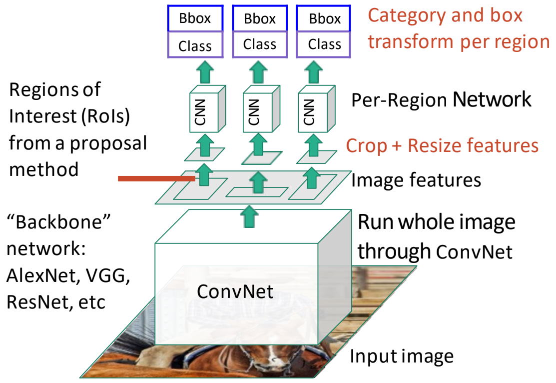
Tian et al, "FCOS: Fully Convolutional One-Stage Object Detection", ICCV 2019

Summary

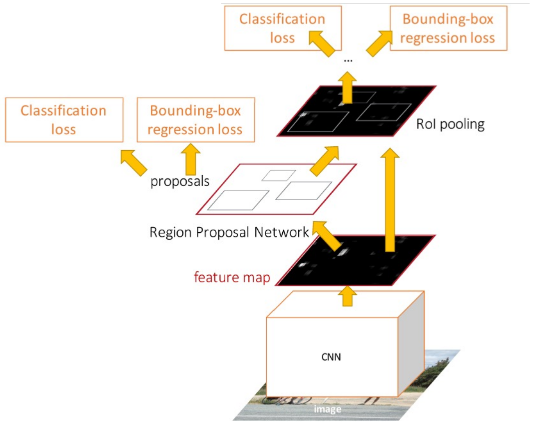
"Slow" R-CNN: Run CNN independently for each region



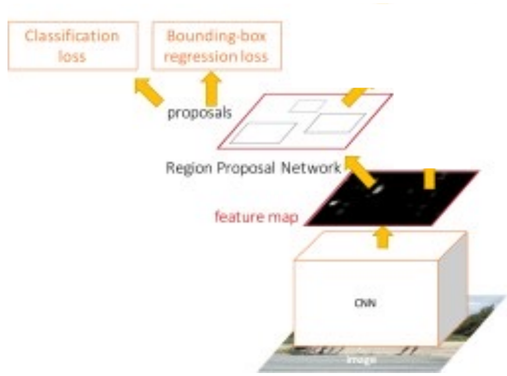
Fast R-CNN: Apply differentiable cropping to shared image features



Faster R-CNN: Compute proposals with CNN



Single-Stage: Fully convolutional detector

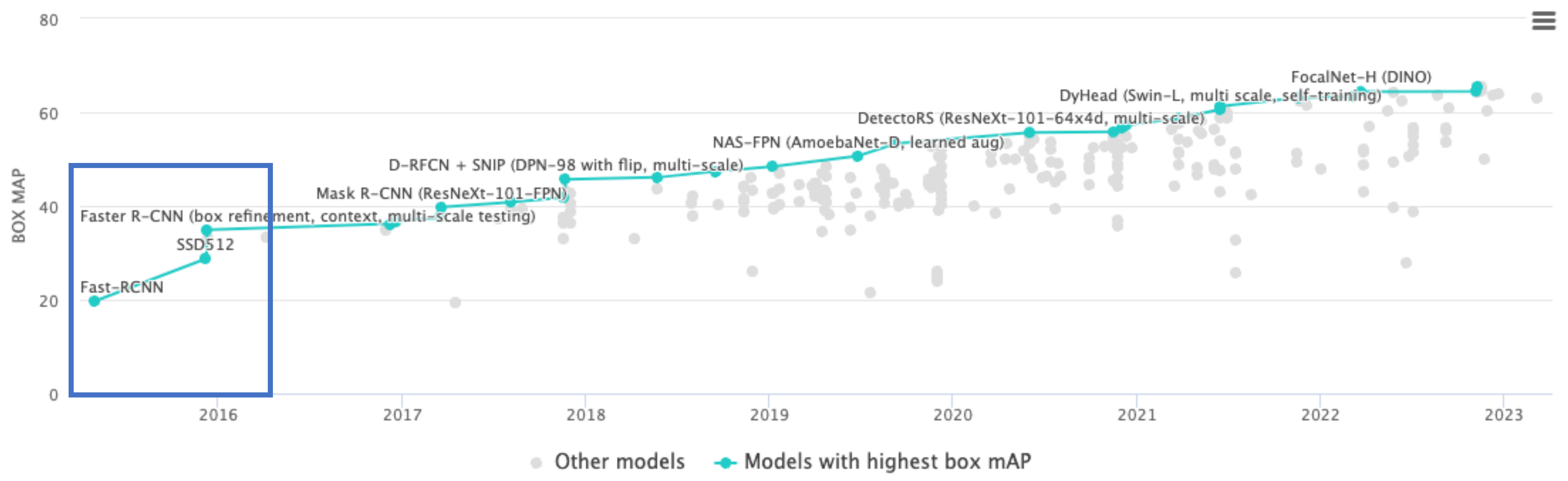


With anchors: RetinaNet
Anchor-Free: FCOS

Object Detection on COCO test-dev

Leaderboard Dataset

View box mAP by Date for All models



Slide Credits

- **EECS 442/498 Computer Vision, by Justin Johnson & David Fouhey, U Michigan.**