Object Detection

Visual Understanding

What you will learn in this course

Low-Level Vision

Image Processing:
Photometric Image formation
Point Operations
Histogram Equalization
Linear & Non-Linear filters
Convolution & Cross correlation

Feature Detection:
Edge detection
Corner detection
Line detection

Feature Description:
SIFT
Feature Matching
Image Stitching
Panorama

Geometric Vision

Transformation and Camera Mode:
Geometric Image Formation
Pinhole Camera Model

Camera Calibration:
Estimating intrinsics
Estimating extrinsics

Visual Understanding

Object Detection & Tracking:
Single-stage detectors
Two-stage detectors
YOLO
Object tracking

Image Segmentation:
Traditional Image Segmentation
Learning-based Image Segmentation

Machine Learning for CV

Machine Learning for Classification:
k-NN
Linear Classifier

Deep Learning:
Optimization
Neural Network
CNN
CNN architectures:

Generative & Geometric Learning

Generative Models:
AE, VAE, GAN

3D Vision and 3D Deep Learning:
3D shape representations
Geometric deep learning overview
3D Deep Learning models : pointnet

What we will learn today

- Region-based CNN (R-CNN)
- Cropping Features
  - ROI Pool
  - ROI Align
- Fast R-CNN
- IoU and mAP
- Faster R-CNN
- Single-Stage Object Detection (SSD)
- YOLOv1

So far: Image Classification

This image is CC0 public domain

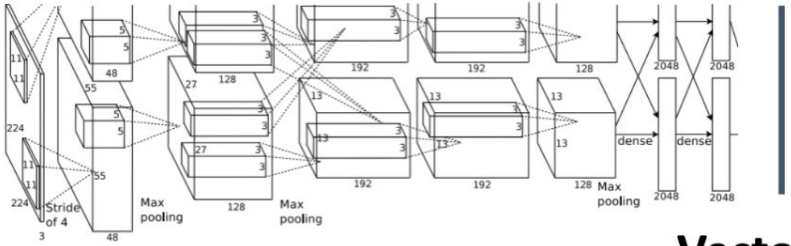


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector:
4096

Fully-Connected:
4096 to 1000

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

Convolution Layers

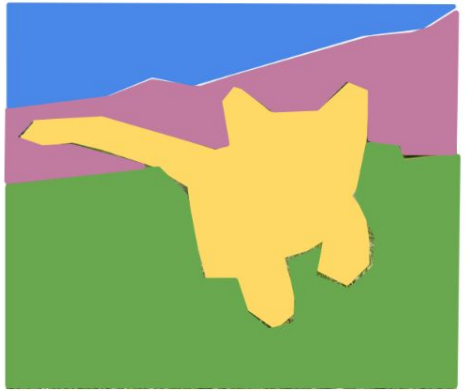
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Computer Vision Tasks

Classification

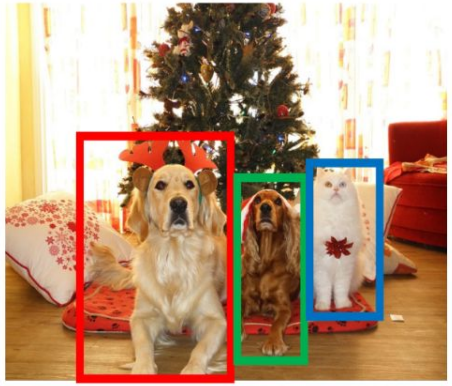
CAT

Semantic Segmentation



GRASS, CAT, TREE, SKY

Object Detection



DOG, DOG, CAT

Instance Segmentation

DOG, DOG, CAT

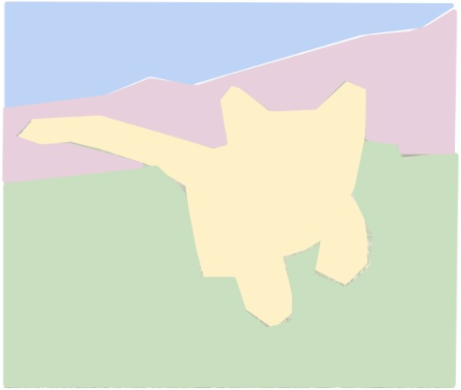
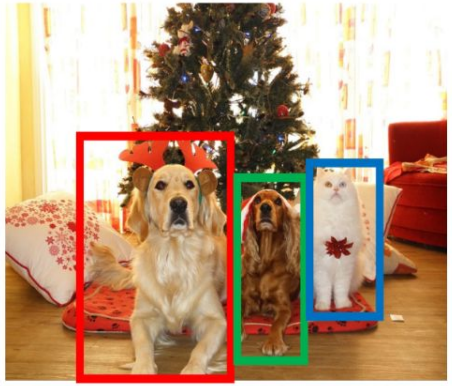
No spatial extent

No objects, just pixels

Multiple objects

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Computer Vision Tasks

Classification	Semantic Segmentation	Object Detection	Instance Segmentation
			
CAT	GRASS, CAT, TREE, SKY	DOG, DOG, CAT	DOG, DOG, CAT
No spatial extent	No objects, just pixels	Multiple objects	

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

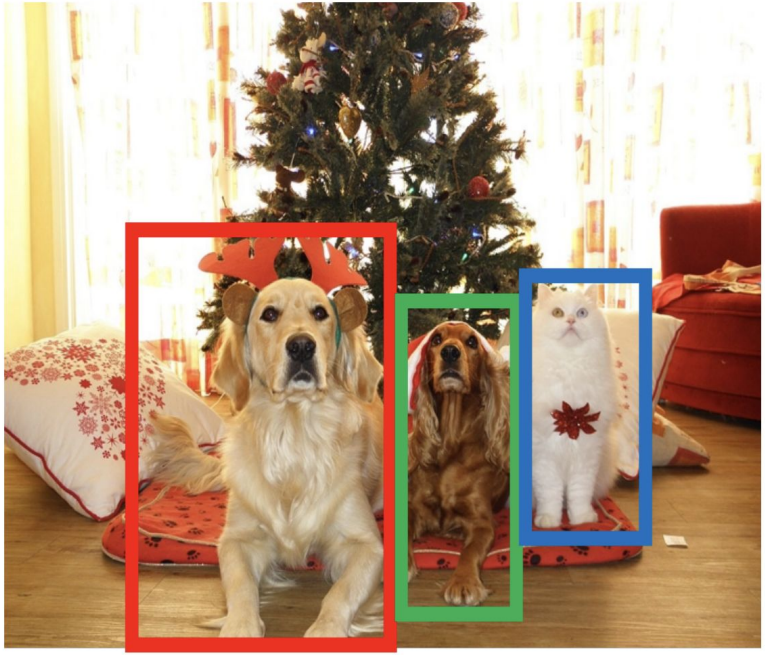
Task Definition

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)

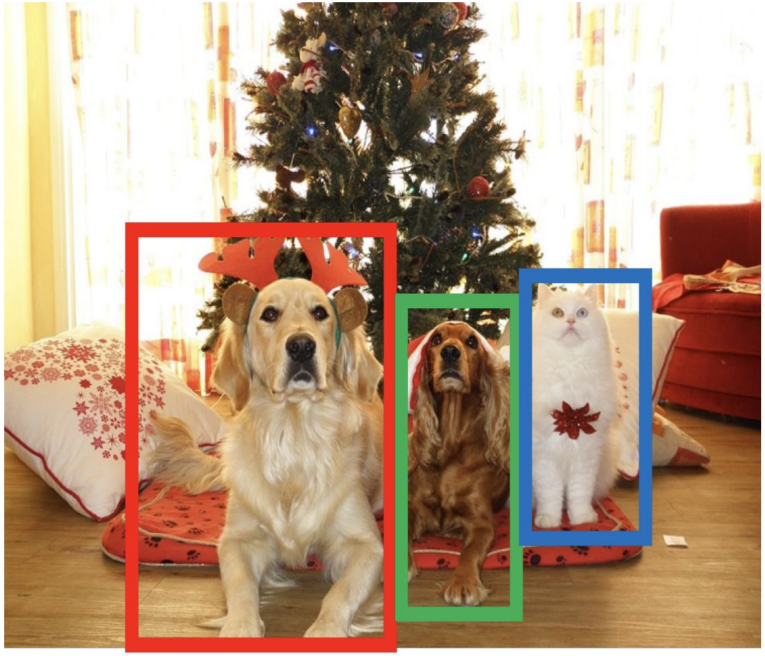


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Challenges

1. Multiple outputs: Need to output variable numbers of objects per image.
2. Multiple types of output: Need to predict "WHAT" (category label) as well as "WHERE" (bounding box)
3. Large images: Classification works at 224x224; need higher resolution for detection, often ~800x600

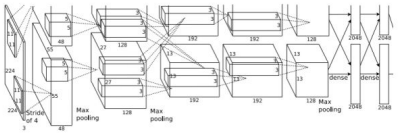


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting a single object

This image is CC0 public domain



Vector:
4096

Fully connected:
4096 to 1000

"WHAT"

Class Scores:
Cat: 0.9
Dog: 0.05
Cow: 0.05
...

Correct Label:
Cat

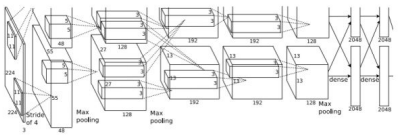
Softmax loss

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting a single object

This image is CC0 public domain



Vector:
4096

Fully connected:
4096 to 1000

Class Scores:
Cat: 0.9
Dog: 0.05
Cow: 0.05
...

"WHAT"

Correct Label:
Cat

Softmax loss

Fully connected:
4096 to **4**

Box coordinates:
(x, y, w, h)

"WHERE"

L2 loss

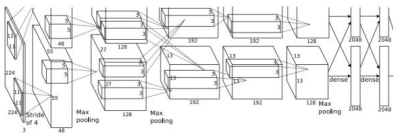
Correct box:
(x', y', w', h')

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting a single object

This image is CC0 public domain



Vector:
4096

Fully connected:
4096 to 1000

Fully connected:
4096 to **4**

"WHAT"

Class Scores:
Cat: 0.9
Dog: 0.05
Cow: 0.05
...

Softmax loss

Correct Label:
Cat

Weighted sum

Loss

Box coordinates:
(x, y, w, h)

L2 loss

Correct box:
(x', y', w', h')

"WHERE"

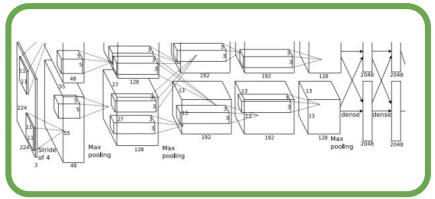
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting a single object

This image is CC0 public domain

Often pre-trained in ImageNet



Vector: 4096

Fully connected: 4096 to 1000

"WHAT"

Class Scores:
Cat: 0.9
Dog: 0.05
Cow: 0.05
...

Correct Label:
Cat

Softmax loss

Multitask Loss

Weighted sum → **Loss**

Fully connected: 4096 to **4**

Box coordinates:
(x, y, w, h)

"WHERE"

L2 loss

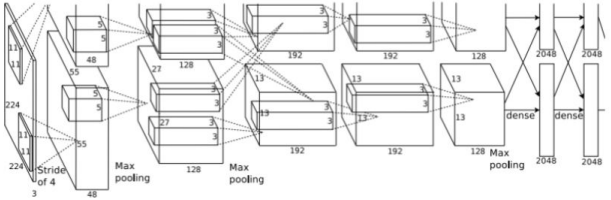
Correct box:
(x', y', w', h')

Problem: Images can have more than one object!

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

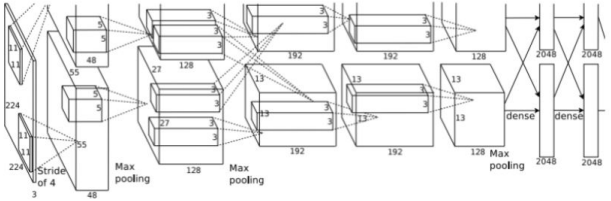
Object Detection

Need different numbers of outputs per image



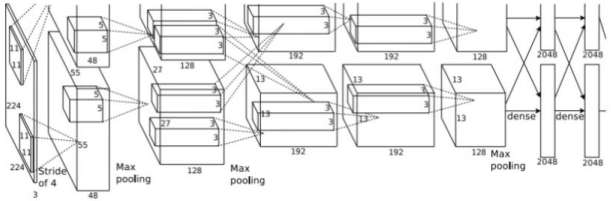
CAT: (x, y, w, h)

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)
....

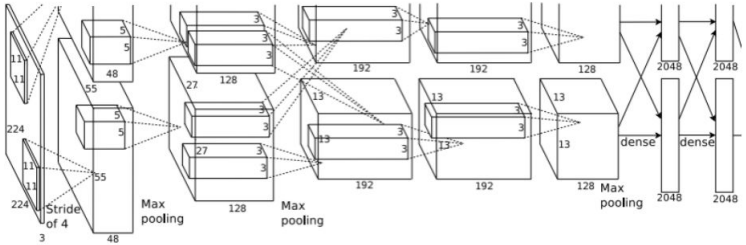
Many numbers

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting multiple objects: Sliding Window

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



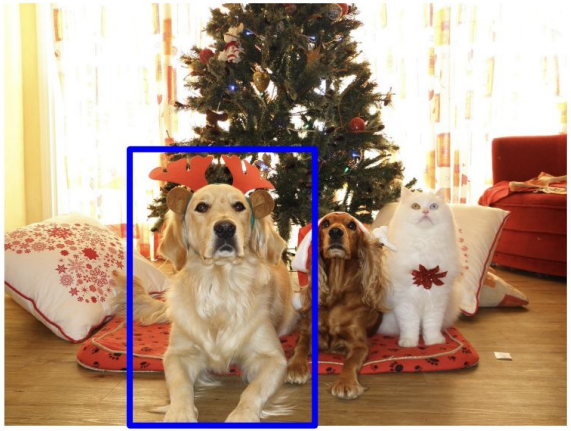
Dog? NO
Cat? NO
Background? YES

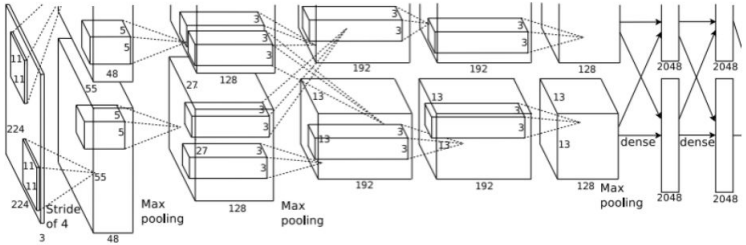
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting multiple objects: Sliding Window

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





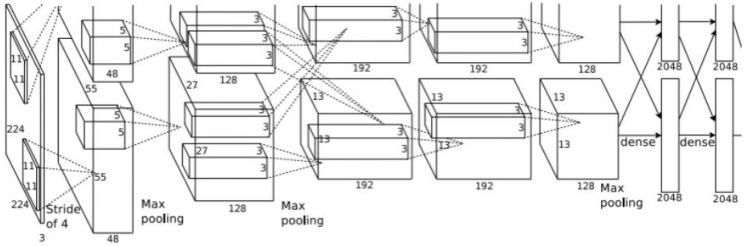
Dog? YES
Cat? NO
Background? NO

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting multiple objects: Sliding Window

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



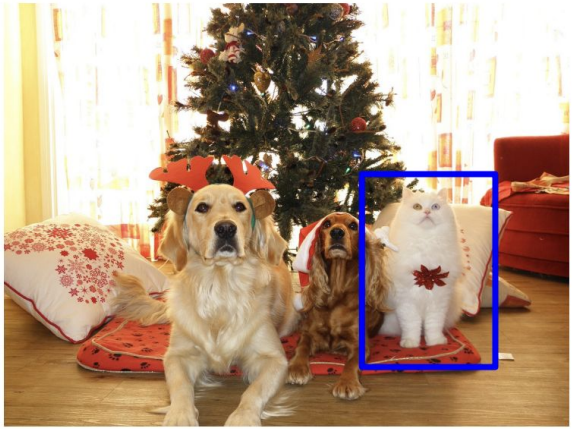
Dog? YES
Cat? NO
Background? NO

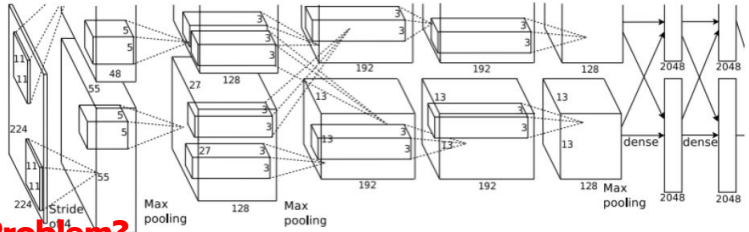
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting multiple objects: Sliding Window

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





Dog? NO
Cat? Yes
Background? NO

Problem?

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:

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

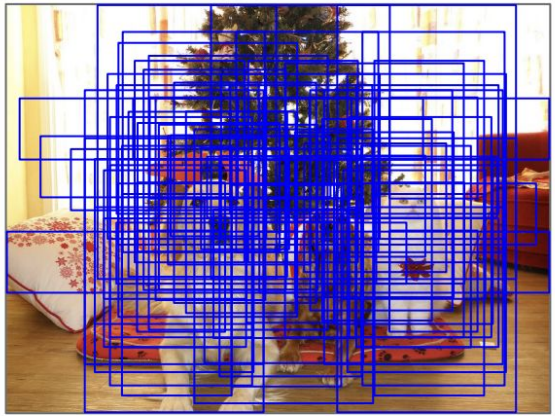
$$=\frac{H(H+1)}{2}\frac{W(W+1)}{2}$$

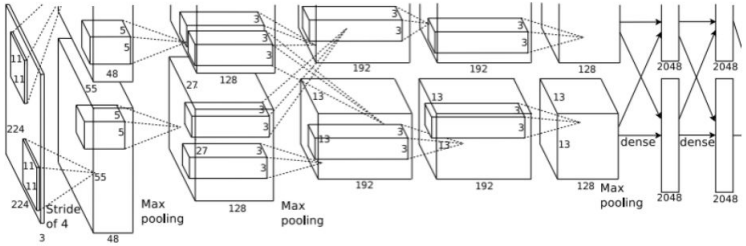
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Detecting multiple objects: Sliding Window

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





Dog? YES
Cat? NO
Background? NO

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

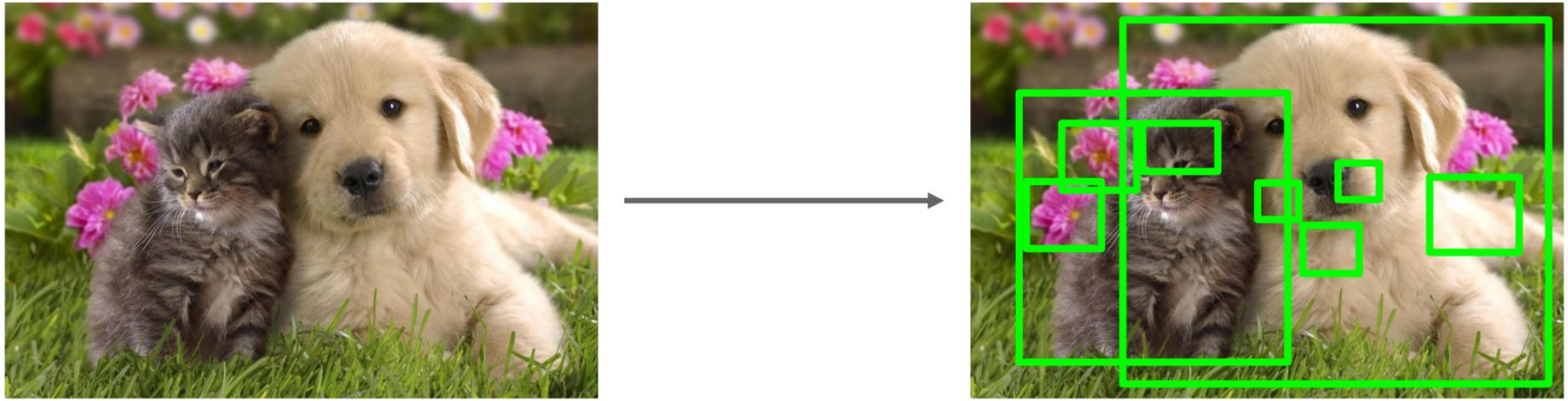
Need to apply CNN to huge number of locations, scales, and aspect ratios, very computationally expensive!

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Region Proposals: Selective Search

- Find "blobby" image regions that are likely to contain objects.
- Often based on heuristics: e.g. look for "blob-like" image regions.
- 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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

R-CNN: Region-Based CNN

Input image

Object Detection

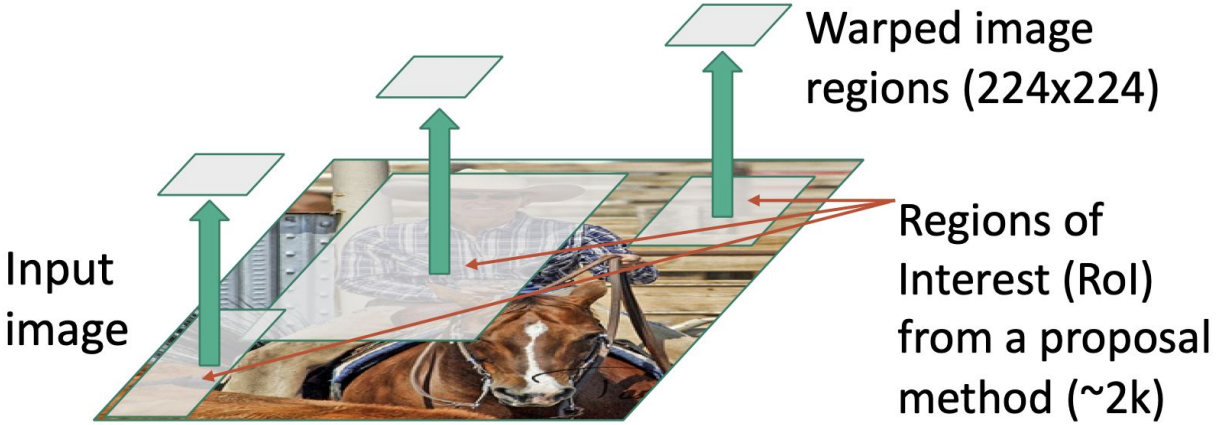
R-CNN: Region-Based CNN

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN

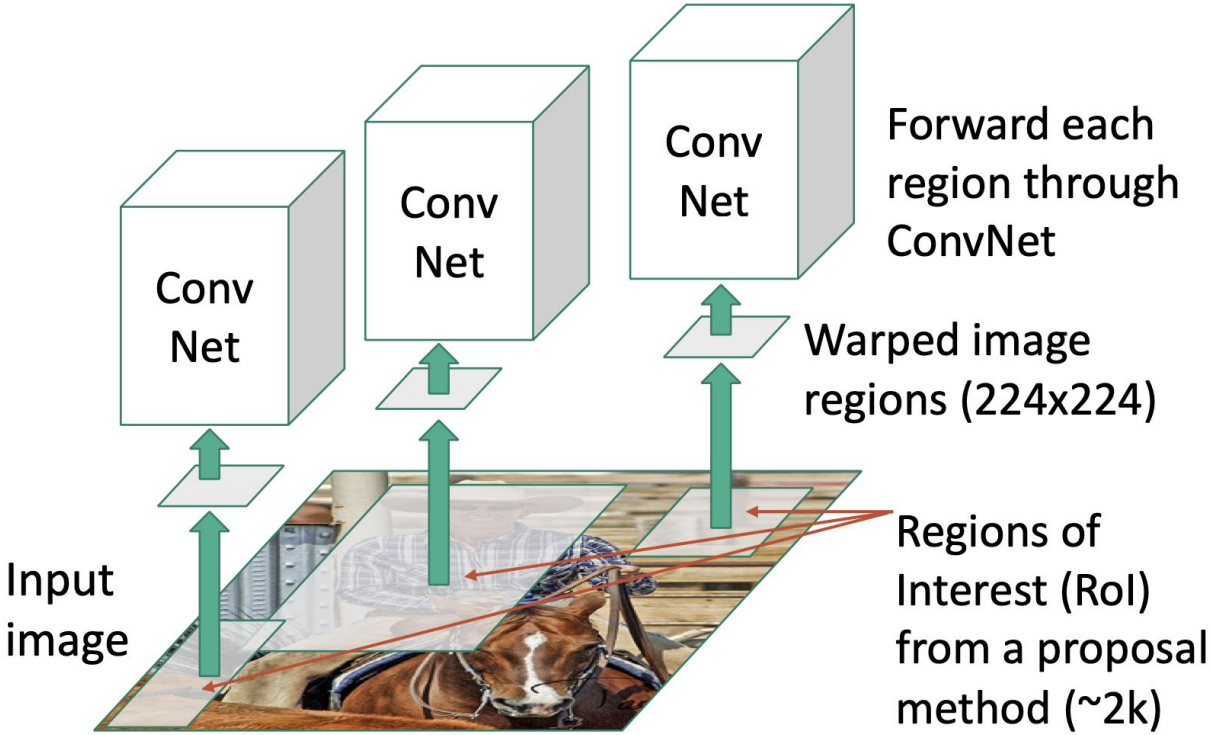


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

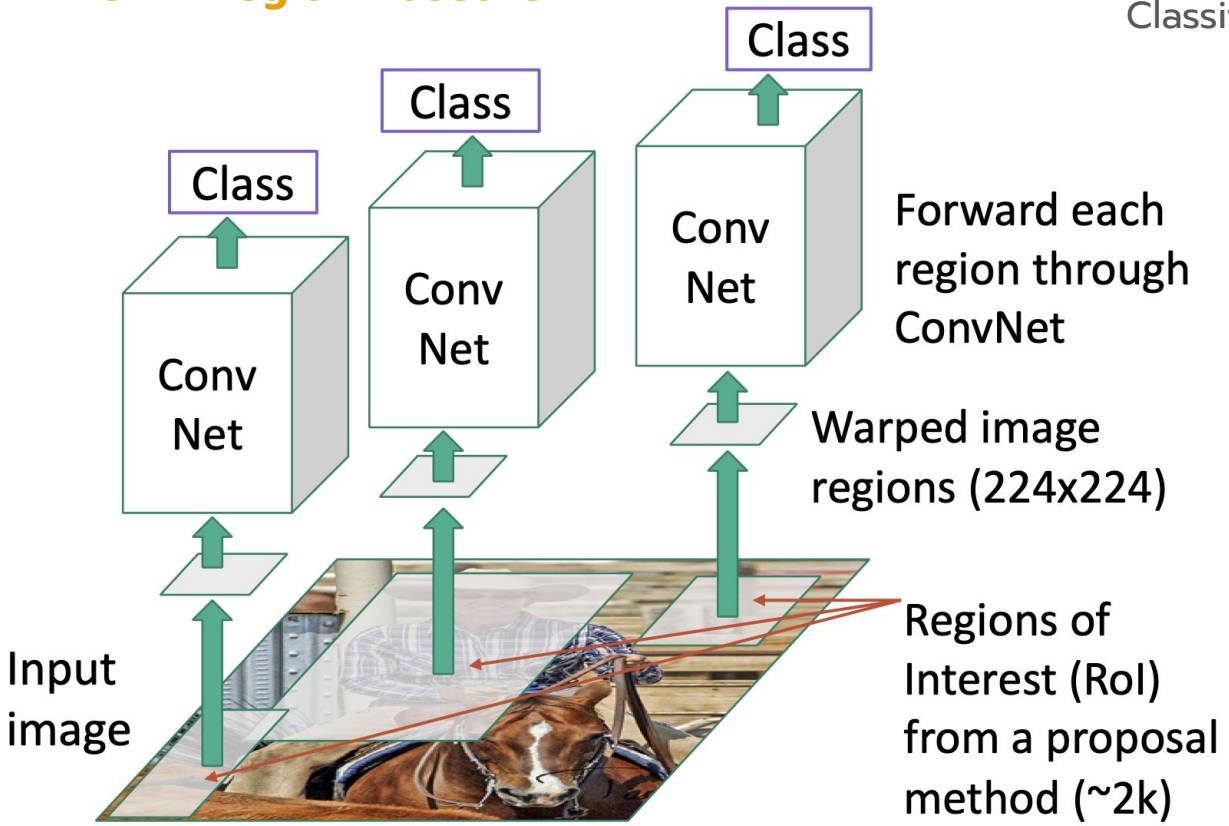
Object Detection

R-CNN: Region-Based CNN

Classify each region

Classification over ***C+1*** categories

Classification over ***C+1*** categories



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Cherdsak Kingkan

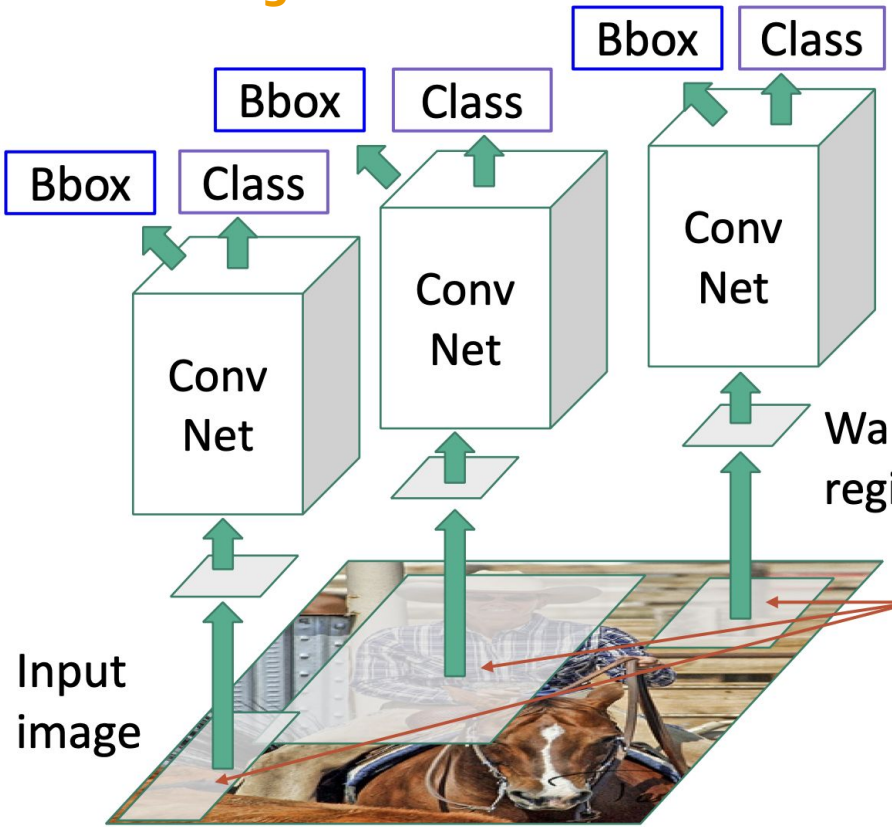
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

Object Detection

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)



Forward each region through ConvNet

Warped image regions (224x224)

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

Input image

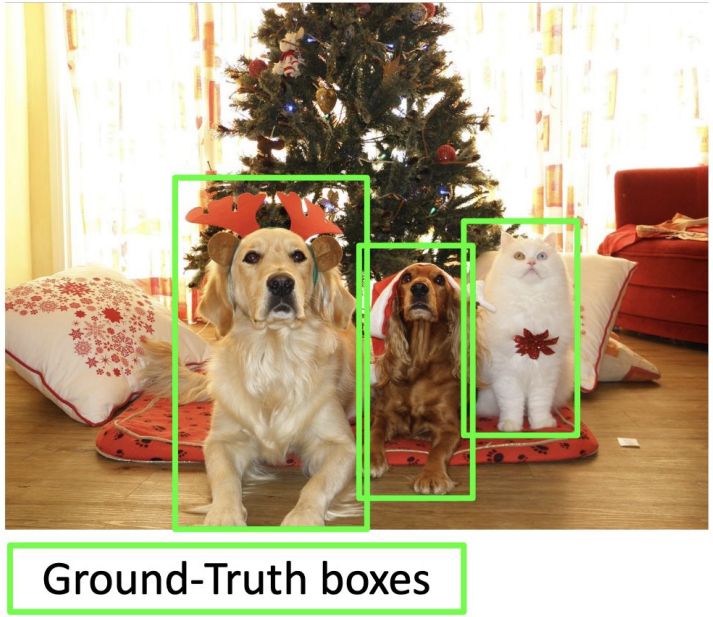
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN - Training

Input Image

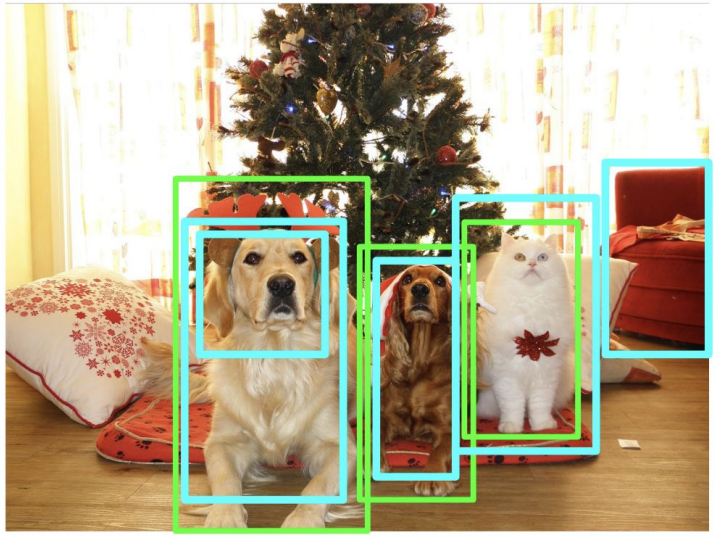


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

R-CNN: Region-Based CNN - Training

Input Image



Ground-Truth boxes

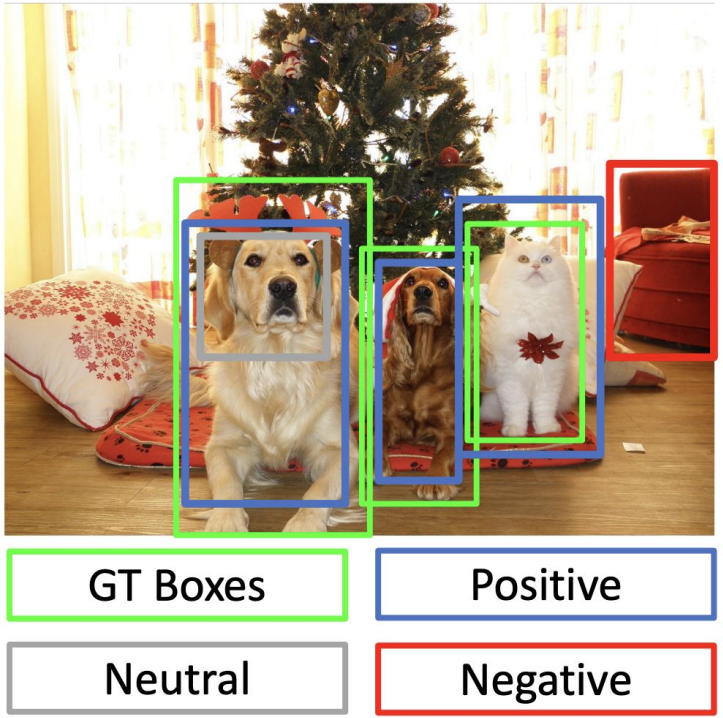
Region Proposals

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

R-CNN: Region-Based CNN - Training

Input Image



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

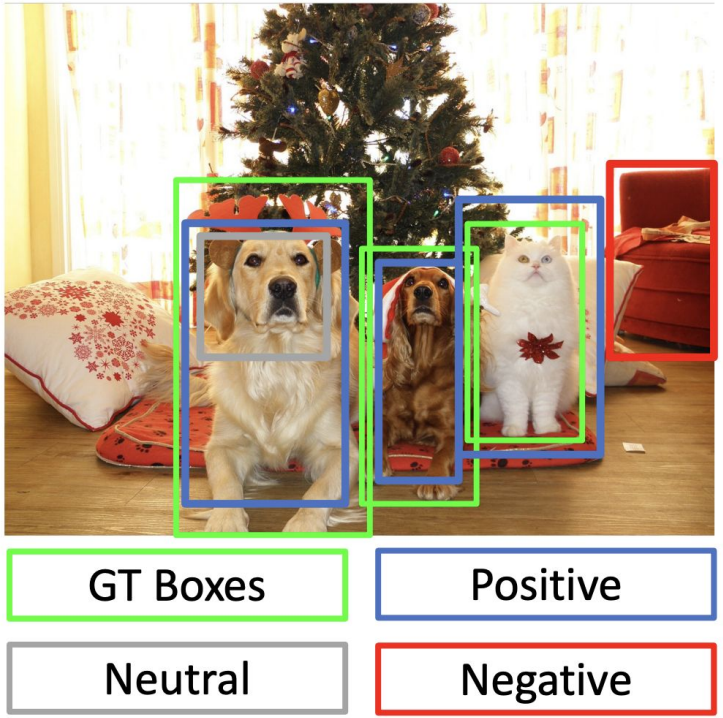
Positive: > 0.5 IoU with a GT box
Negative: < 0.3 IoU with all GT boxes
Neutral: between 0.3 and 0.5 IoU with GT boxes

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

R-CNN: Region-Based CNN - Training

Input Image



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

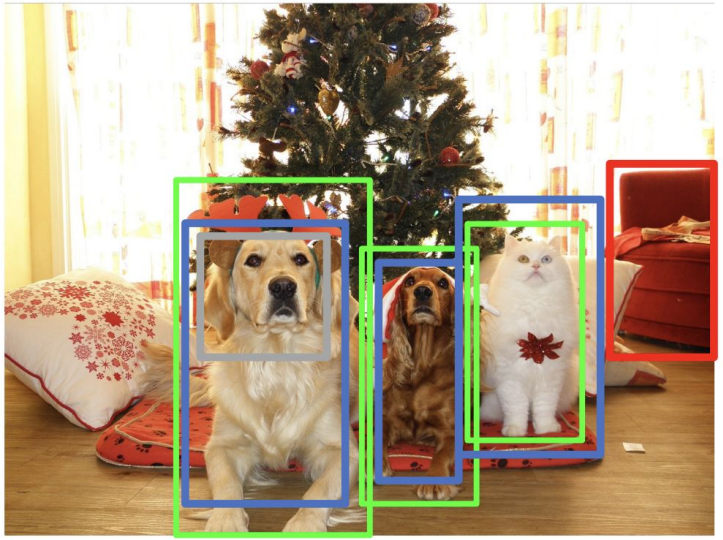
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

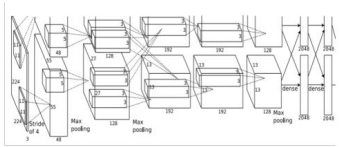
R-CNN: Region-Based CNN - Training

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

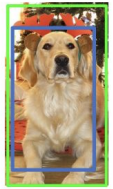
Input Image

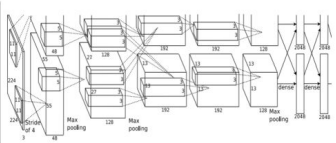


GT Boxes | Positive
Neutral | Negative

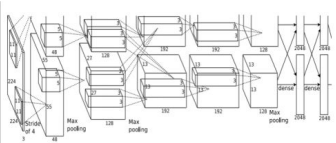


Class target: Dog
Box target: →

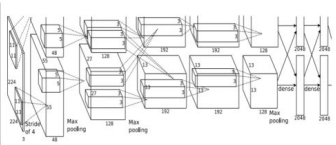




Class target: Cat
Box target: →



Class target: Dog
Box target: →



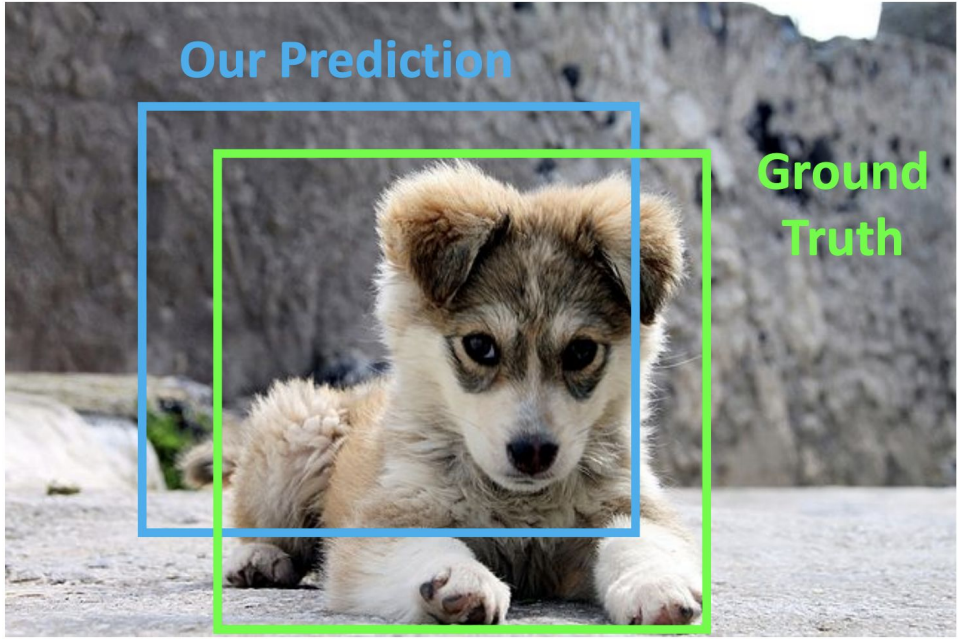
Class target: Background
Box target: None

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Comparing Boxes: Intersection over Union (IoU)

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



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

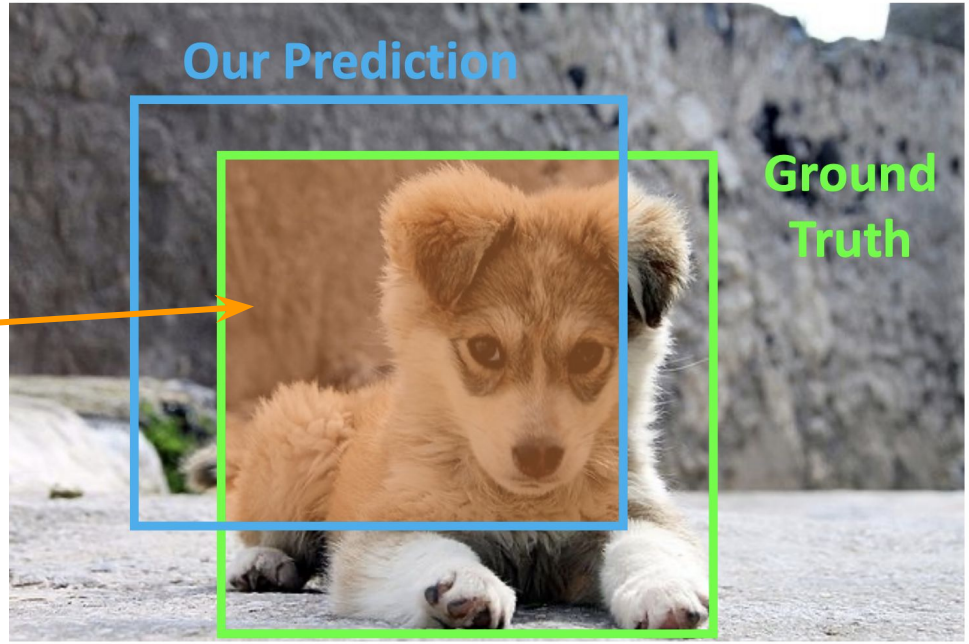
Object Detection

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"):

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



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

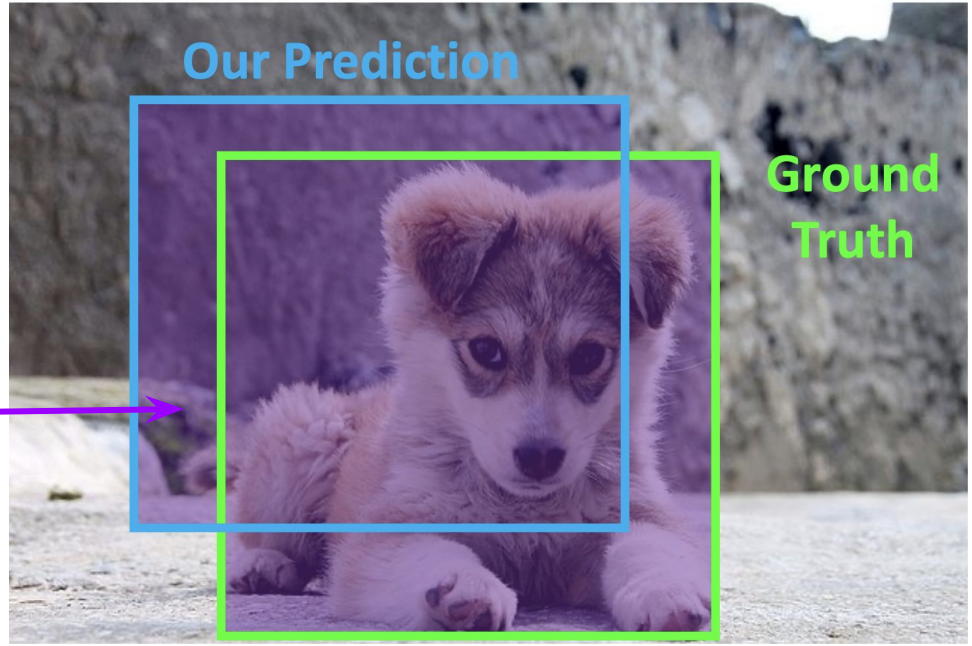
Object Detection

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"):

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



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

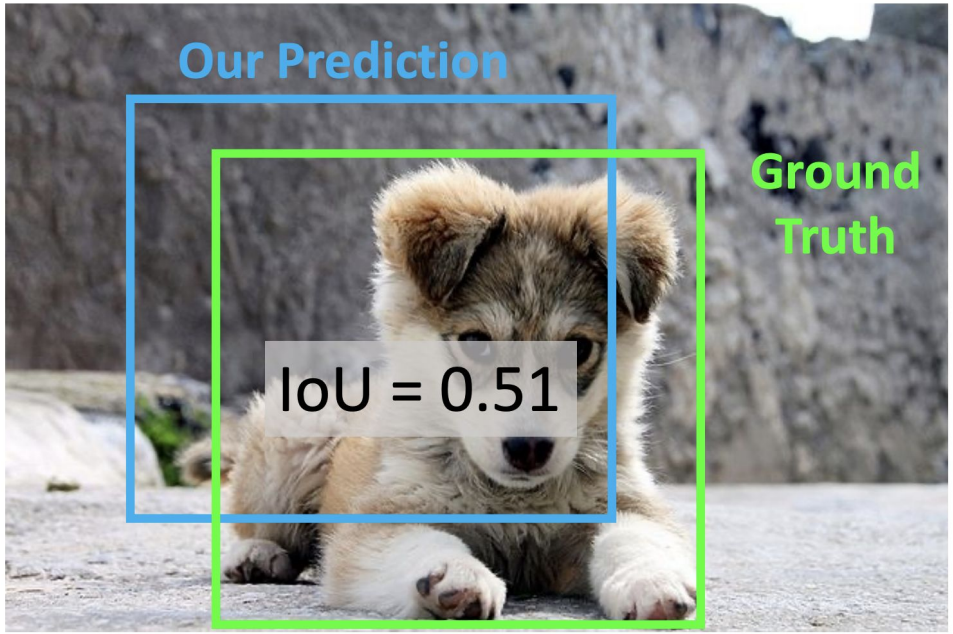
Comparing Boxes: Intersection over Union (IoU)

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

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

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

IoU > 0.5 is "decent".



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

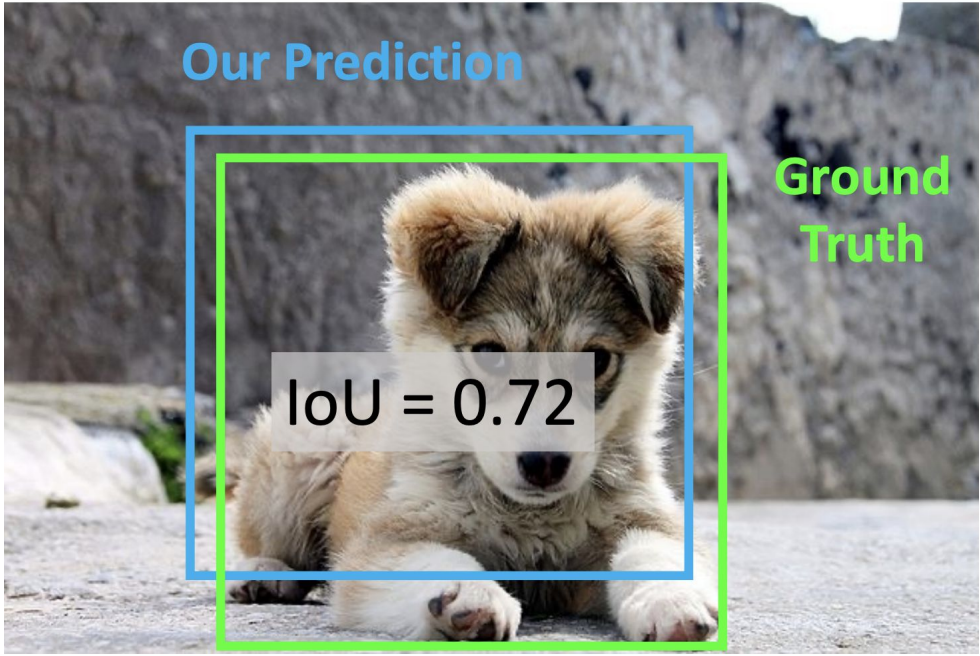
Comparing Boxes: Intersection over Union (IoU)

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

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

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

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



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

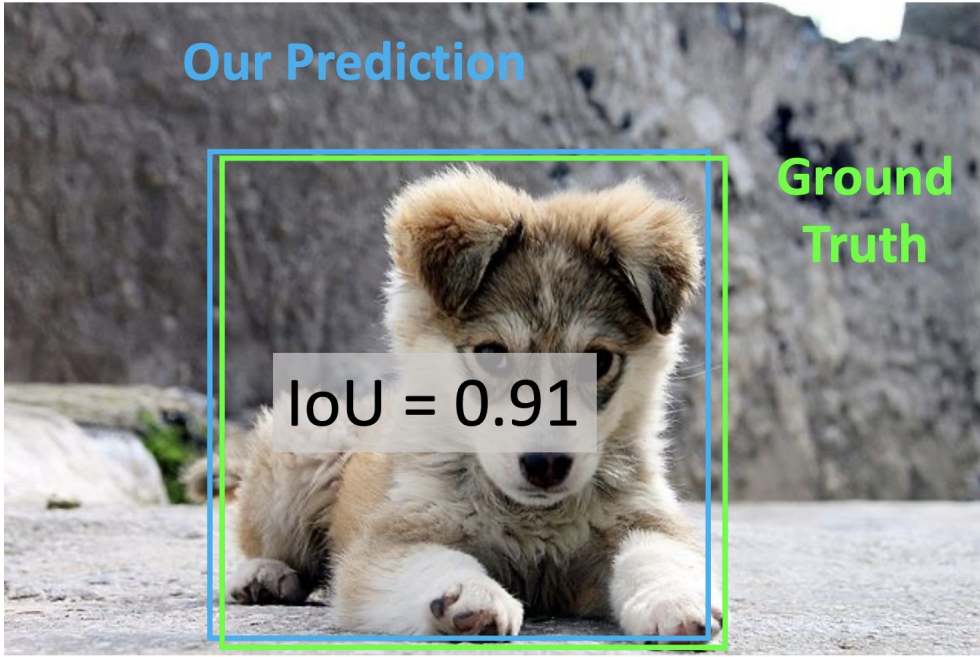
Comparing Boxes: Intersection over Union (IoU)

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

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

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

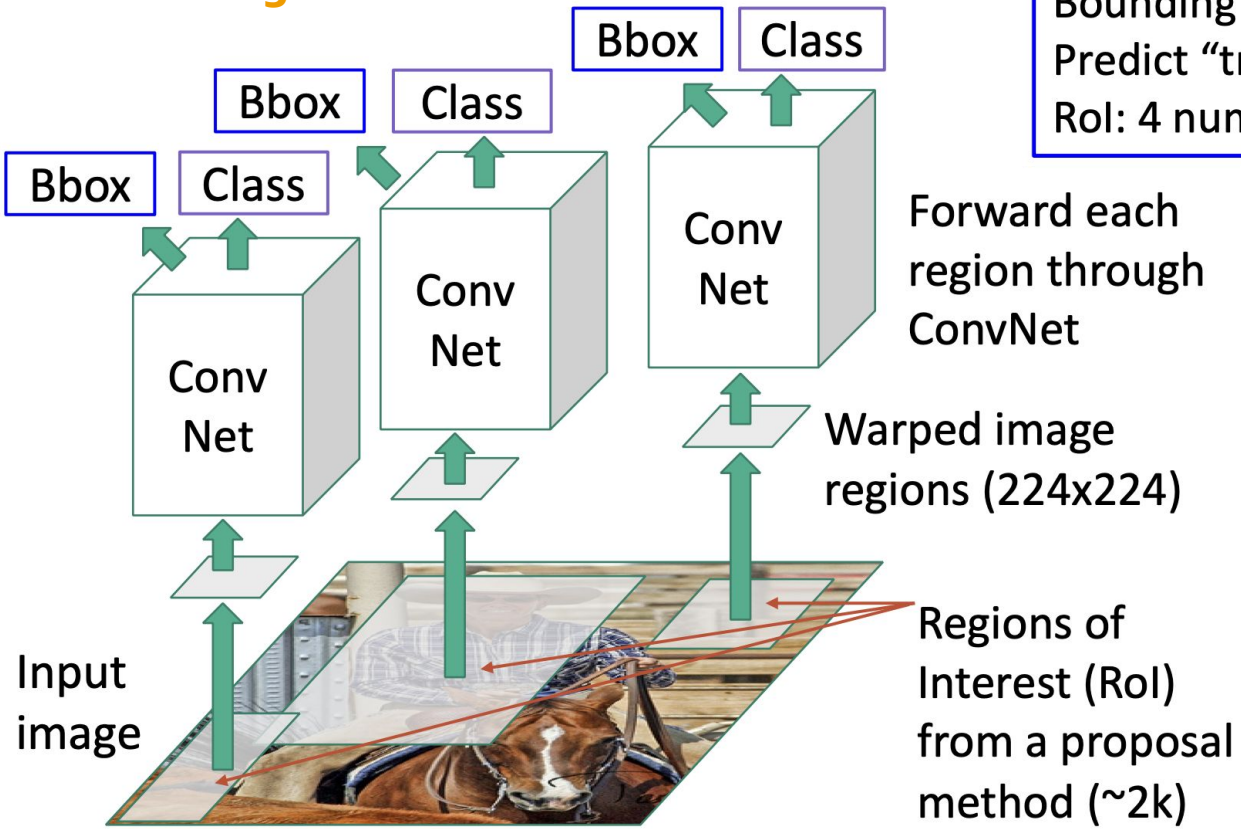
IoU > 0.5 is "decent".
IoU > 0.7 is "pretty good"
IoU > 0.9 is "almost perfect"



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

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)

Forward each region through ConvNet

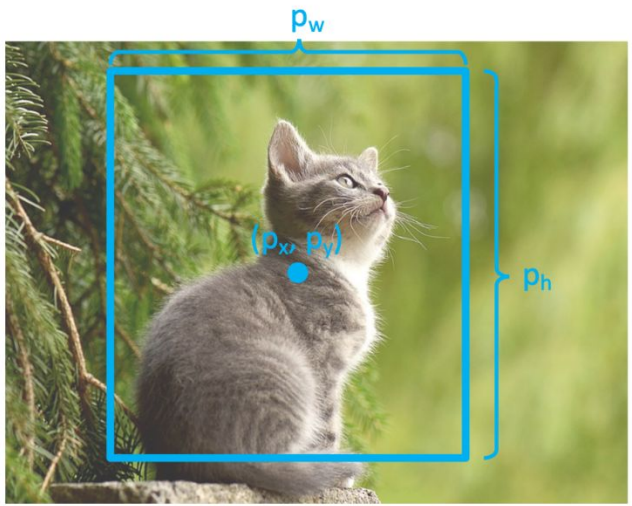
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN

Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

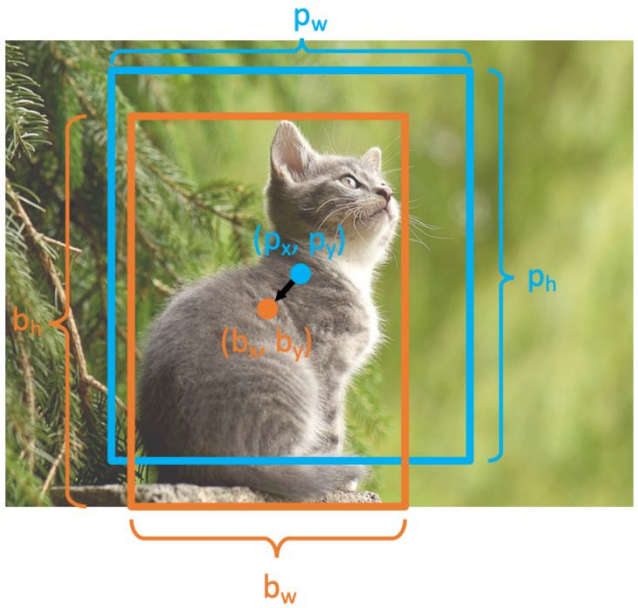
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN

Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

The output box is defined by:

$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$

Shift center by amount relative to proposal size

$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

Scale proposal; exp ensures that scaling factor is > 0

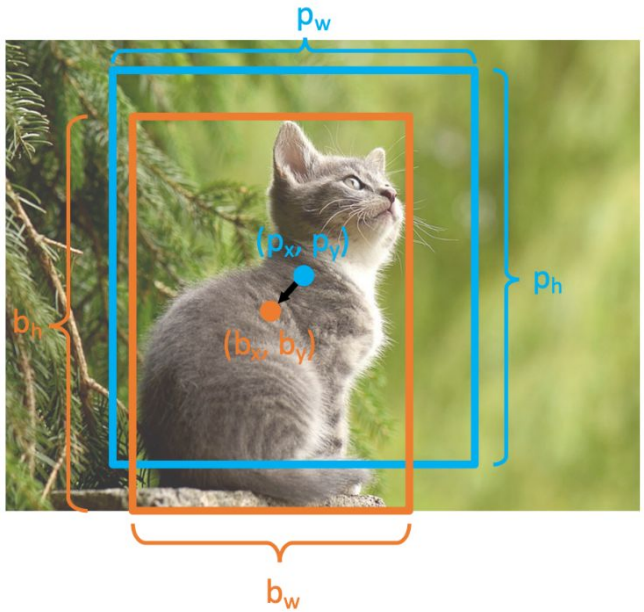
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN

Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

The output box is:

$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$
$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

When transform is 0, output = proposal

L2 regularization encourages leaving proposal unchanged

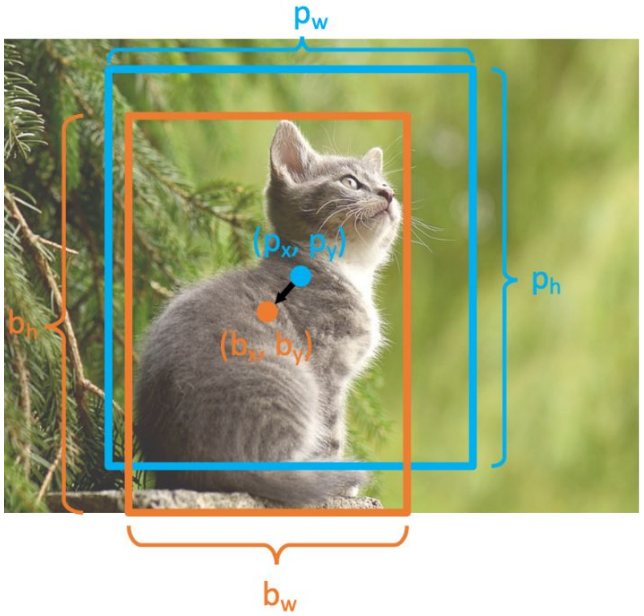
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN

Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

The output box is:

$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$
$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

Scale / Translation invariance: Transform encodes *relative* difference between proposal and output; important since CNN doesn't see absolute size or position after cropping

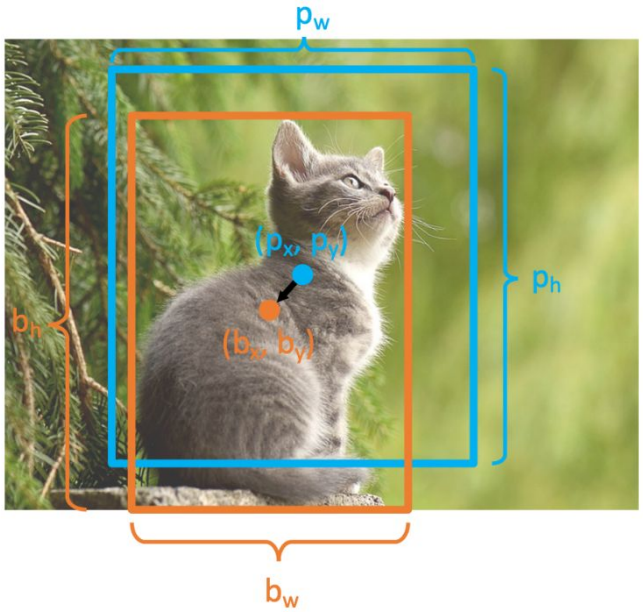
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN

Box Regression



Consider a region proposal with center (p_x, p_y), width p_w, height p_h

Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

The output box is:

$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$
$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

Given proposal and target output, we can solve for the transform the network should output:

$$t_x = (b_x - p_x)/p_w$$
$$t_y = (b_y - p_y)/p_h$$
$$t_w = \log(b_w/p_w)$$
$$t_h = \log(b_h/p_h)$$

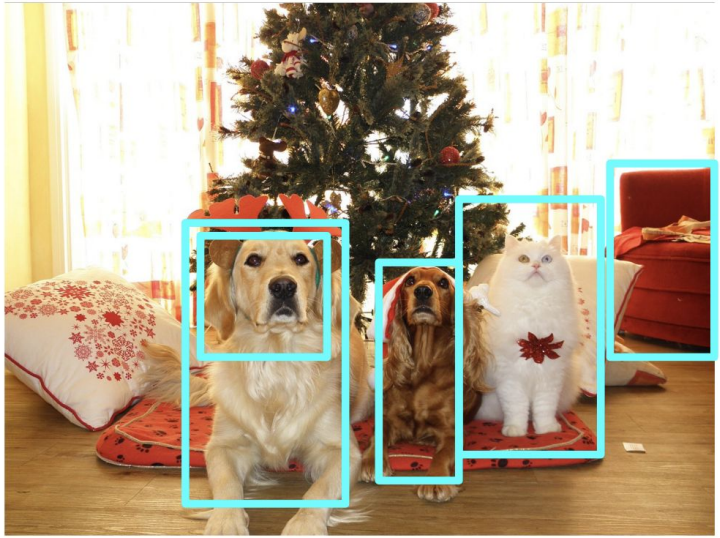
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

R-CNN: Region-Based CNN - Test-Time

Input Image



Region Proposals

1. Run proposal method (~2k region proposals)
2. Resize and run CNN on each proposal to get class scores, transforms
3. Threshold class scores to get a set of detections.

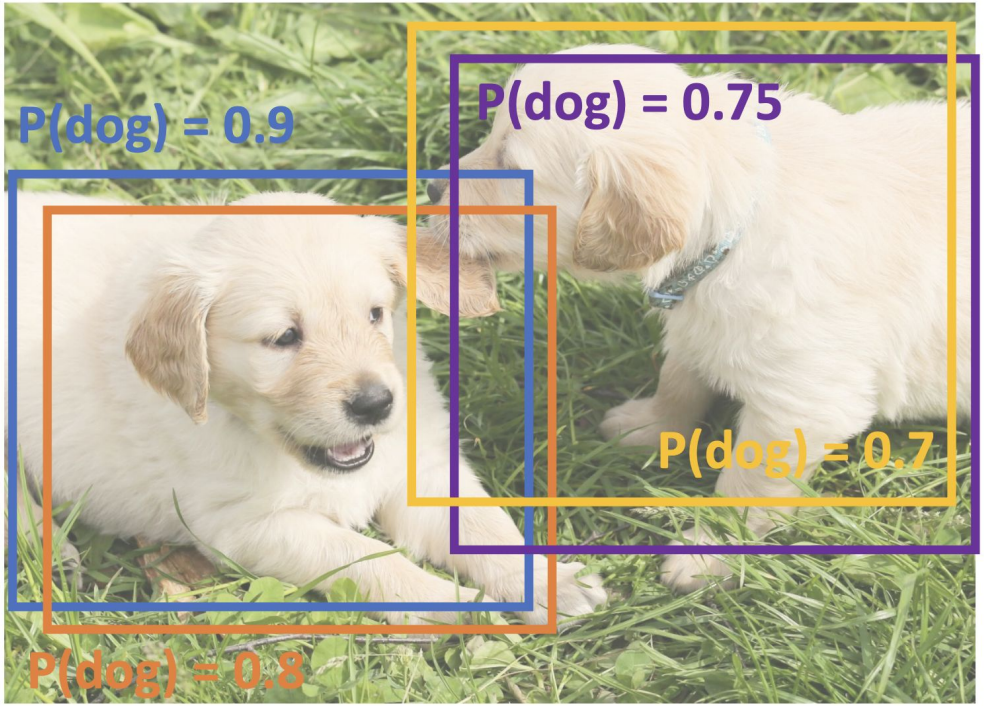
2 problems:
- CNN often outputs overlapping boxes
- How to set thresholds?

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Overlapping Boxes

Problem: Object detectors often output many overlapping detections:



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

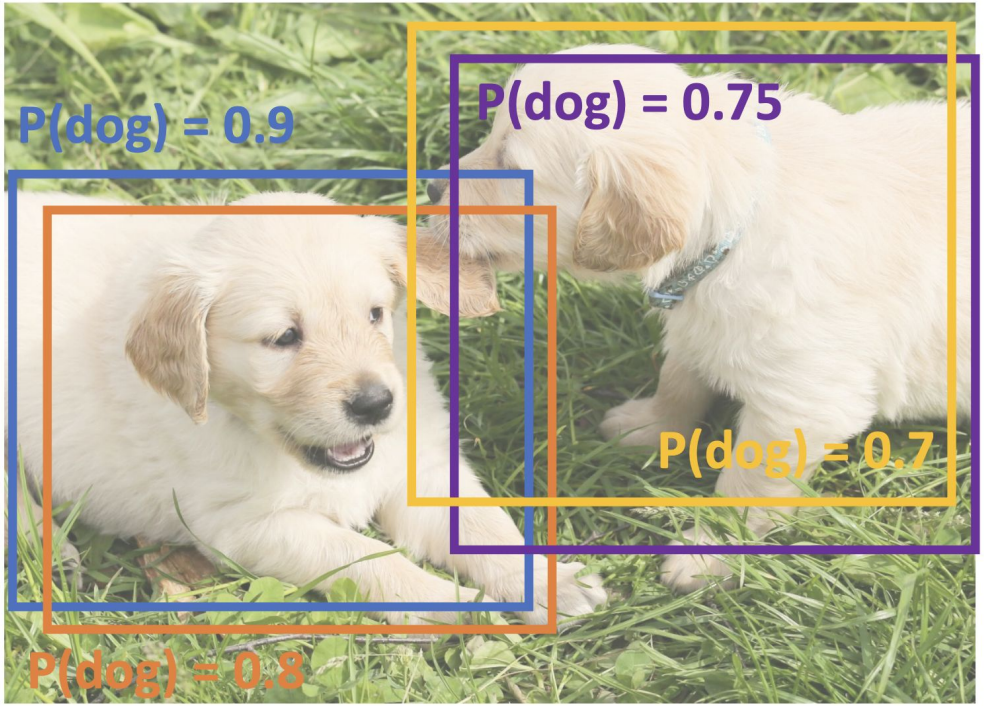
Object Detection

Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

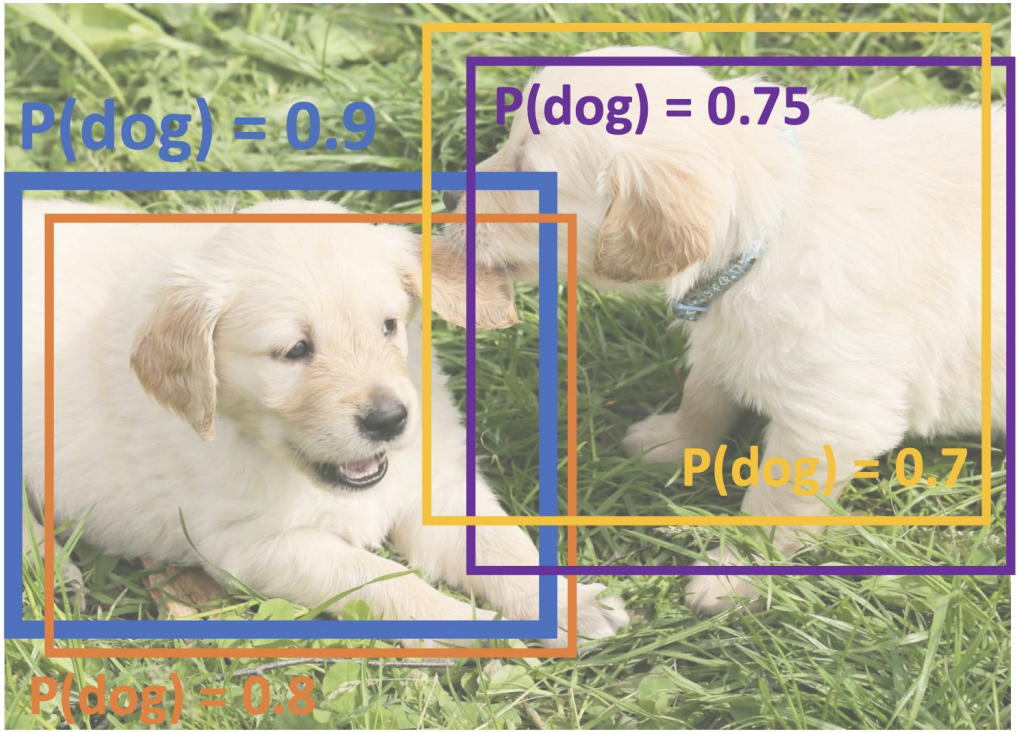
Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

IoU(,) = **0.78**
IoU(,) = 0.05
IoU(,) = 0.07



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

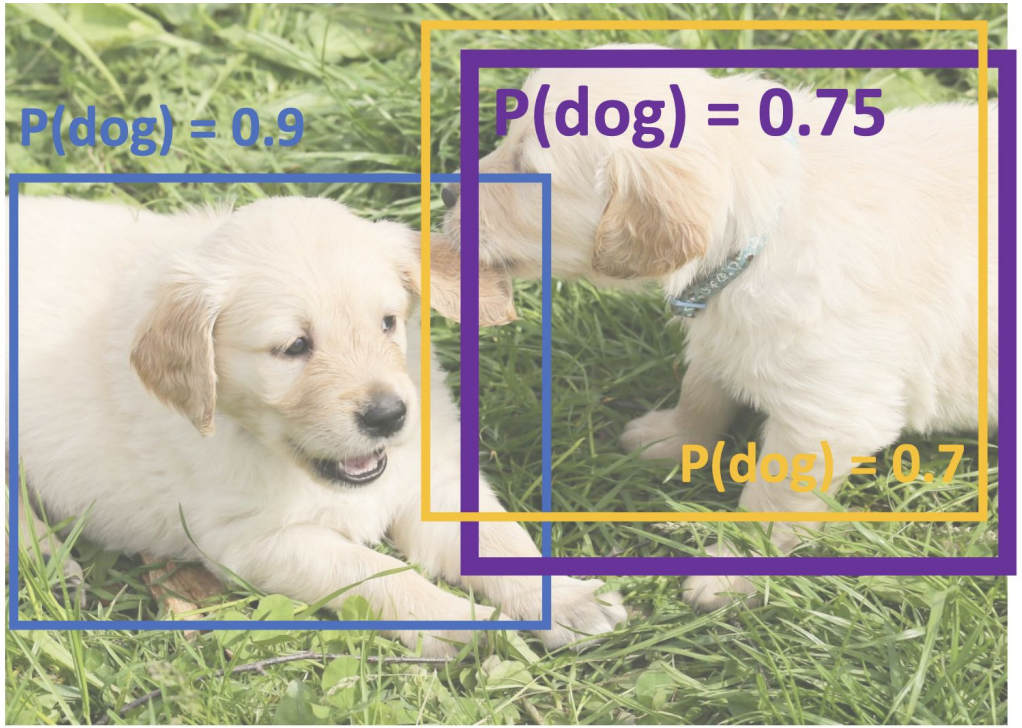
Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1

IoU(,) = **0.74**



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

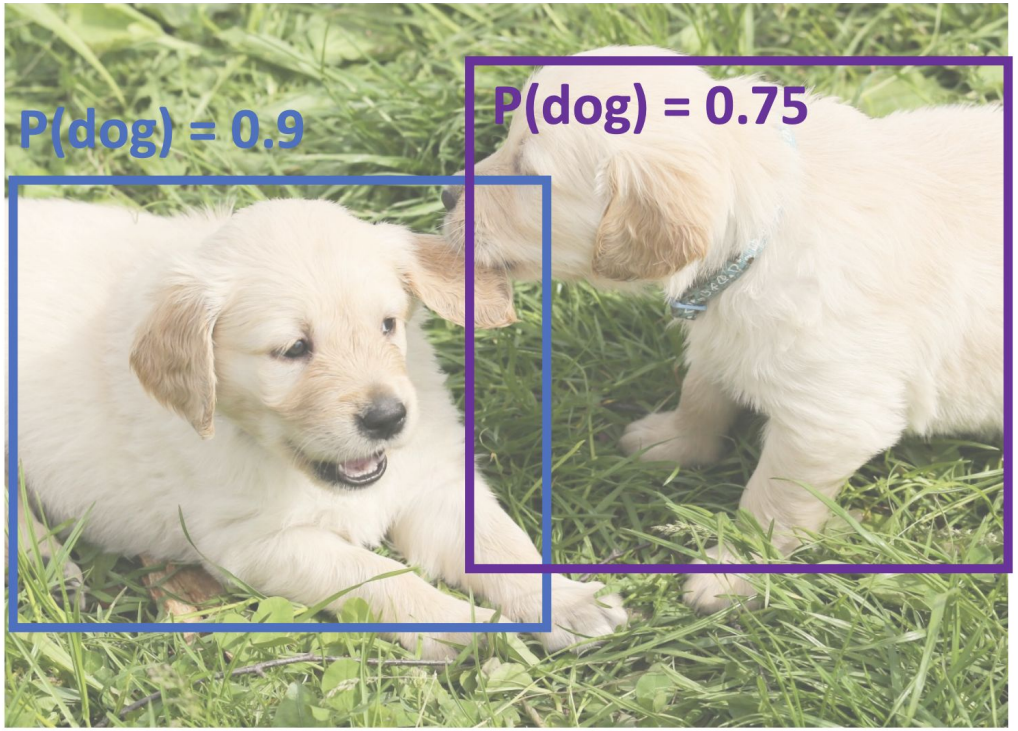
Object Detection

Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
3. If any boxes remain, GOTO 1



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Overlapping Boxes: Non-Max Suppression (NMS)

Problem: Object detectors often output many overlapping detections:

Solution: Post-process raw detections using **Non-Max Suppression (NMS)**

- Select next highest-scoring box
- Eliminate lower-scoring boxes with IoU > threshold (e.g. 0.7)
- If any boxes remain, GOTO 1

Problem: NMS may eliminate "good" boxes when objects are highly overlapping… no good solution

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (mAP)

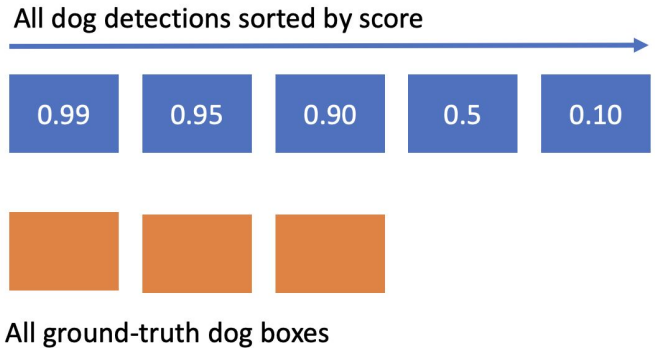
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP)
   = area under Precision vs Recall Curve

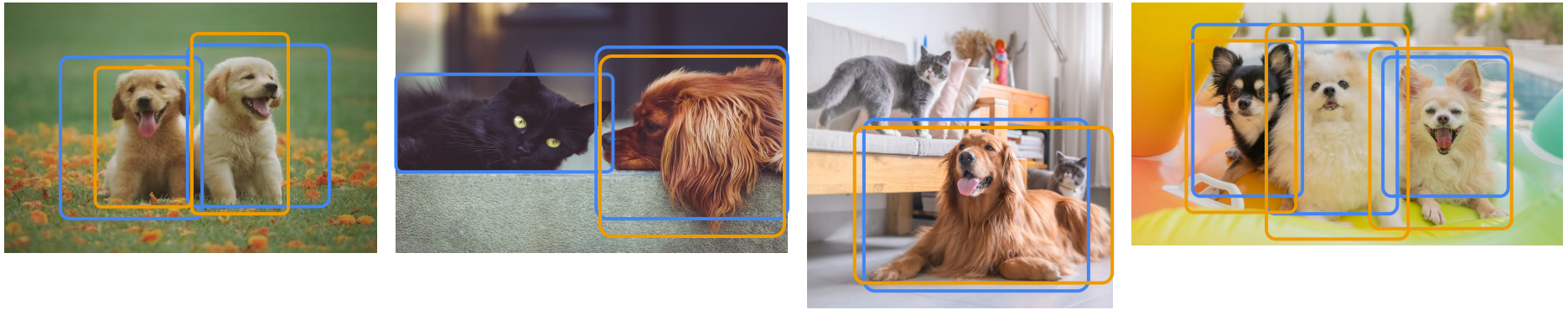
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)

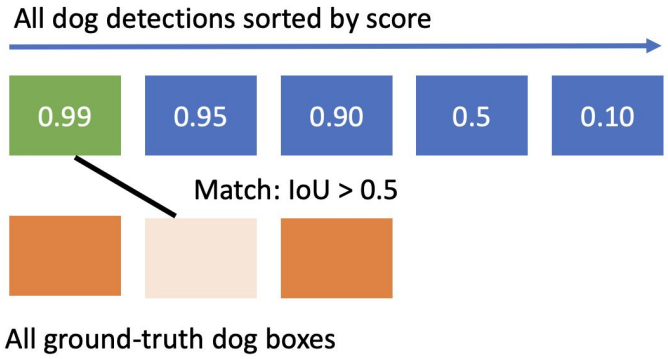




Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative

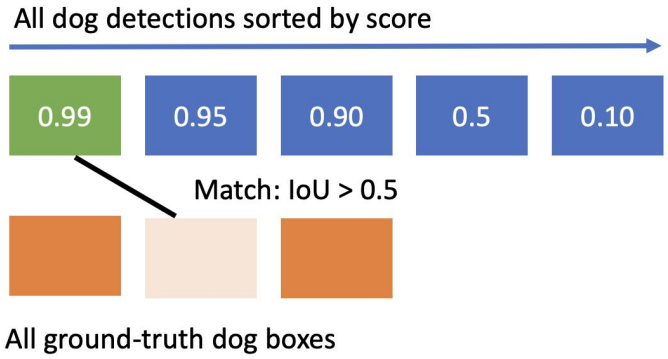


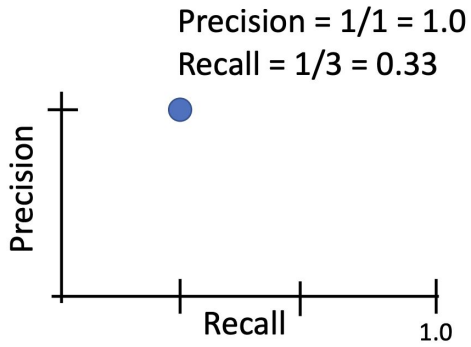
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve



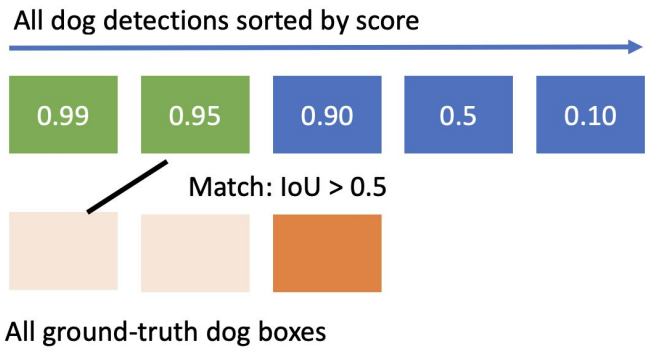


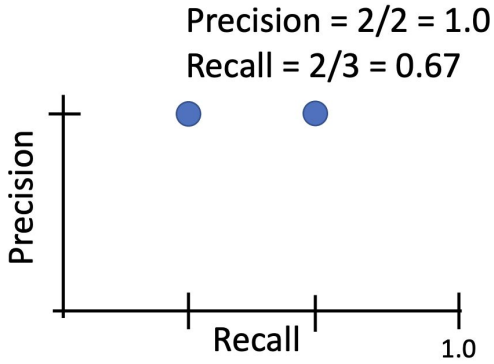
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve



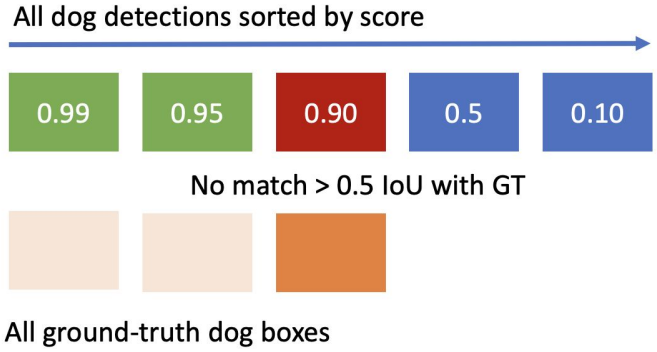


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

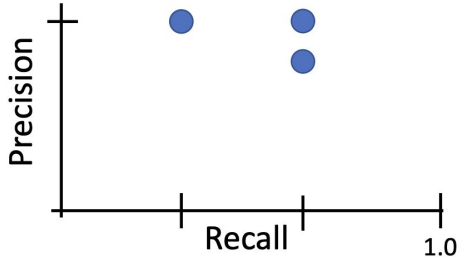
Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve



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

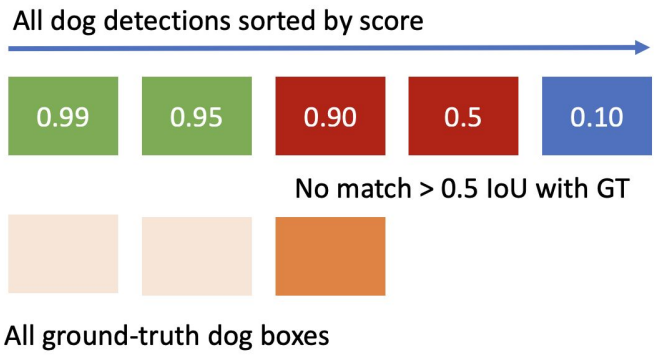


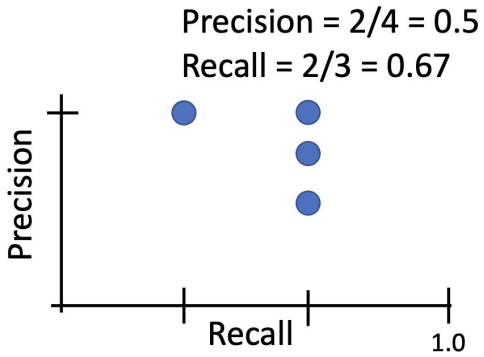
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve



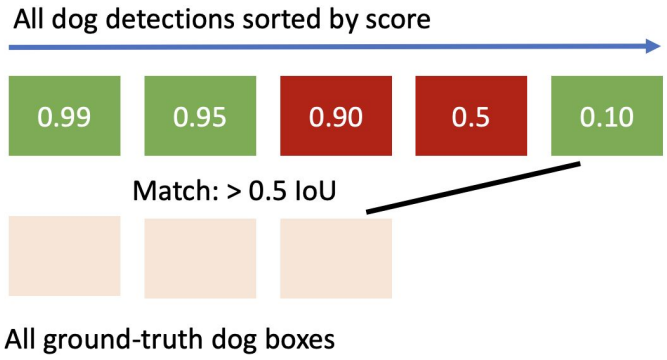


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

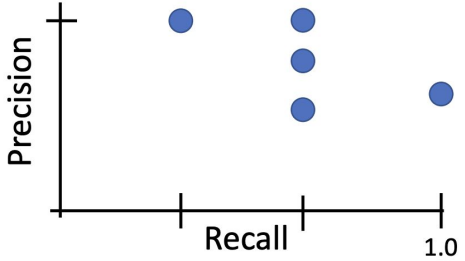
Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve



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

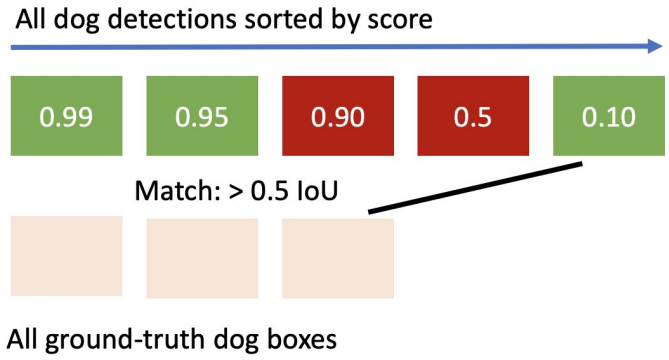


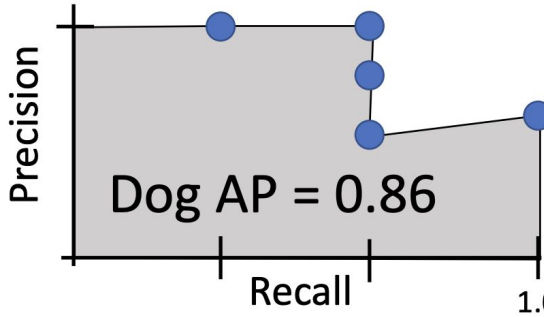
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve
   b. Average Precision (AP) = area under PR curve



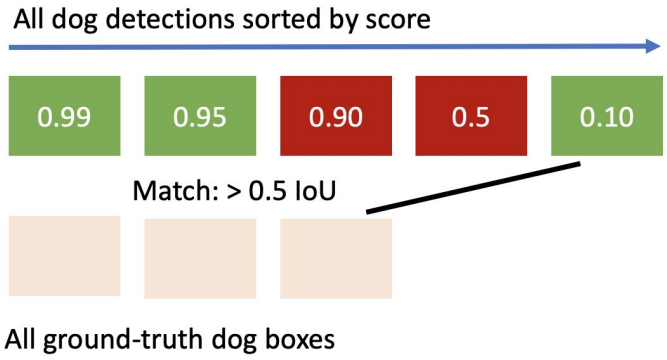


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

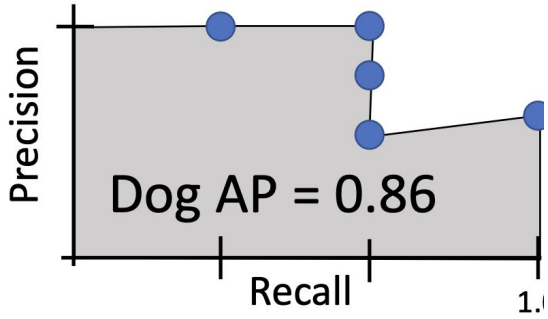
Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve
   b. 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".



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve
   b. 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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Evaluating Object Detectors: Mean Average Precision (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
   a. For each detection (highest score to lowest score)
      i. If it matches some GT box with IoU > 0.5, mark it as positive and eliminate the GT
      ii. Otherwise mark it as negative
      iii. Plot a point on PR Curve
   b. 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

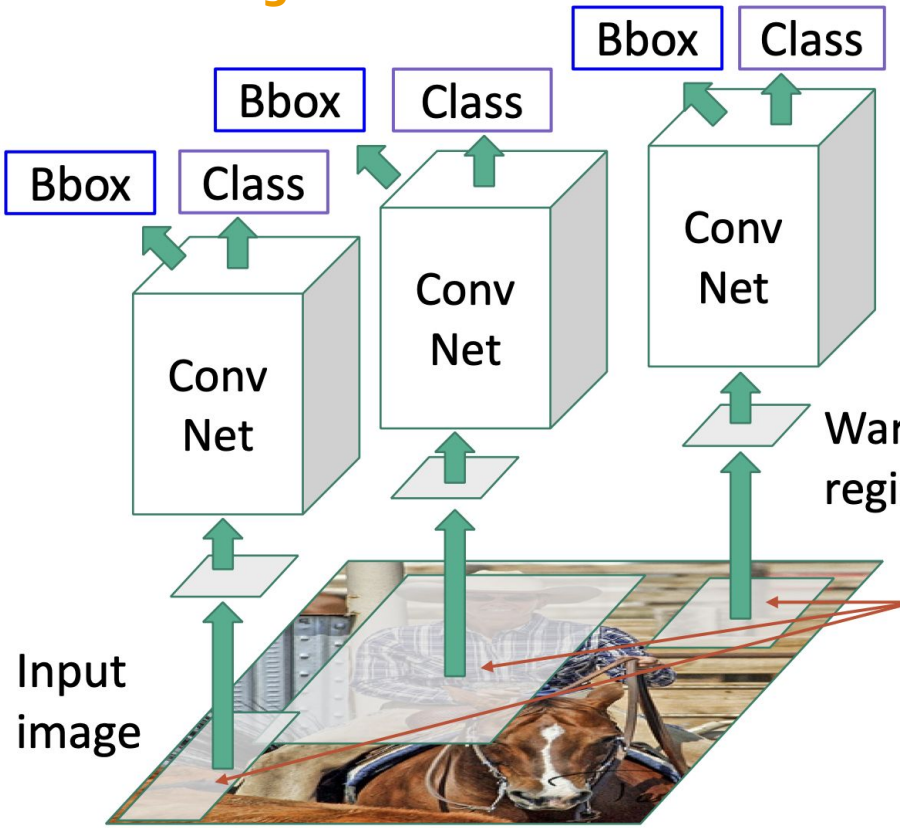
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

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)



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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

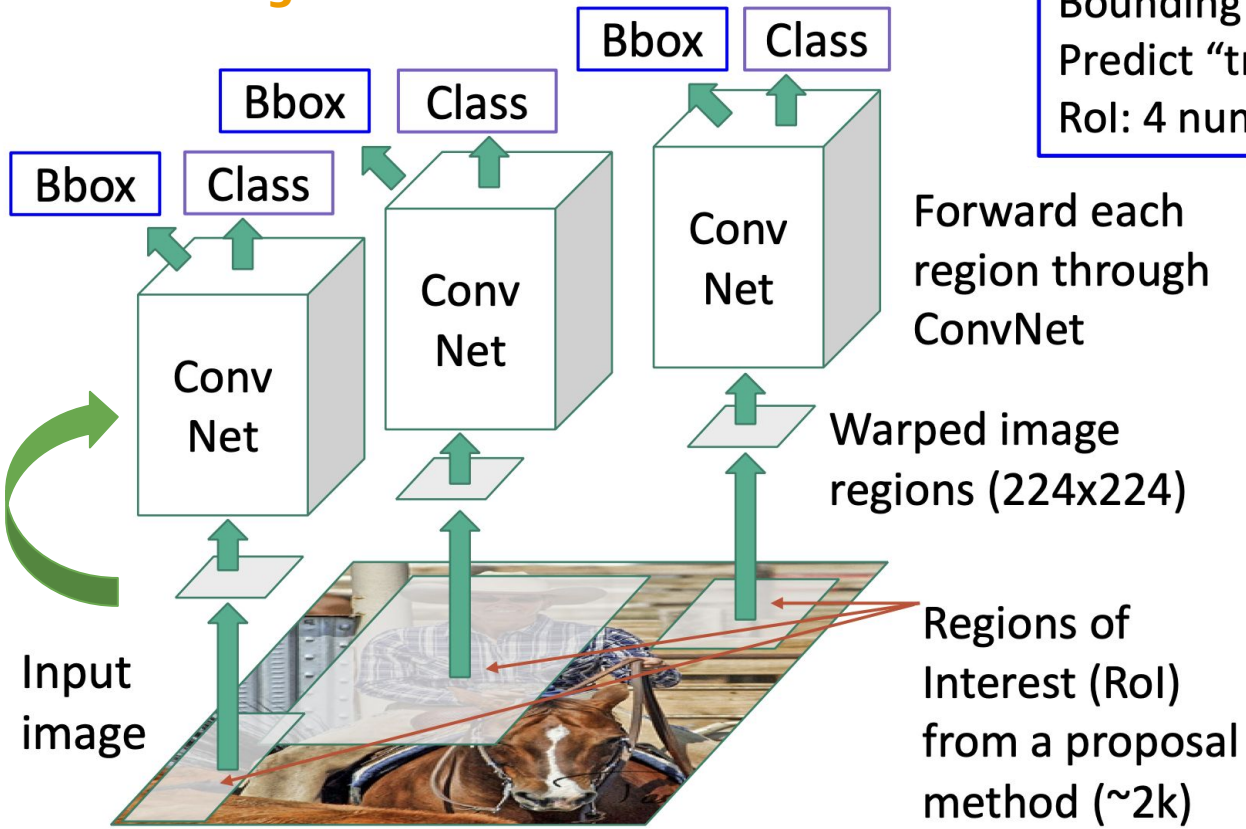
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

Object Detection

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)



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

Solution: Run CNN "before" warping.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

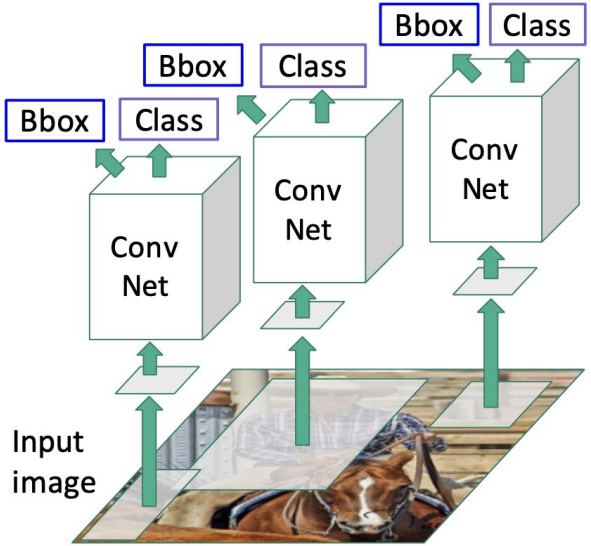
Object Detection

Fast R-CNN

Input image

"Slow" R-CNN

Process each region independently

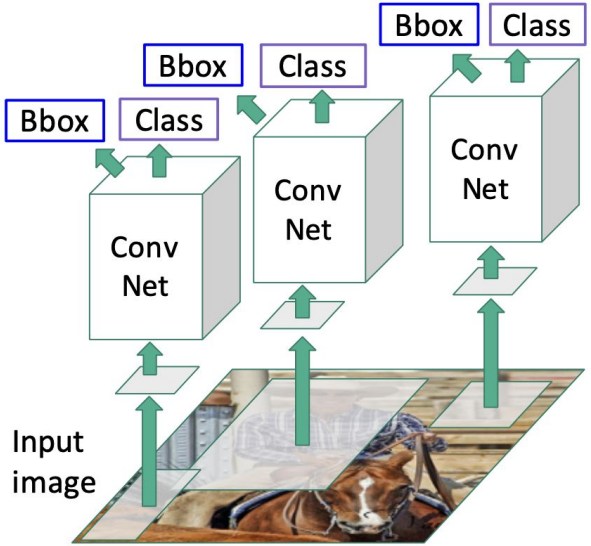


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN

"Slow" R-CNN
Process each region independently



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

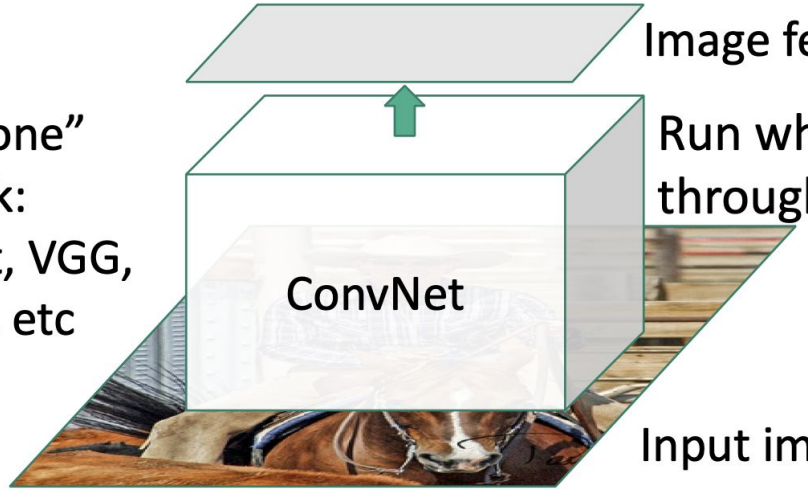


Image features

Run whole image through ConvNet

Input image

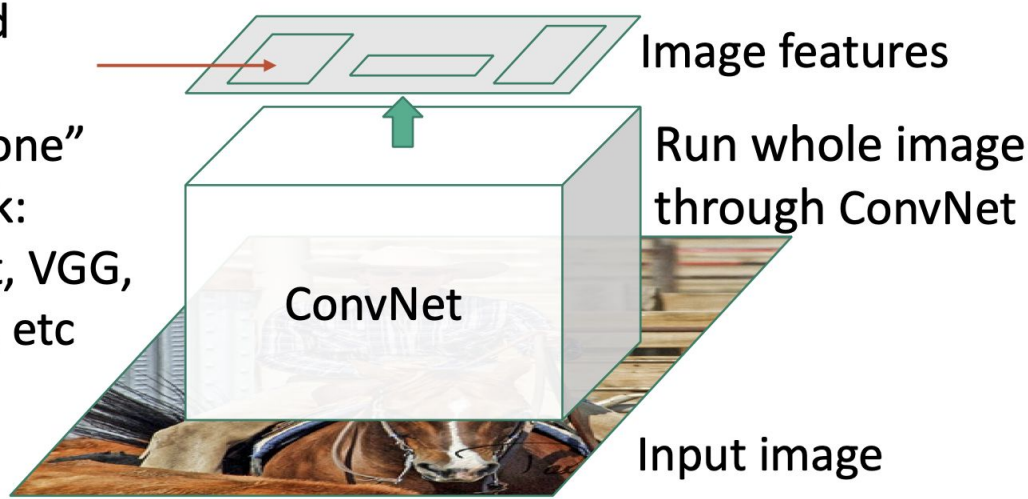
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN

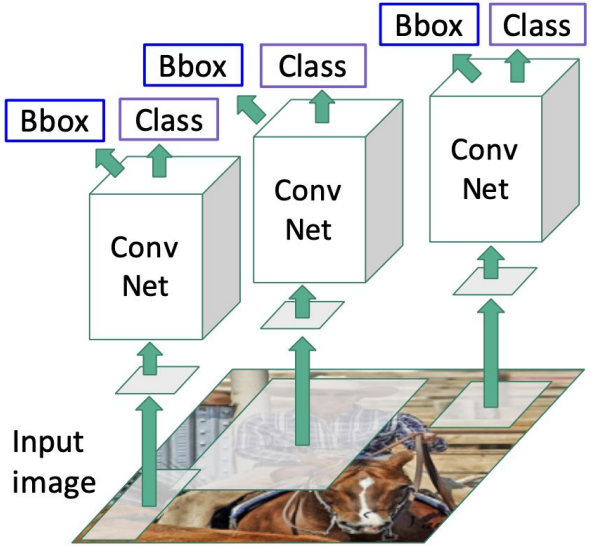
Regions of Interest (RoIs) from a proposal method

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



"Slow" R-CNN

Process each region independently



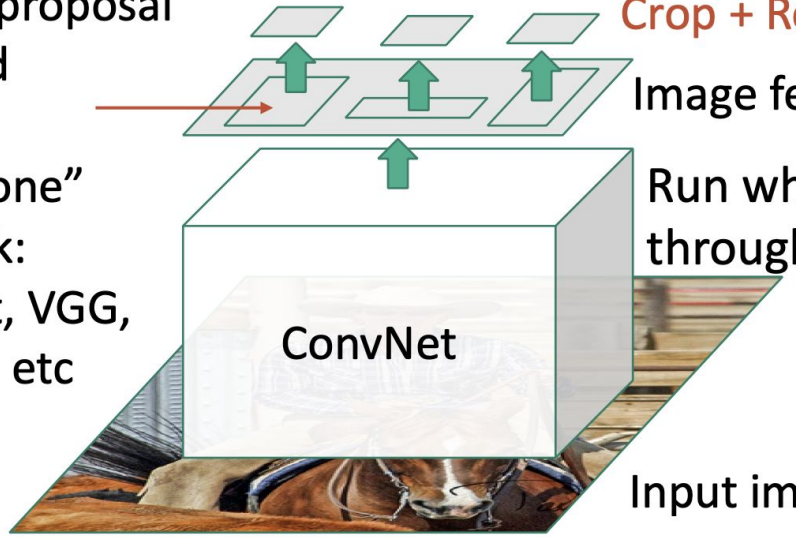
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN

Regions of Interest (RoIs) from a proposal method

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



Crop + Resize features

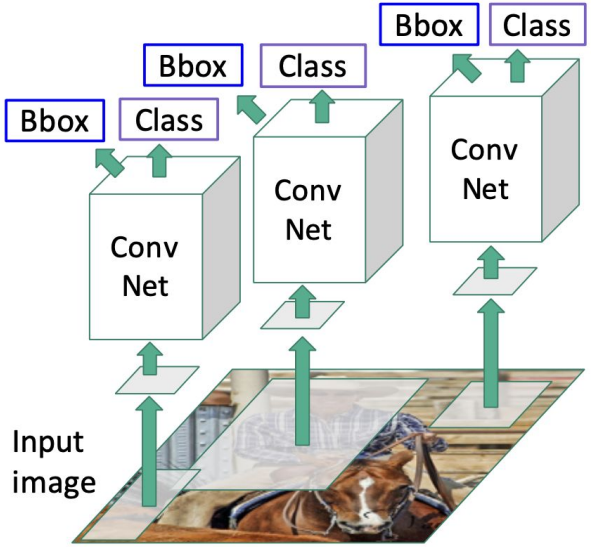
Image features

Run whole image through ConvNet

Input image

"Slow" R-CNN

Process each region independently



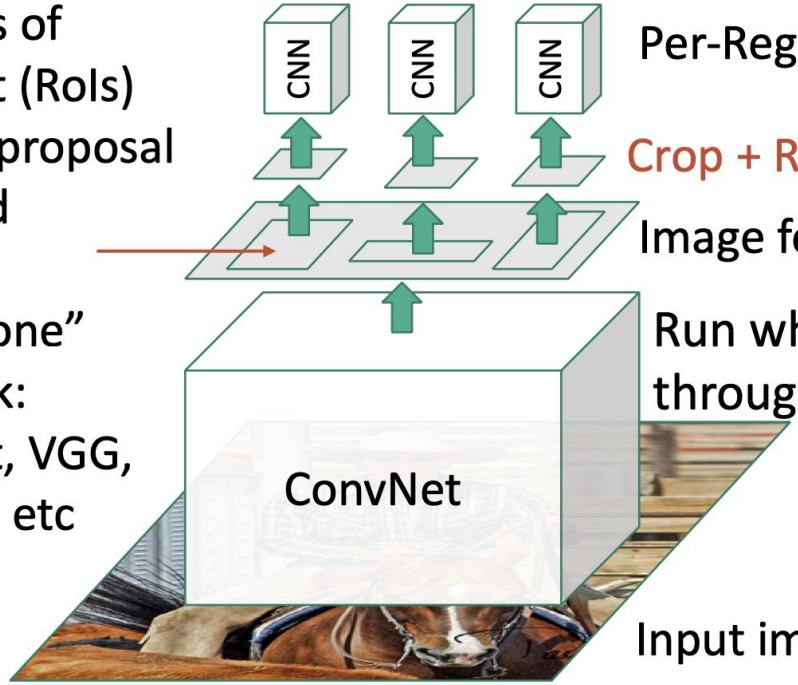
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN

Regions of Interest (RoIs) from a proposal method

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



Per-Region Network

Crop + Resize features

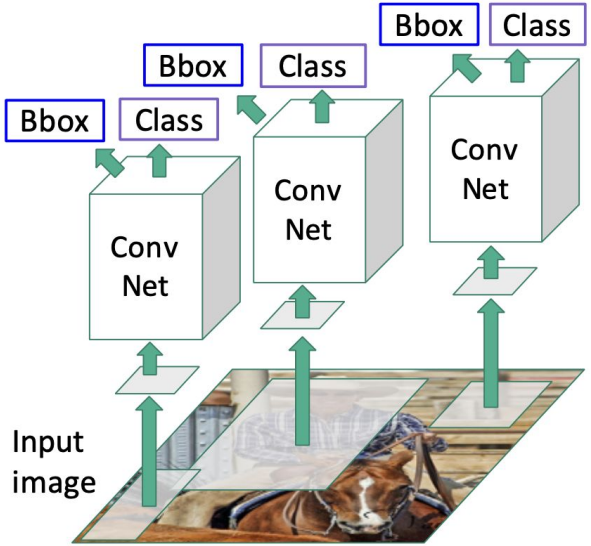
Image features

Run whole image through ConvNet

Input image

"Slow" R-CNN

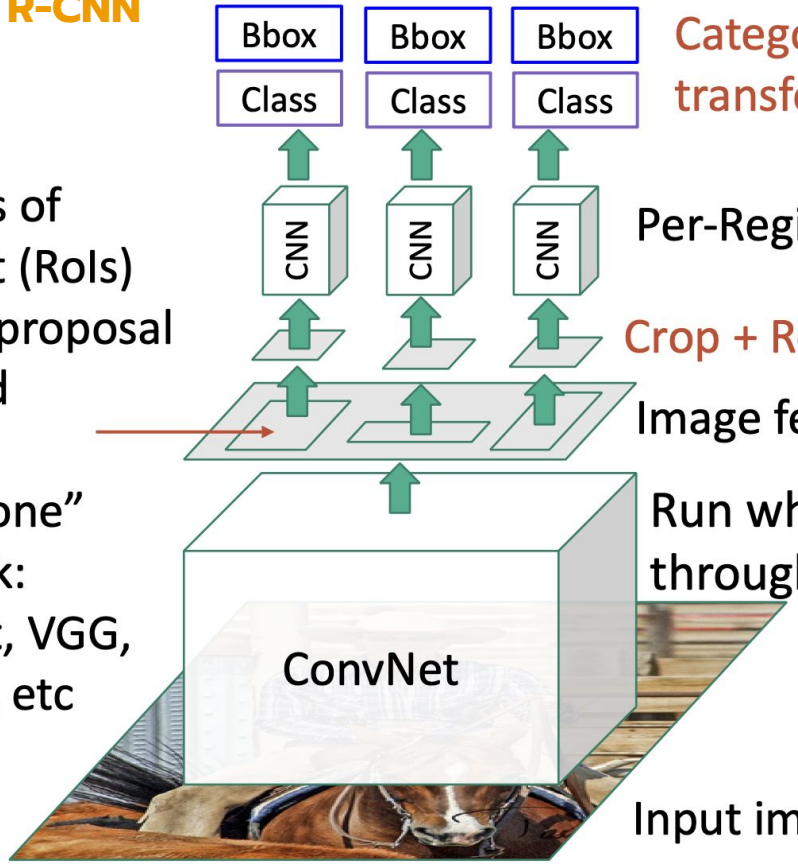
Process each region independently



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN



Category and box transform per region

Per-Region Network

Crop + Resize features

Image features

Run whole image through ConvNet

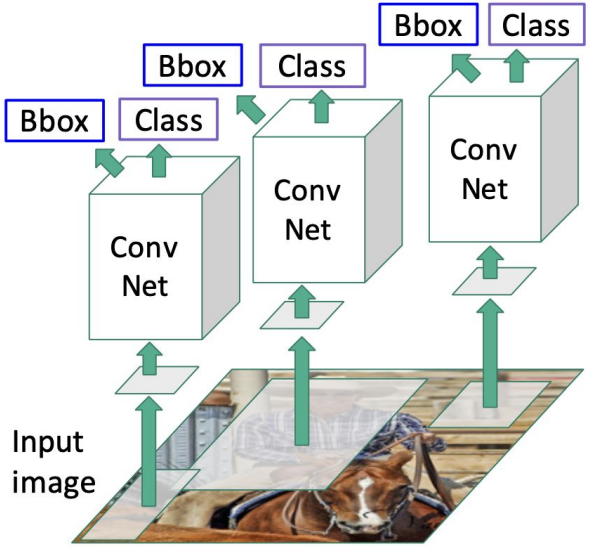
Input image

Regions of Interest (RoIs) from a proposal method

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

"Slow" R-CNN

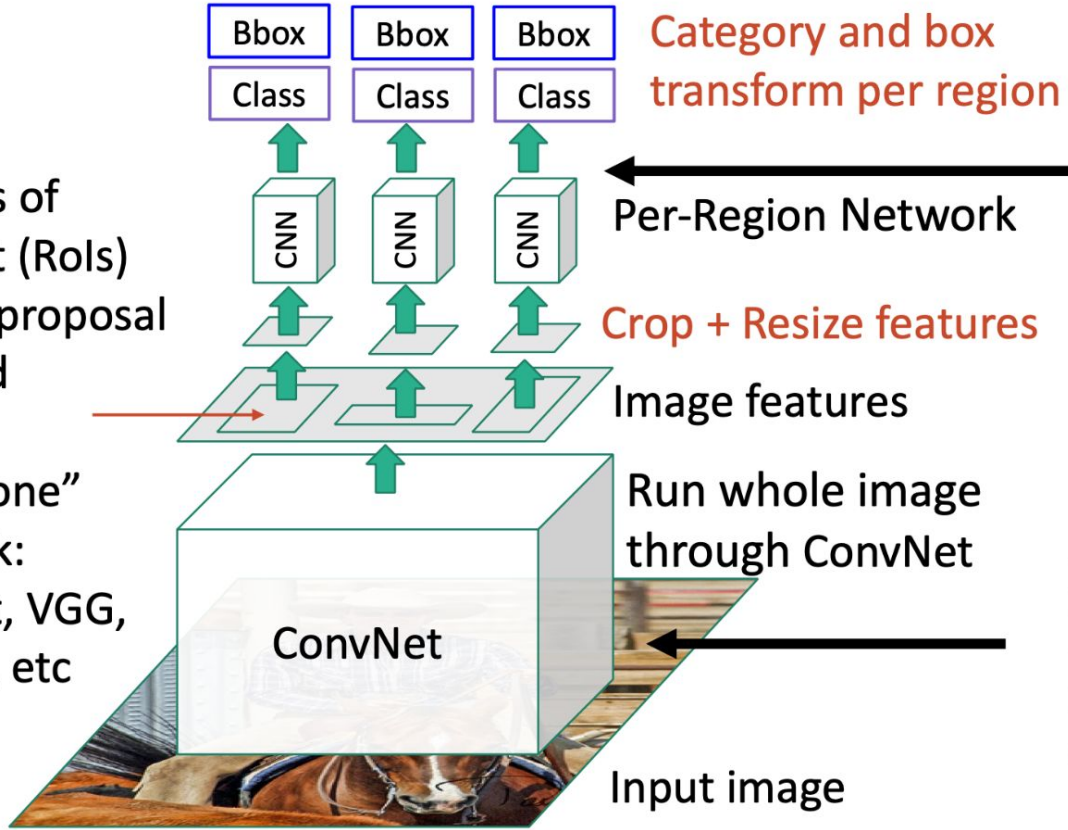
Process each region independently



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN



Category and box transform per region

Per-Region network is relatively lightweight

Per-Region Network

Regions of Interest (RoIs) from a proposal method

Crop + Resize features

Image features

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

Run whole image through ConvNet

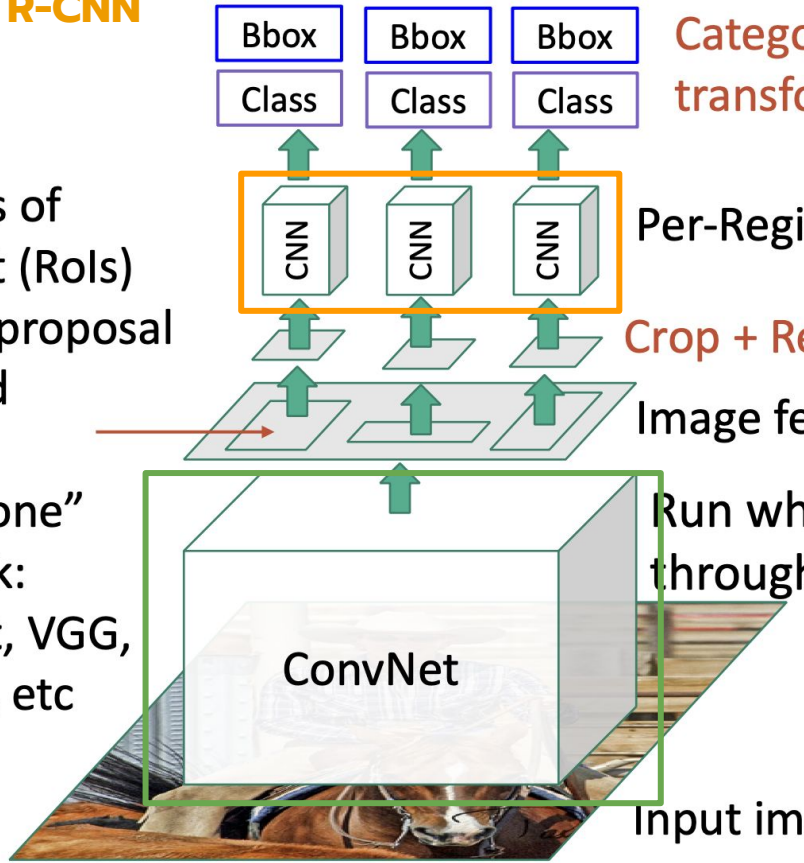
Most of the computation happens in backbone network; this saves work for overlapping region proposals

Input image

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN



Category and box transform per region

Per-Region Network

Crop + Resize features

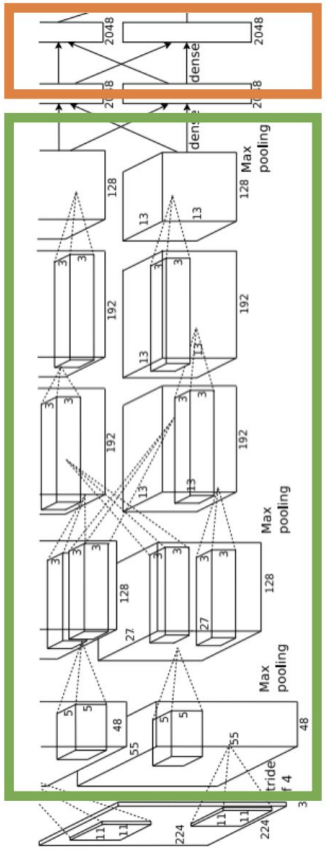
Image features

Run whole image through ConvNet

Input image

Regions of Interest (RoIs) from a proposal method

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

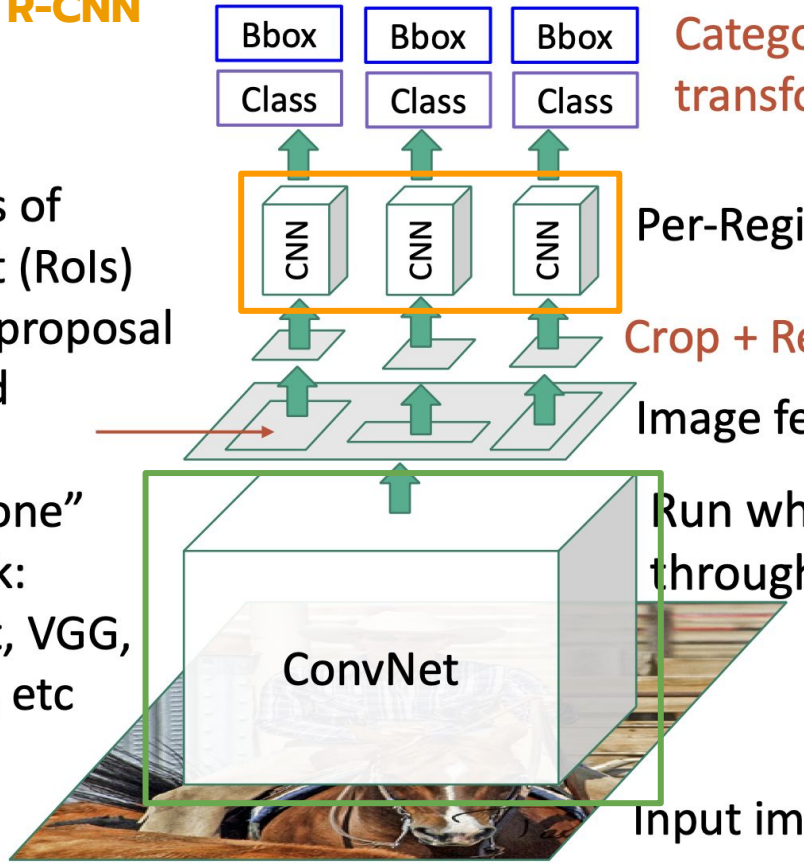


Example: When using AlexNet for detection, 5 conv layers are used for backbone and 2 FC layers are used for per-region network

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN



Category and box transform per region

Per-Region Network

Crop + Resize features

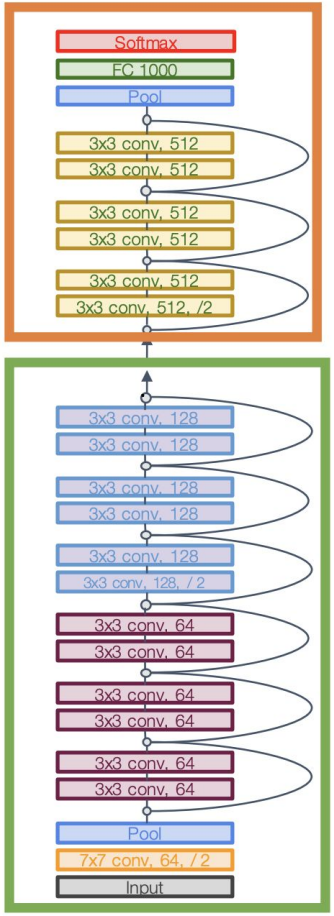
Image features

Run whole image through ConvNet

Input image

Regions of Interest (RoIs) from a proposal method

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

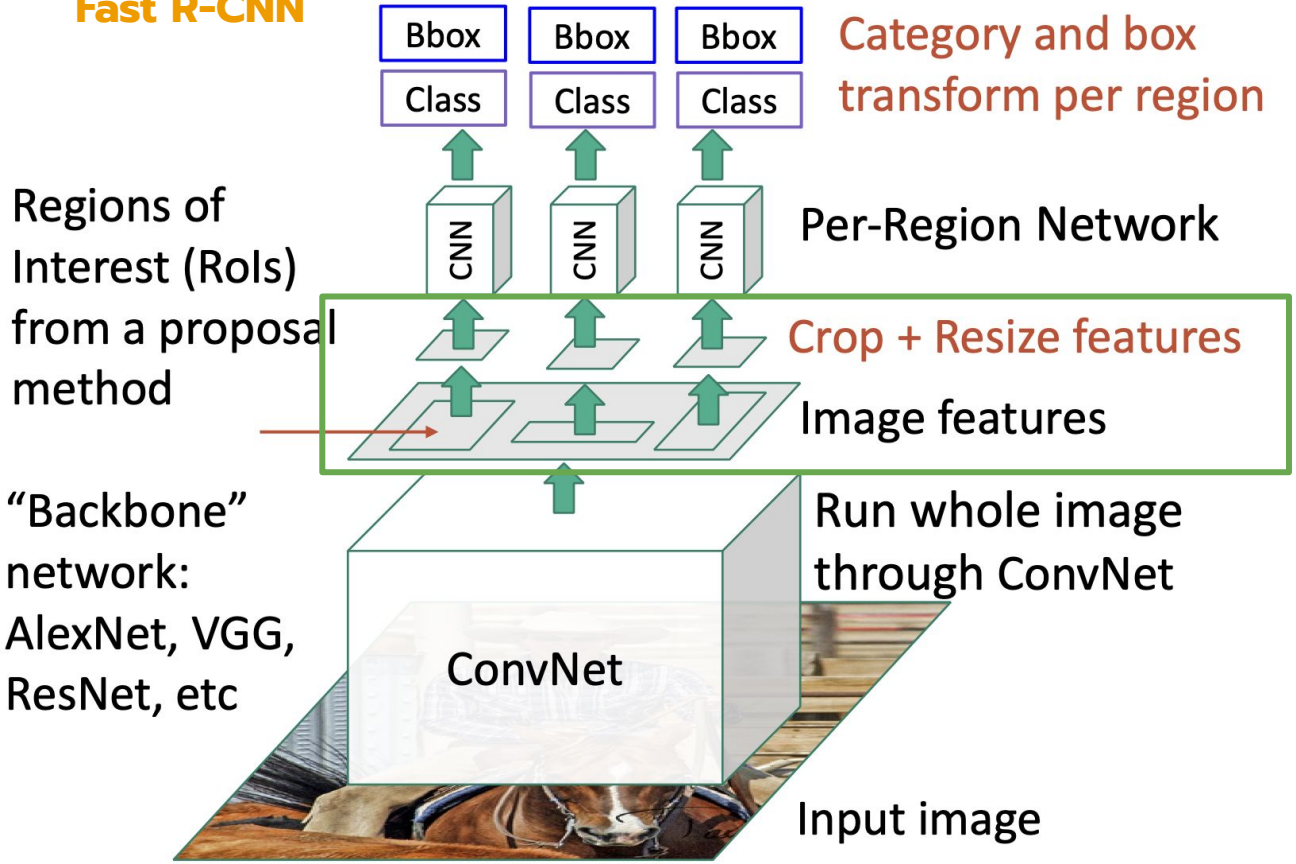


Example: For ResNet, last stage is used as per-region network; the rest of the network is used as backbone

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN



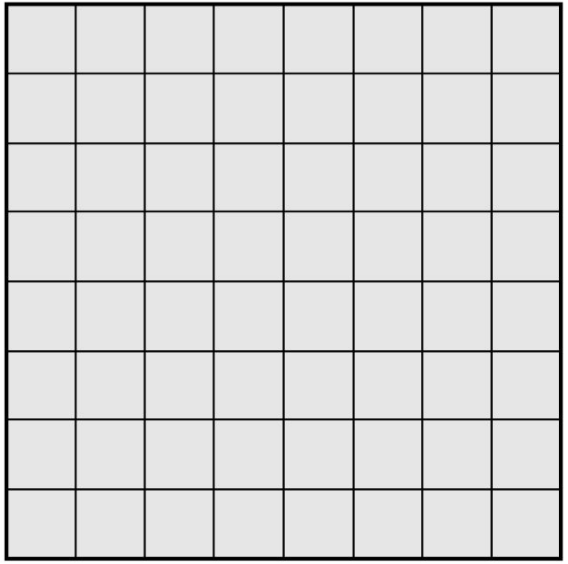
We need to **backprop** into the weights of the **backbone net** as well, so we need to crop these features in a way that is **differentiable**.

How to crop feature?

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

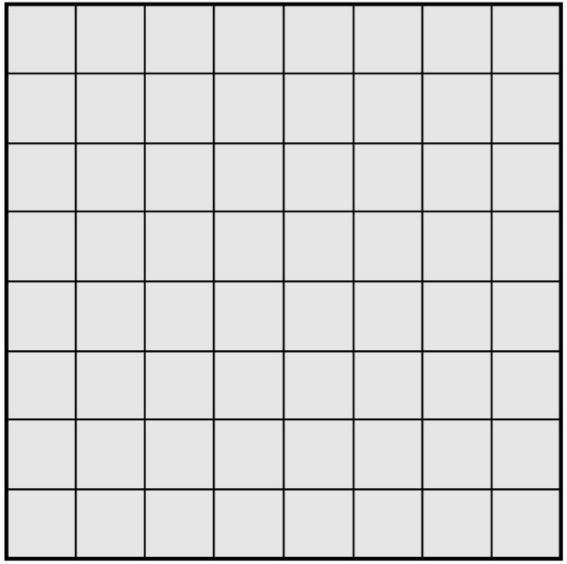
Recall Receptive Fields



Input Image: 8 x 8

Every position in the output feature map depends on a 3x3 receptive field in the input

3x3 Conv
Stride 1, pad 1

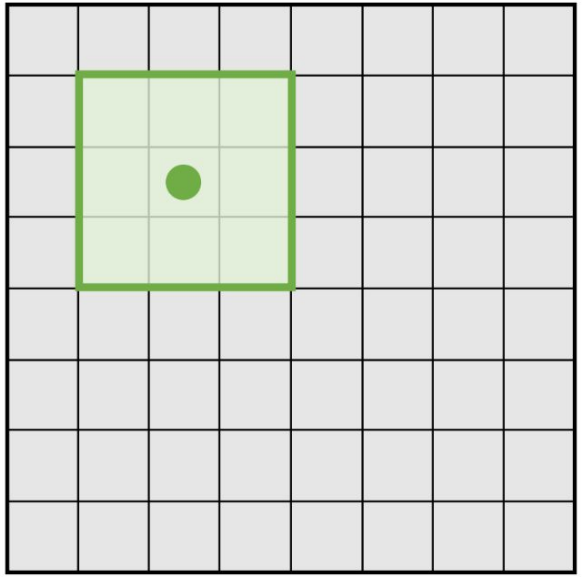


Output Image: 8 x 8

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

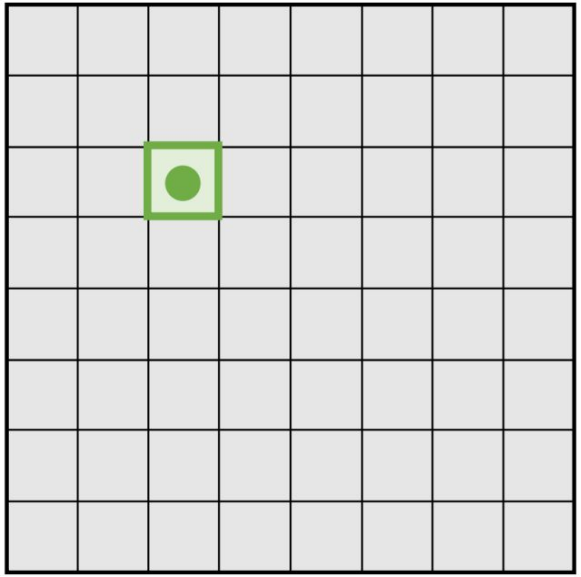
Recall Receptive Fields



Input Image: 8 x 8

Every position in the output feature map depends on a 3x3 receptive field in the input

3x3 Conv
Stride 1, pad 1

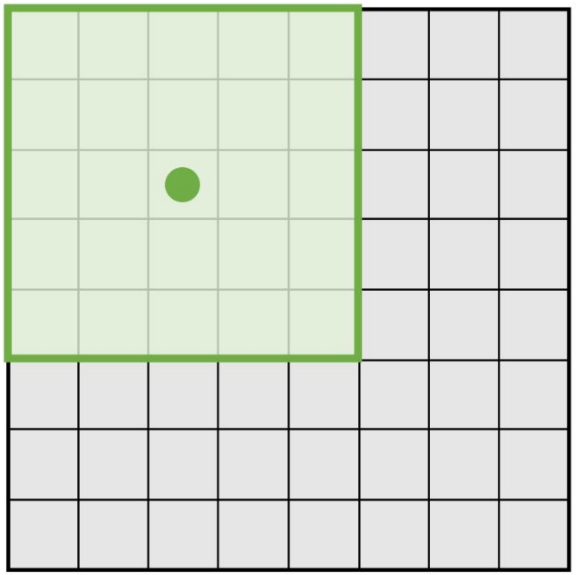


Output Image: 8 x 8

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Recall Receptive Fields

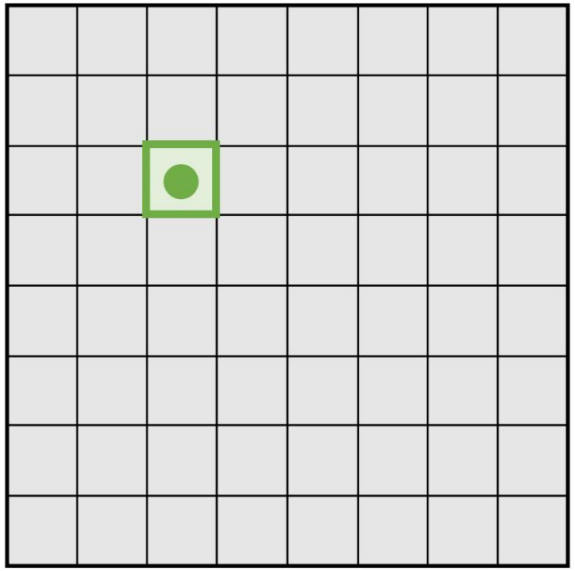


Input Image: 8 x 8

Every position in the output feature map depends on a 5x5 receptive field in the input

3x3 Conv
Stride 1, pad 1

3x3 Conv
Stride 1, pad 1

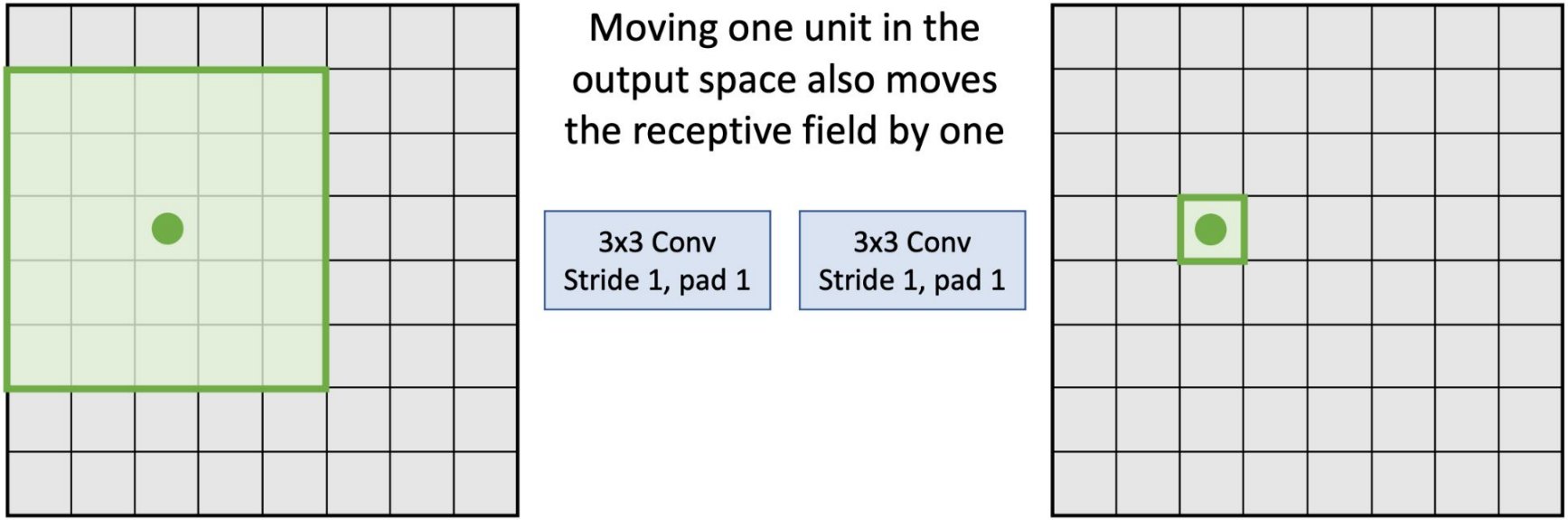


Output Image: 8 x 8

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Recall Receptive Fields



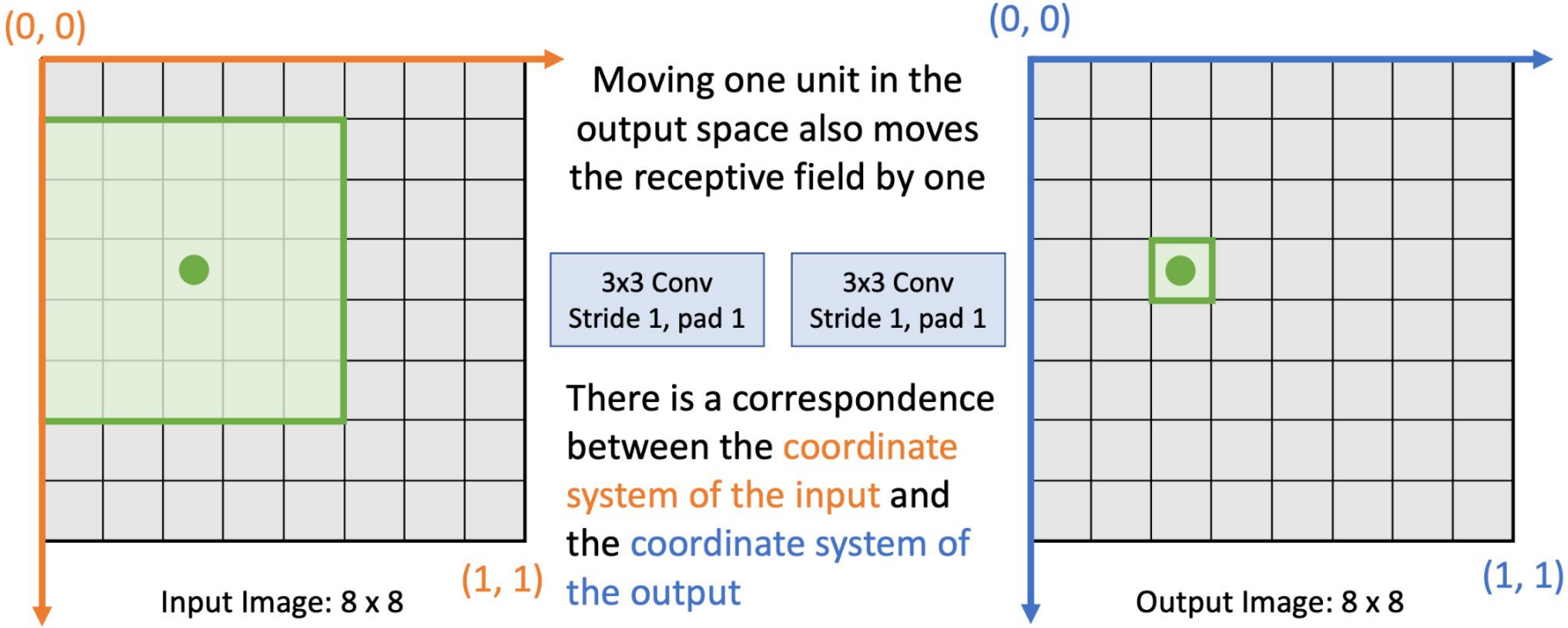
Input Image: 8 x 8

Output Image: 8 x 8

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

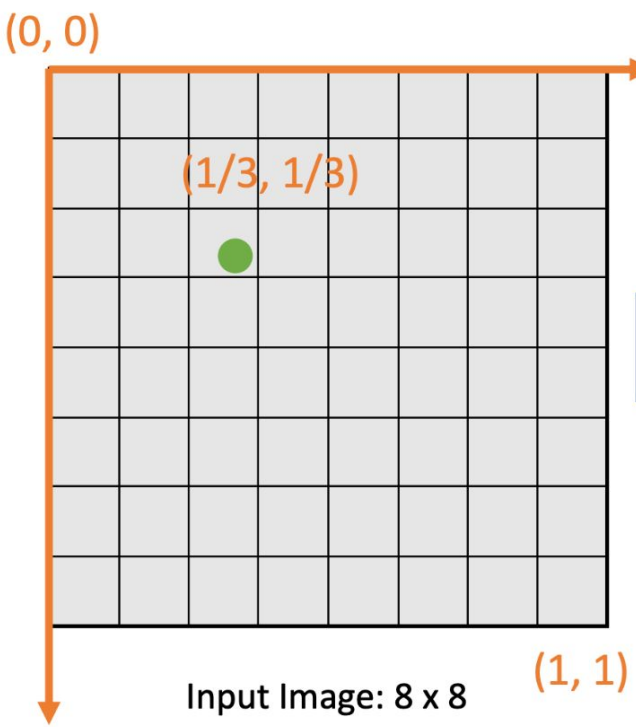
Recall Receptive Fields



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Projecting Points

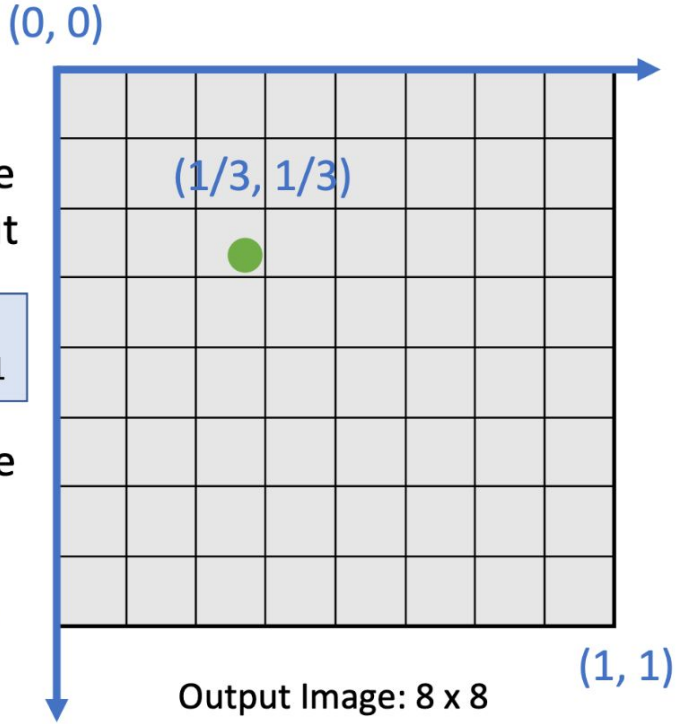


We can align arbitrary points between coordinate system of input and output

3x3 Conv
Stride 1, pad 1

3x3 Conv
Stride 1, pad 1

There is a correspondence between the coordinate system of the input and the coordinate system of the output

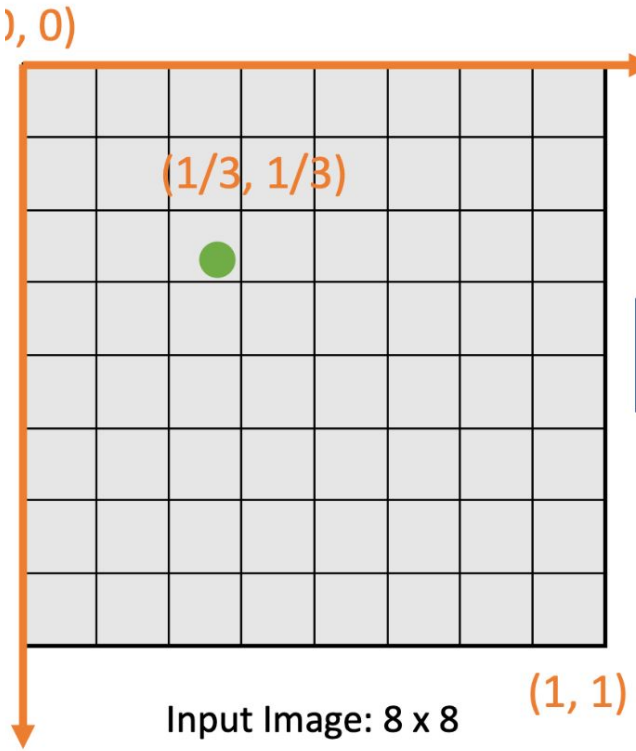


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Projecting Points

Same logic holds for more complicated CNNs, even if spatial resolution of input and output are different

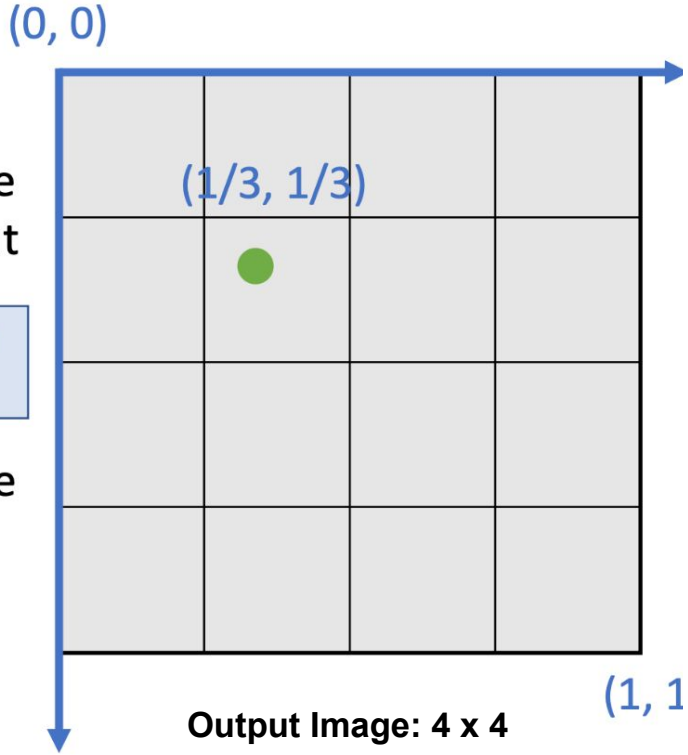


We can align arbitrary points between coordinate system of input and output

3x3 Conv
Stride 1, pad 1

2x2 MaxPool
Stride 2

There is a correspondence between the coordinate system of the input and the coordinate system of the output

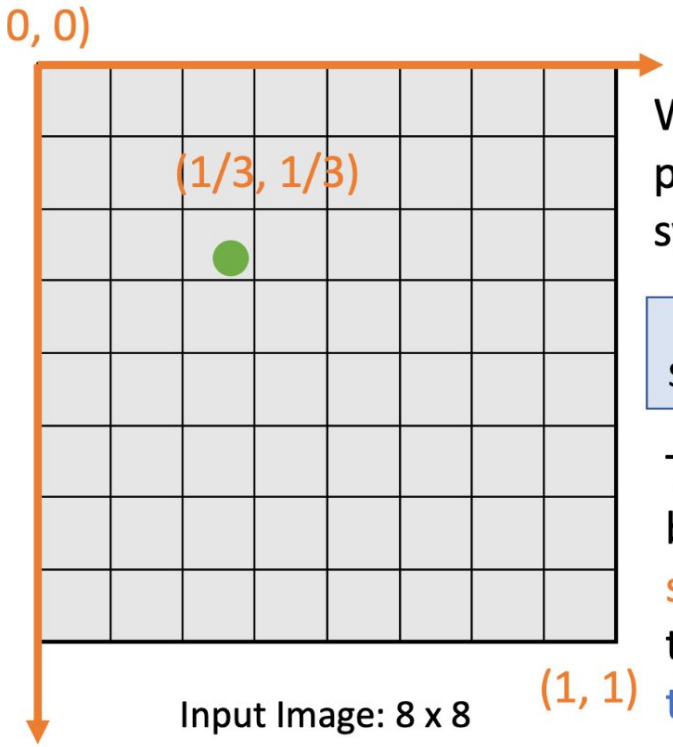


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Projecting Points

Same logic holds for more complicated CNNs, even if spatial resolution of input and output are different

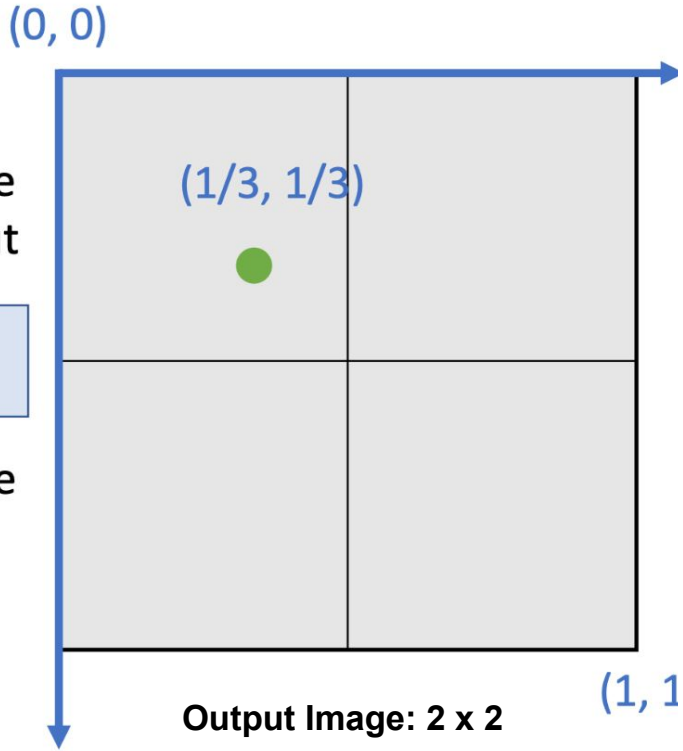


We can align arbitrary points between coordinate system of input and output

3x3 Conv
Stride 1, pad 1

4x4 MaxPool
Stride 4

There is a correspondence between the coordinate system of the input and the coordinate system of the output

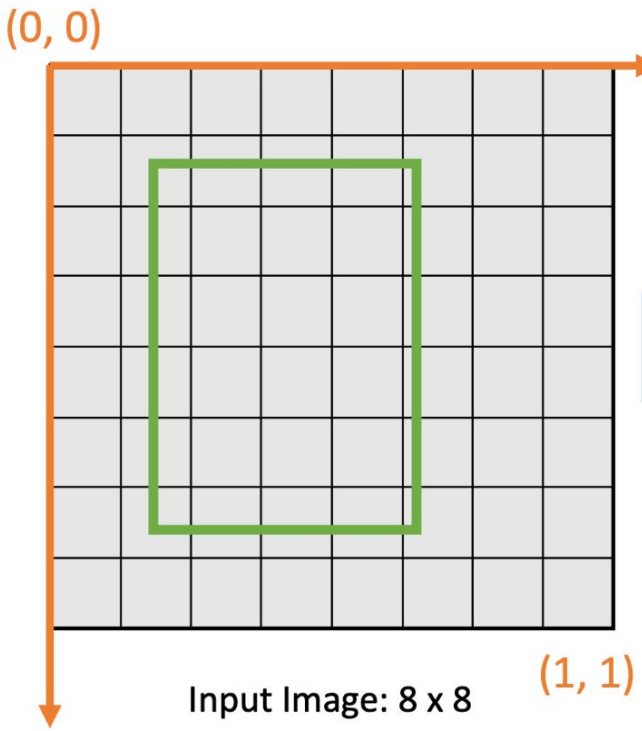


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Projecting Boxes

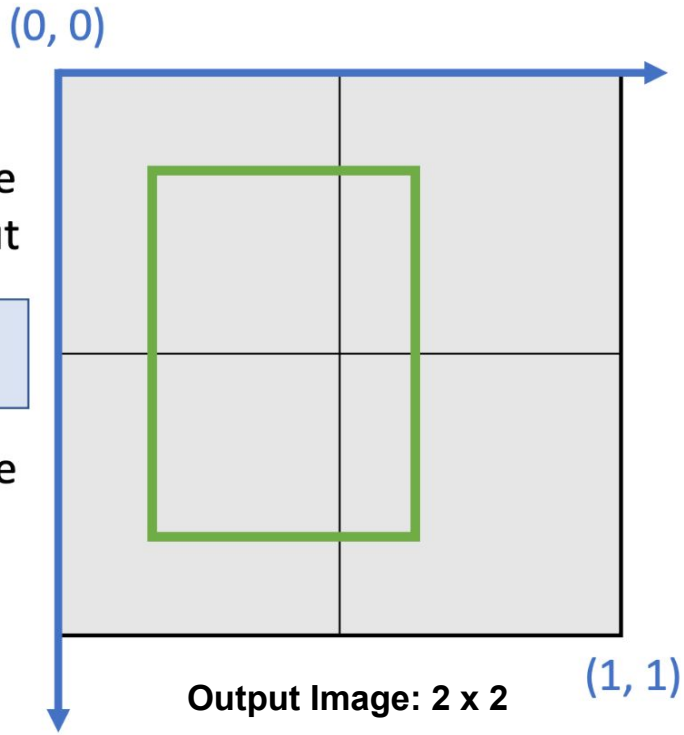
We can use this idea to project **bounding boxes** between an input image and a feature map



We can align arbitrary points between coordinate system of input and output

3x3 Conv Stride 1, pad 1	4x4 MaxPool Stride 4

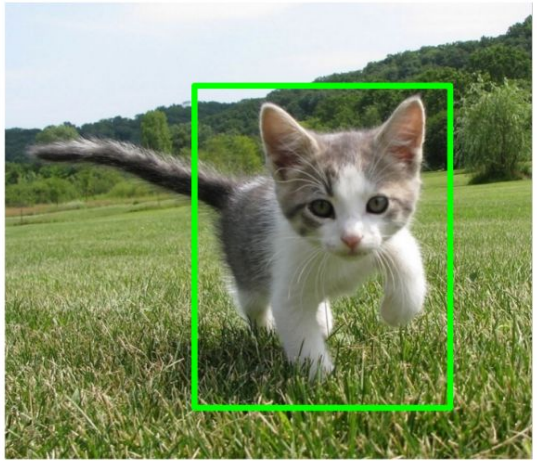
There is a correspondence between the coordinate system of the input and the coordinate system of the output



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Pool

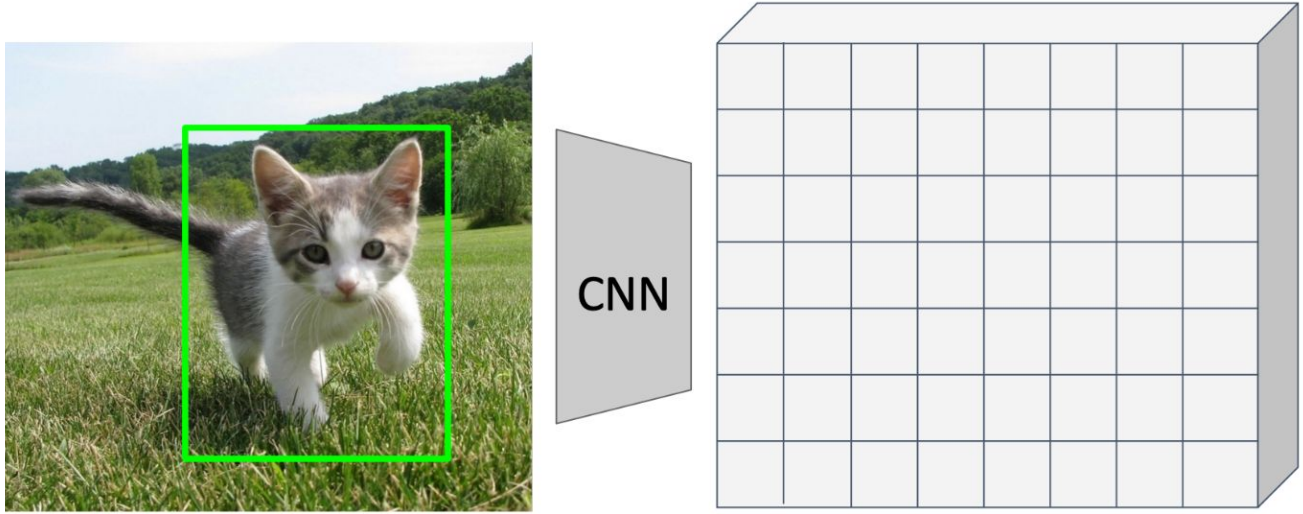


Input Image
(e.g. 3 x 640 x 480)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Pool



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

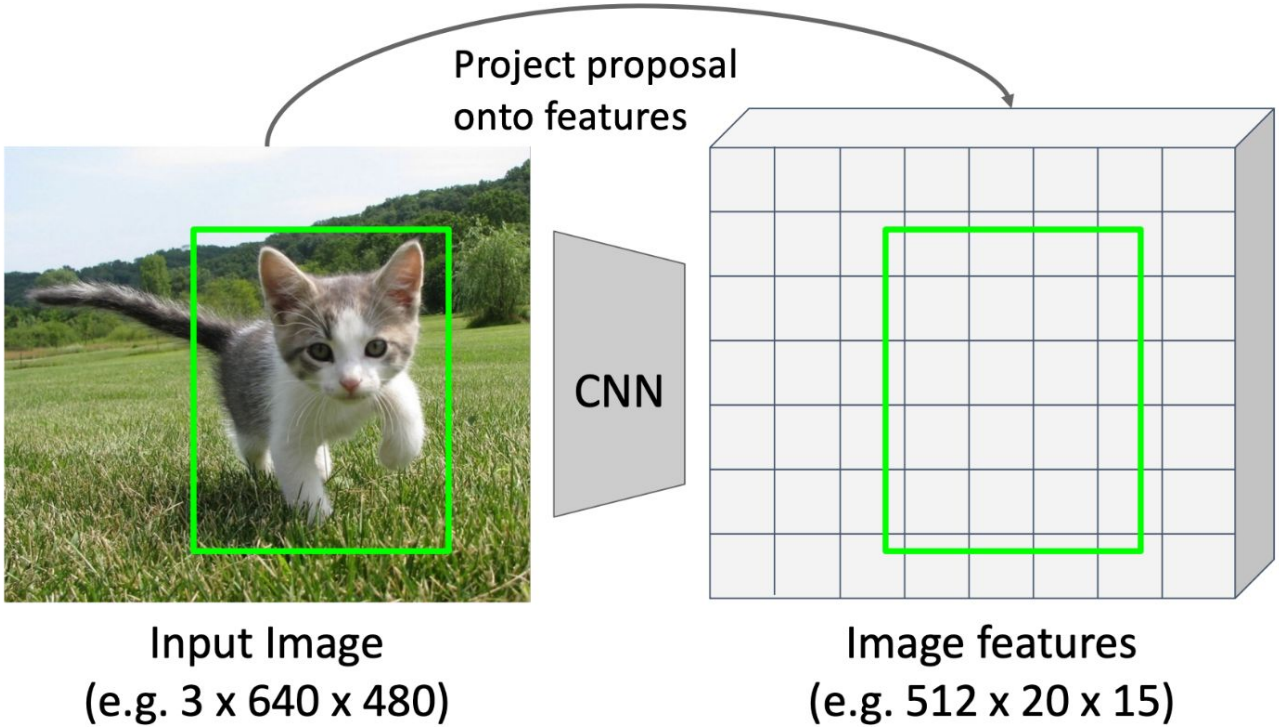
Image features
(e.g. 512 x 20 x 15)

Want features for the box of a fixed size (2x2 in this example, 7x7 or 14x14 in practice)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Pool

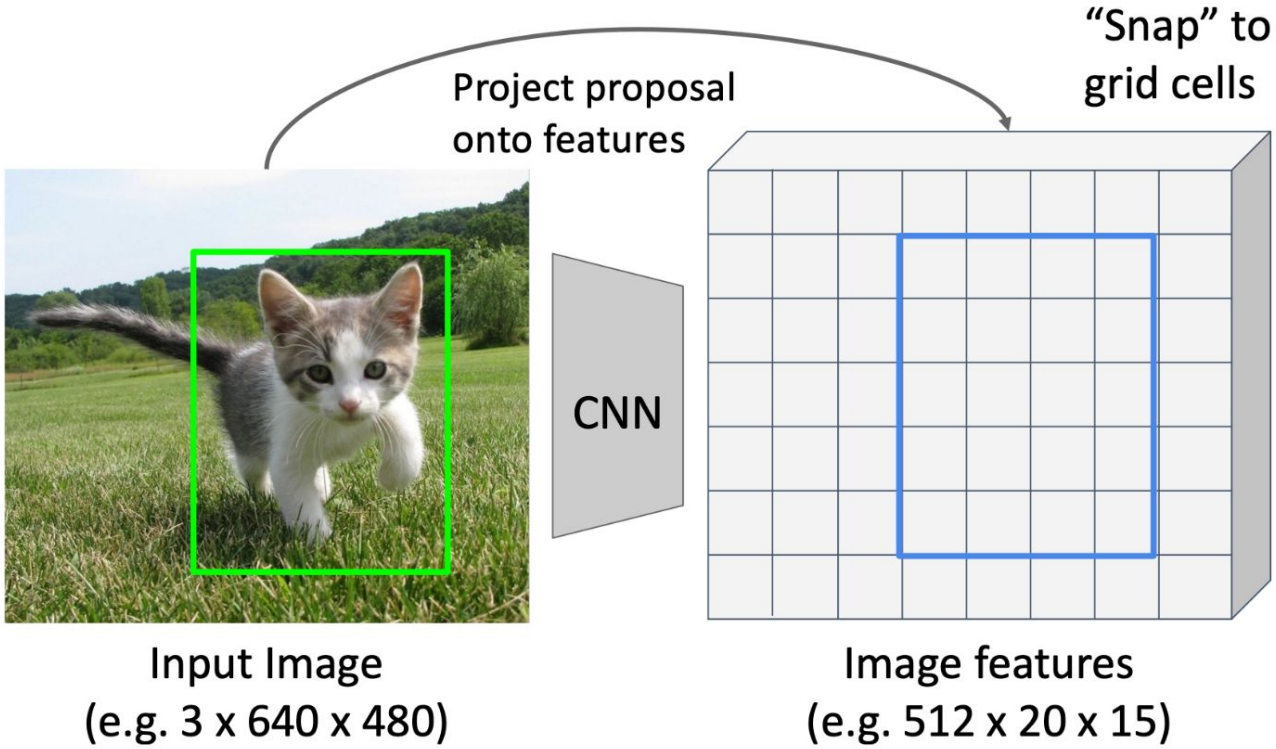


Want features for the box of a fixed size (2x2 in this example, 7x7 or 14x14 in practice)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Pool

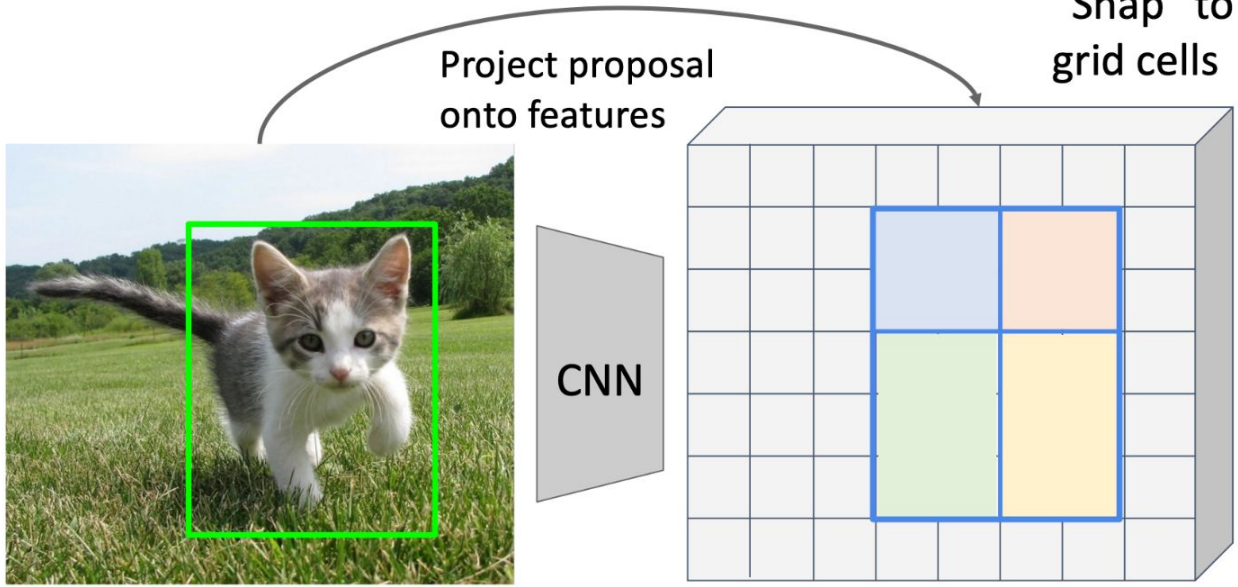


Want features for the box of a fixed size (2x2 in this example, 7x7 or 14x14 in practice)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Pool



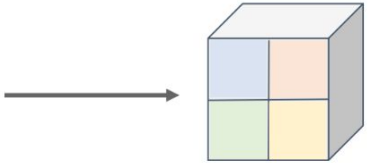
Input Image
(e.g. 3 x 640 x 480)

Image features
(e.g. 512 x 20 x 15)

"Snap" to
grid cells

Divide into 2x2
grid of (roughly)
equal subregions

Max-pool within
each subregion



Region features
(here 512 x 2 x 2;
In practice 512x7x7)

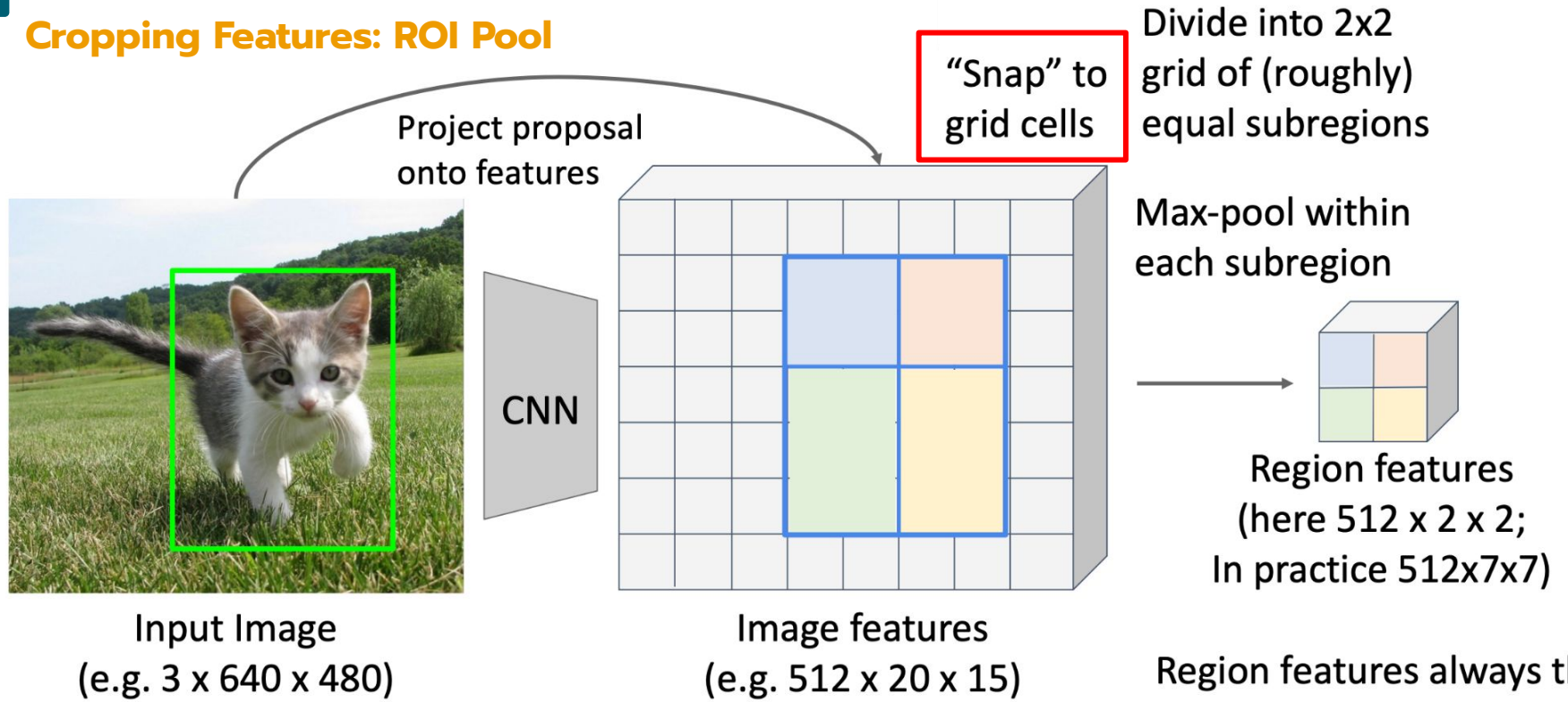
Region features always the
same size even if input
regions have different sizes!

Girshick, "Fast R-CNN", ICCV 2015.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Pool



Problem: Slight misalignment due to snapping; different-sized subregions is weird

Region features always the same size even if input regions have different sizes!

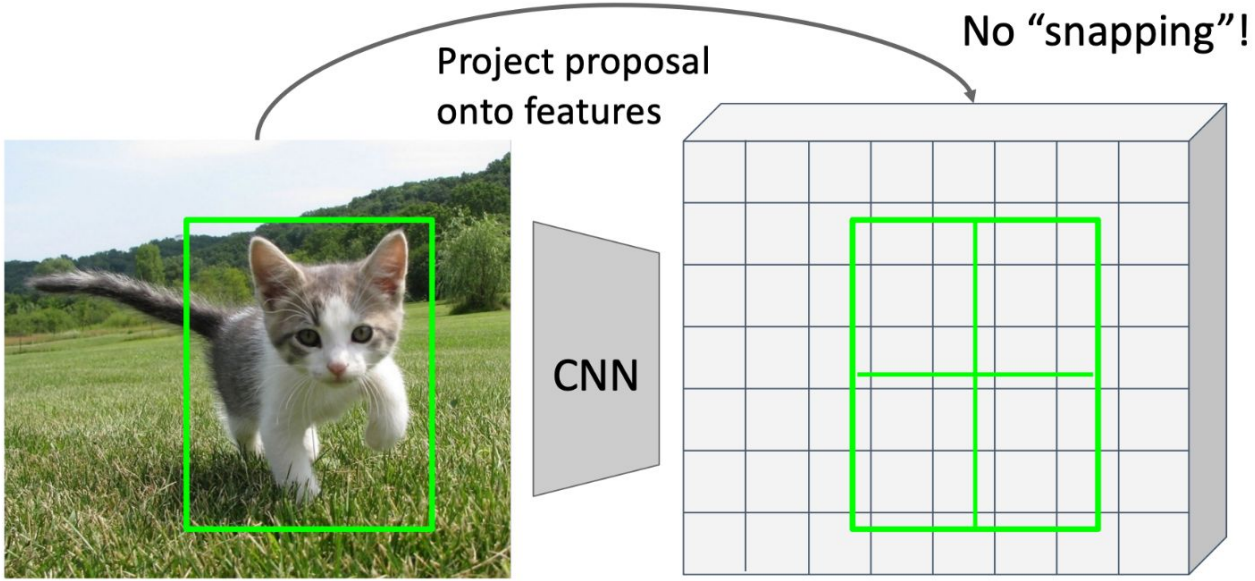
Girshick, "Fast R-CNN", ICCV 2015.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Align

Divide into equal-sized subregions
(may not be aligned to grid!)



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

Image features
(e.g. 512 x 20 x 15)

Want features for the
box of a fixed size
(2x2 in this example,
7x7 or 14x14 in practice)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

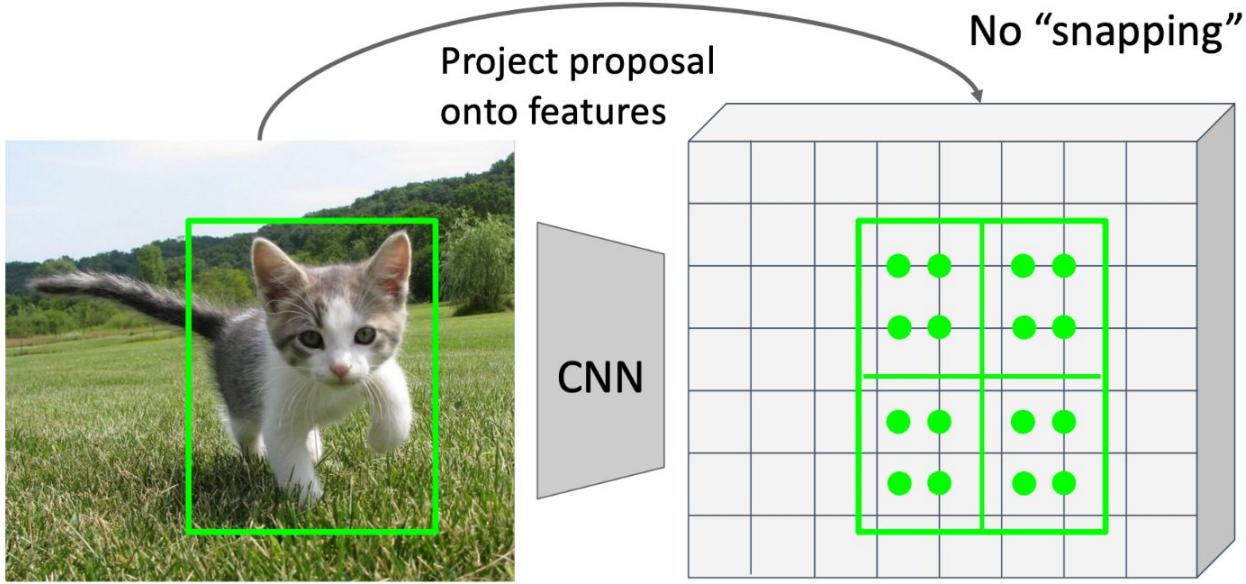
Object Detection

Cropping Features: ROI Align

Divide into equal-sized subregions (may not be aligned to grid!)

No “snapping”!

Sample features at regularly-spaced points in each subregion using **bilinear interpolation**



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

Image features
(e.g. 512 x 20 x 15)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

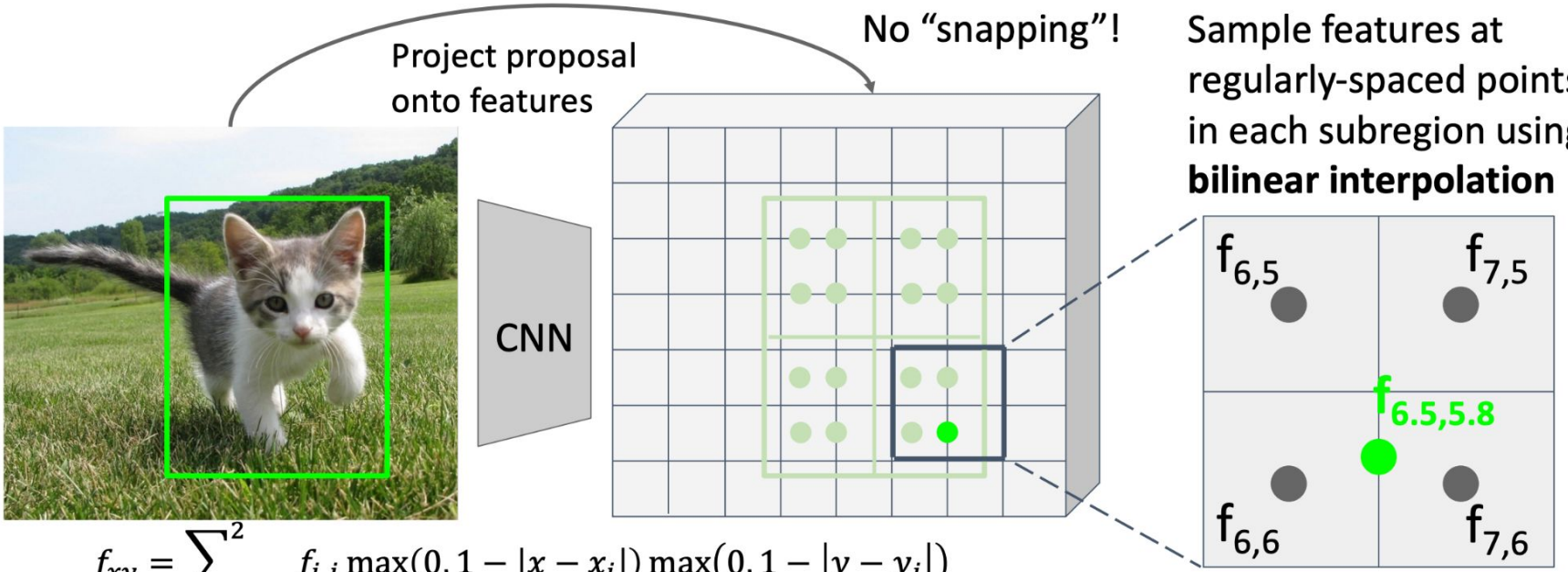
Object Detection

Cropping Features: ROI Align

Divide into equal-sized subregions (may not be aligned to grid!)

No "snapping"!

Sample features at regularly-spaced points in each subregion using **bilinear interpolation**



$$f_{xy} = \sum_{i,j=1}^{2} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_j|)$$

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

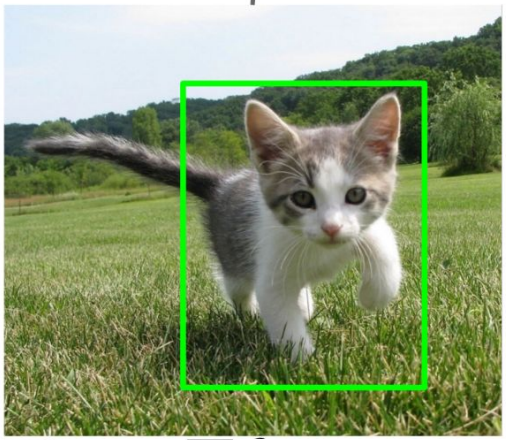
Object Detection

Cropping Features: ROI Align

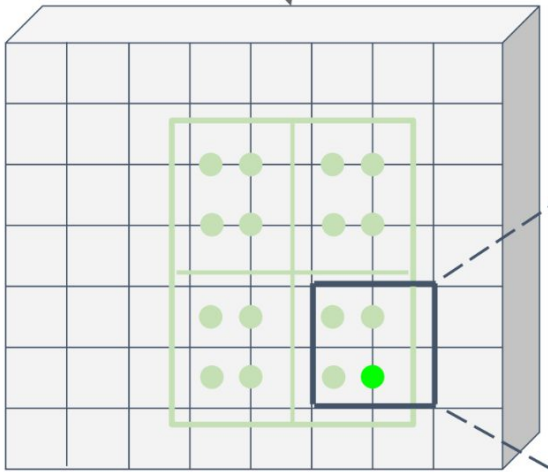
Divide into equal-sized subregions (may not be aligned to grid!)

No "snapping"!

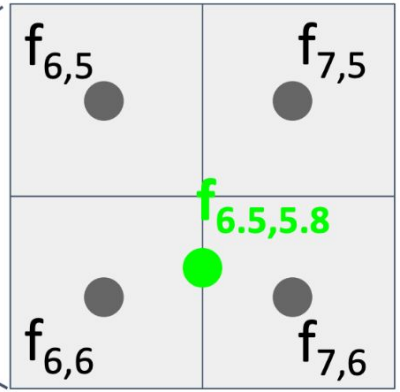
Project proposal onto features



CNN



Sample features at regularly-spaced points in each subregion using **bilinear interpolation**



$$f_{xy} = \sum_{i,j} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_i|)$$

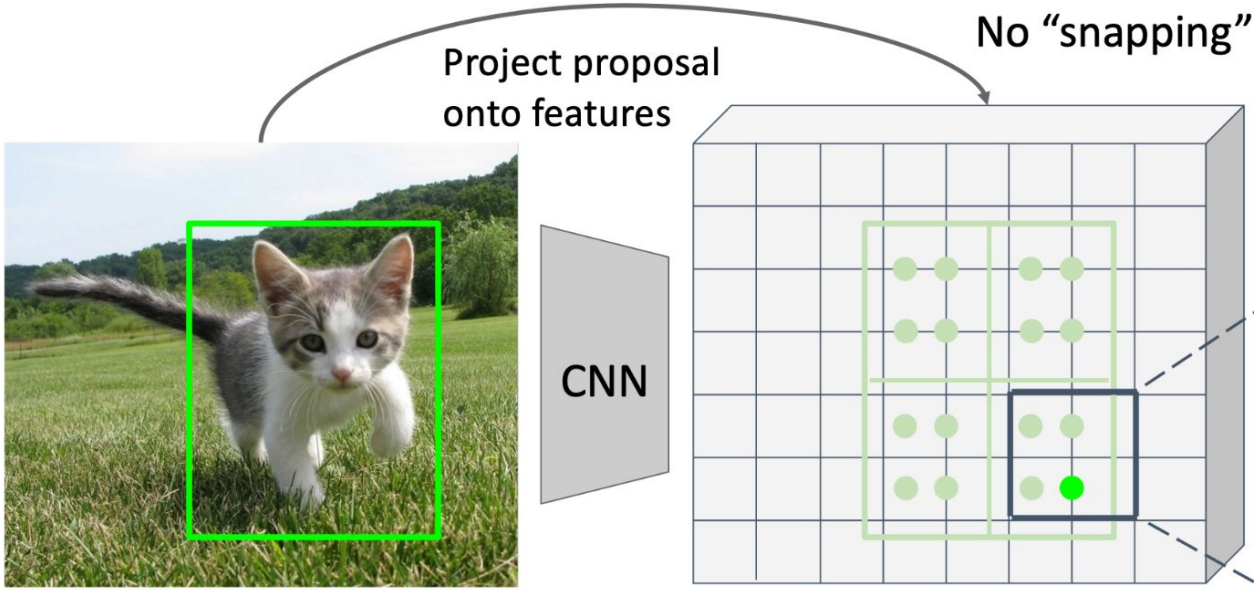
$$f_{6.5,5.8} = (f_{6,5} * 0.5 * 0.2) + (f_{7,5} * 0.5 * 0.2) + (f_{6,6} * 0.5 * 0.8) + (f_{7,6} * 0.5 * 0.8)$$

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

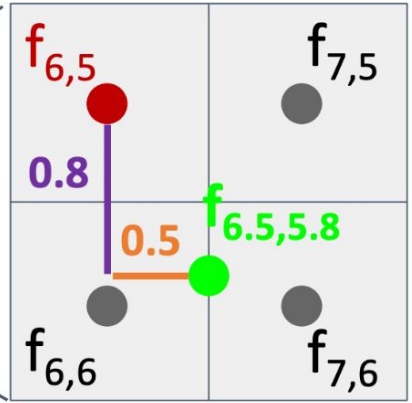
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Align



Sample features at regularly-spaced points in each subregion using **bilinear interpolation**



$$f_{xy} = \sum_{i,j} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_i|)$$

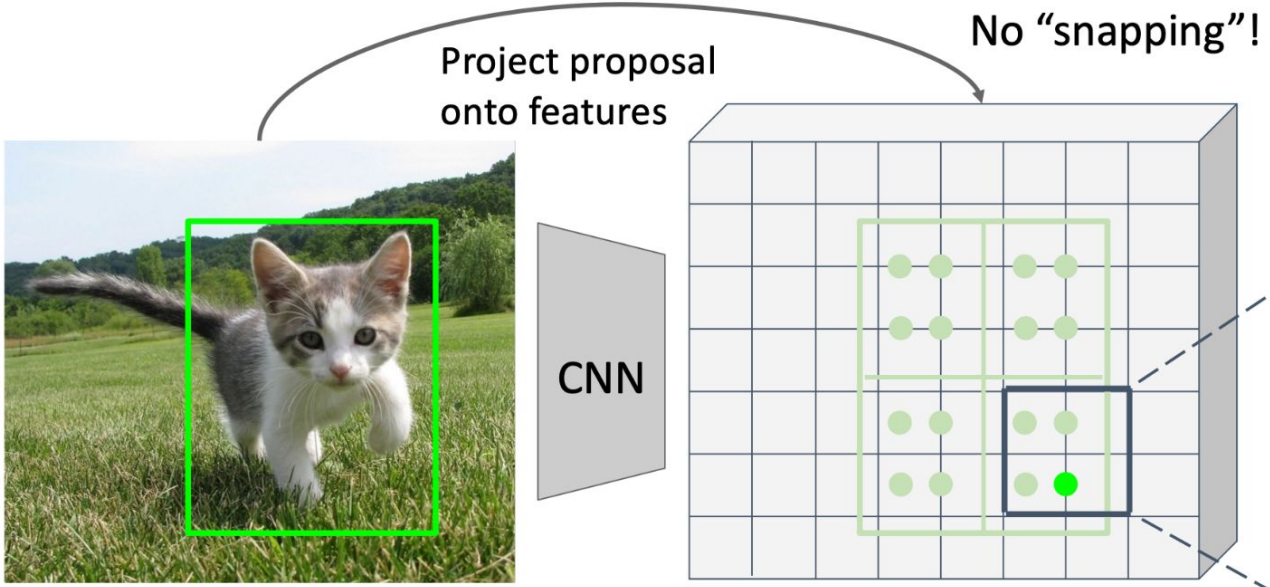
$$f_{6.5,5.8} = (f_{6,5} * 0.5 * 0.2) + (f_{7,5} * 0.5 * 0.2) + (f_{6,6} * 0.5 * 0.8) + (f_{7,6} * 0.5 * 0.8)$$

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

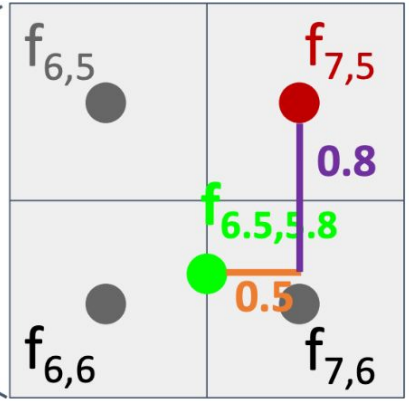
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Align



Sample features at regularly-spaced points in each subregion using **bilinear interpolation**



$$f_{xy} = \sum_{i,j} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_i|)$$

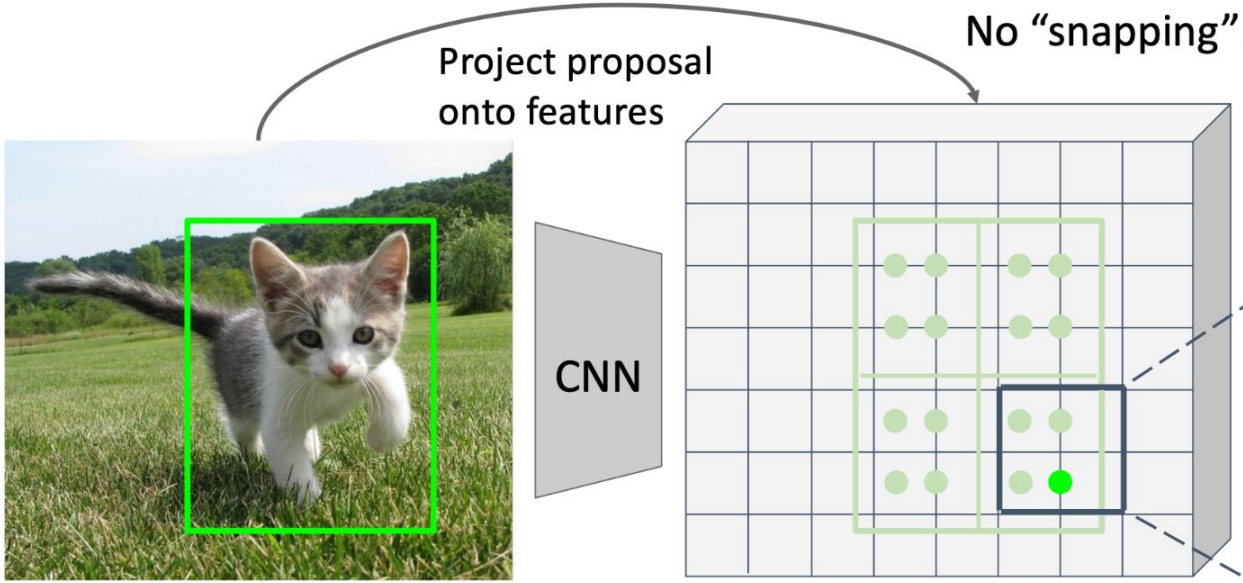
$$f_{6.5,5.8} = (f_{6,5} * 0.5 * 0.2) + (f_{7,5} * 0.5 * 0.2) + (f_{6,6} * 0.5 * 0.8) + (f_{7,6} * 0.5 * 0.8)$$

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

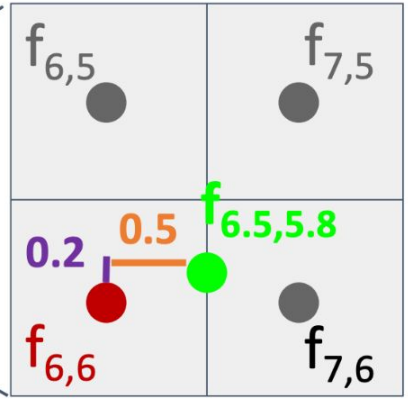
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Align



Sample features at regularly-spaced points in each subregion using **bilinear interpolation**



$$f_{xy} = \sum_{i,j} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_i|)$$

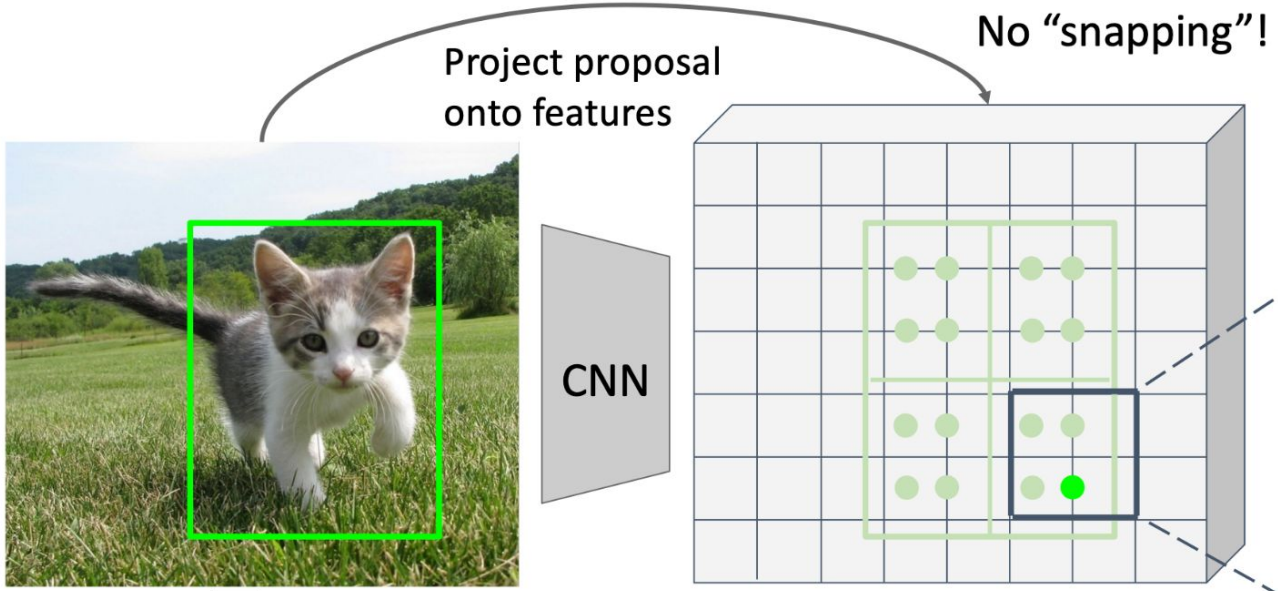
$$f_{6.5,5.8} = (f_{6,5} * 0.5 * 0.2) + (f_{7,5} * 0.5 * 0.2) + (f_{6,6} * 0.5 * 0.8) + (f_{7,6} * 0.5 * 0.8)$$

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

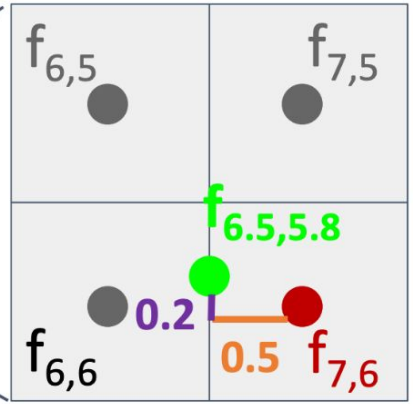
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Align



Sample features at regularly-spaced points in each subregion using **bilinear interpolation**



$$f_{xy} = \sum_{i,j} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_i|)$$

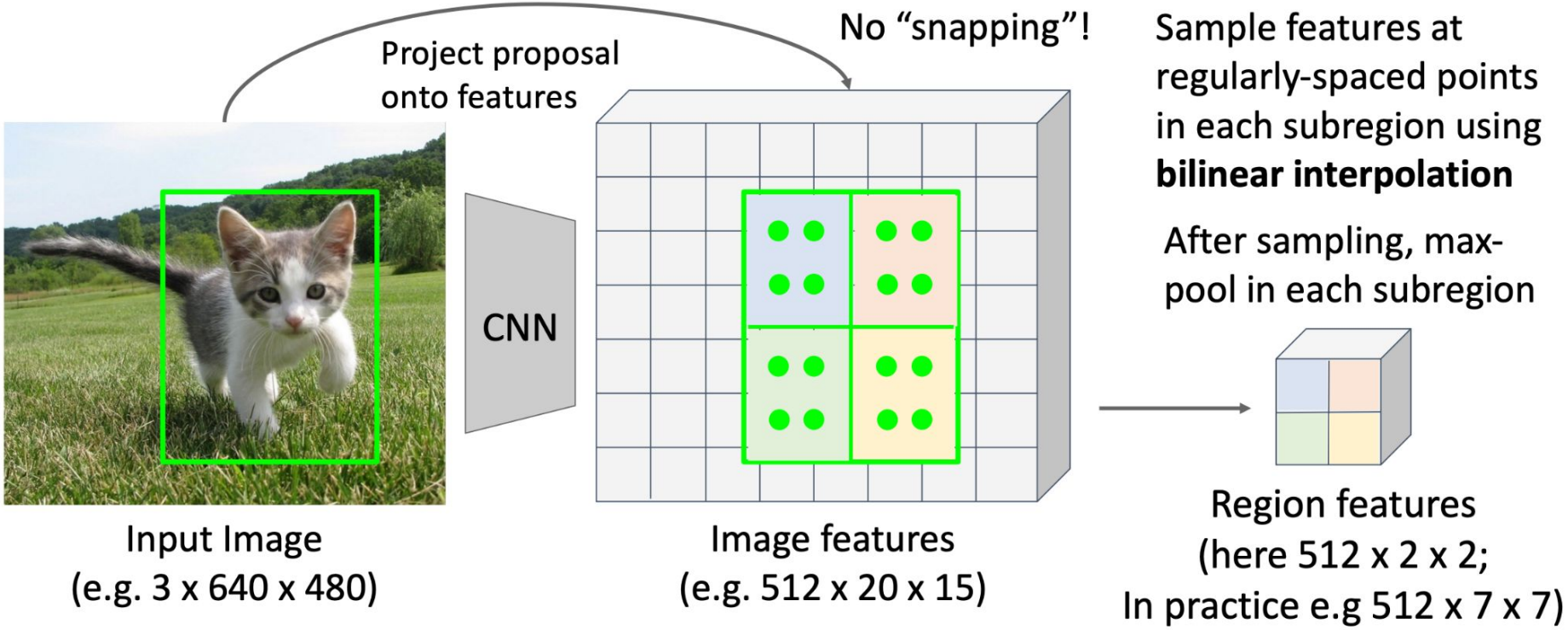
$$f_{6.5,5.8} = (f_{6,5} * 0.5 * 0.2) + (f_{7,5} * 0.5 * 0.2) + (f_{6,6} * 0.5 * 0.8) + (f_{7,6} * 0.5 * 0.8)$$

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Cropping Features: ROI Align

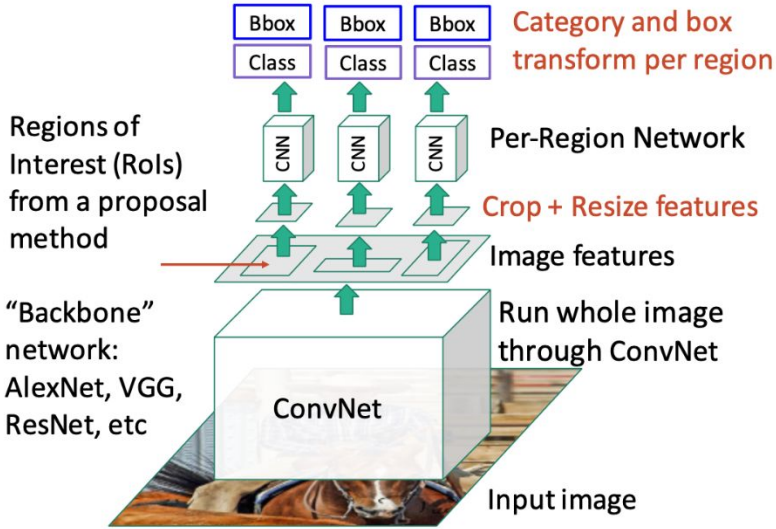


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

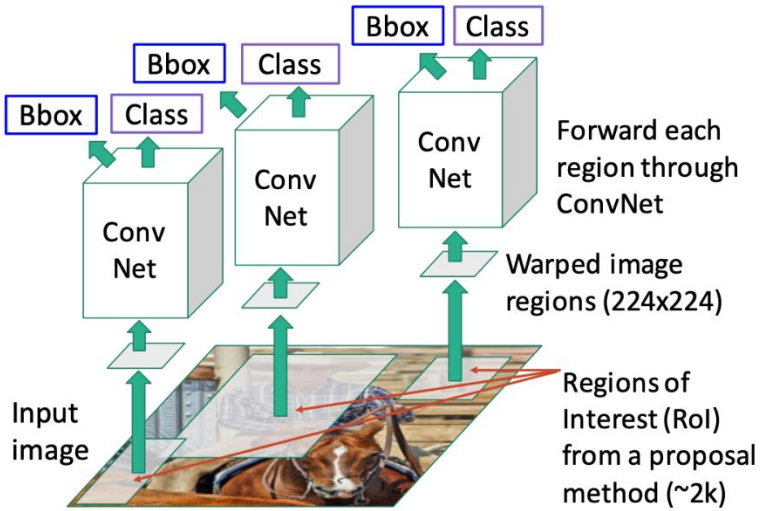
Object Detection

Fast R-CNN vs. "Slow" R-CNN

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



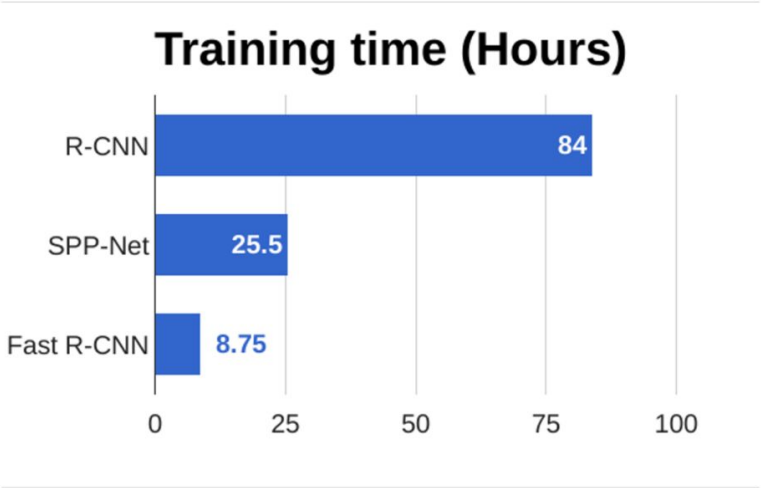
"Slow" R-CNN: Apply differentiable cropping to shared image features

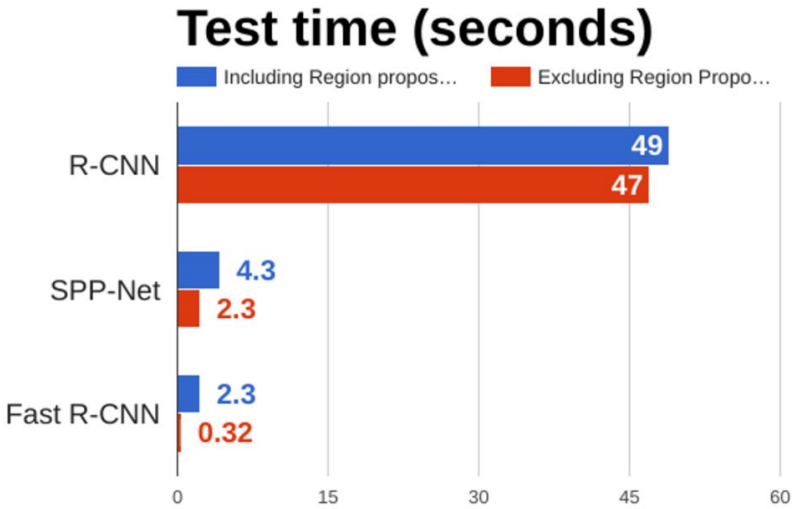


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Fast R-CNN vs. "Slow" R-CNN



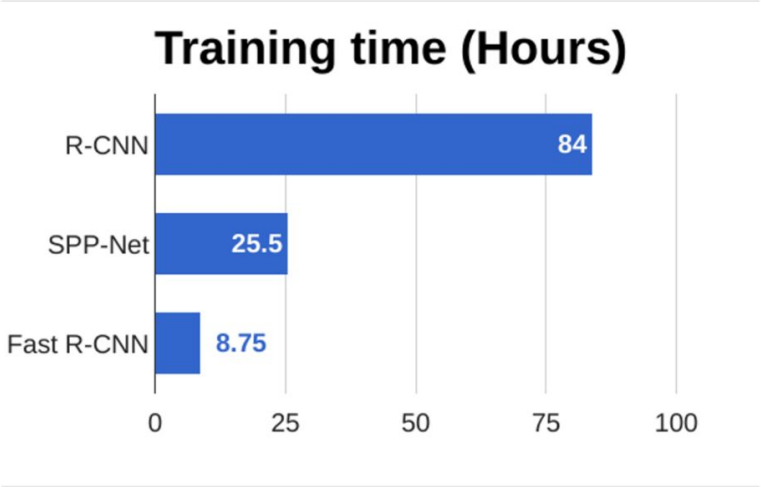


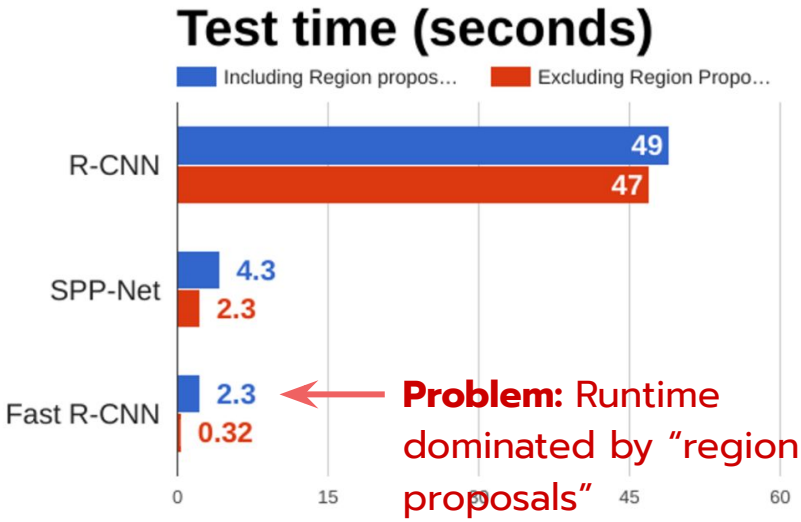
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

Fast R-CNN vs. "Slow" R-CNN





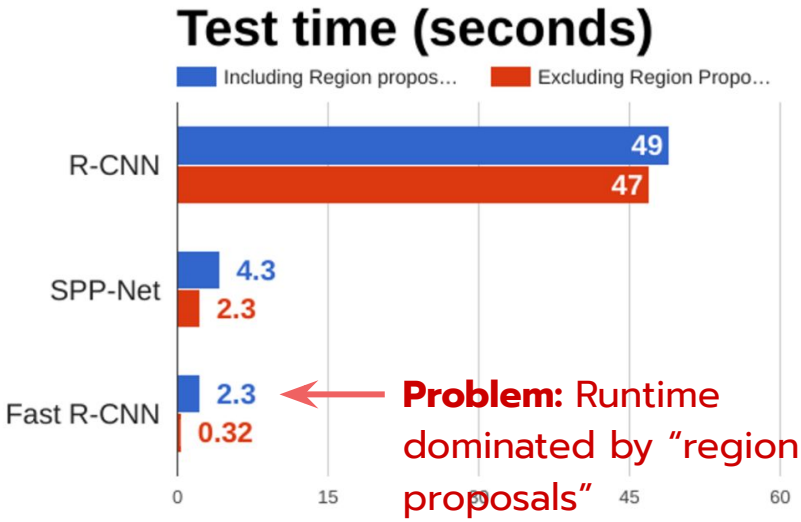
Problem: Runtime dominated by "region proposals"

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

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

Object Detection

Fast R-CNN vs. "Slow" R-CNN



Problem: Runtime dominated by "region proposals"

Recall: "Region proposals" computed by heuristic "selective search" algorithm on CPU -> Let's learn them with a CNN instead.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

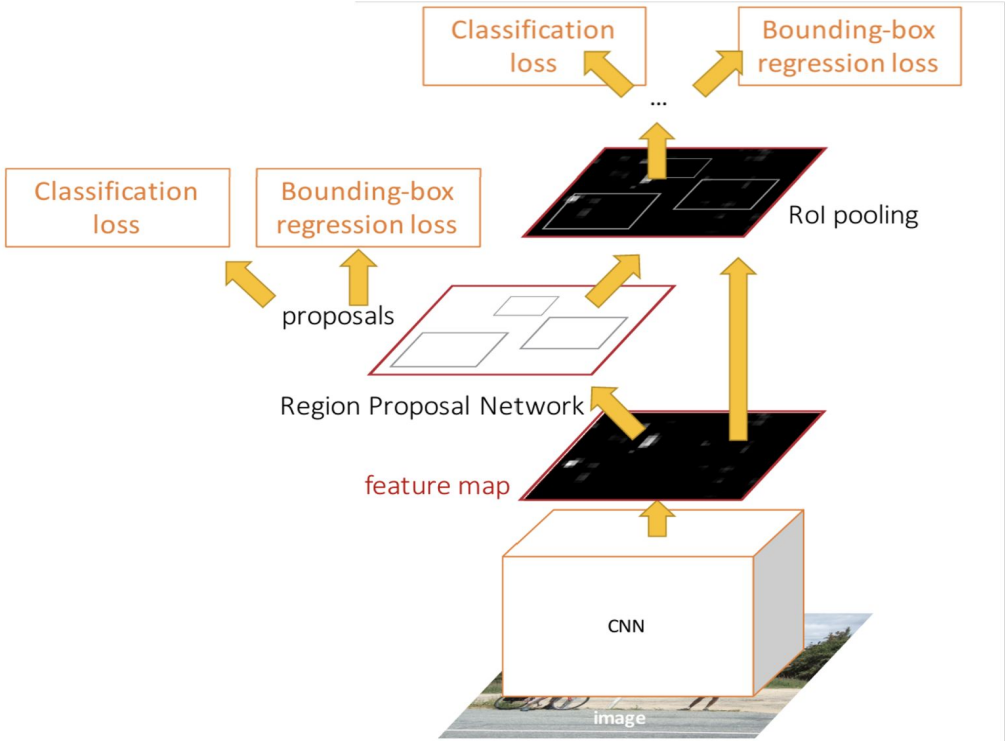
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
He et al, "Spatial pyramid pooling in deep convolutional networks for visual recognition", ECCV 2014
Girshick, "Fast R-CNN", ICCV 2015

Object 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

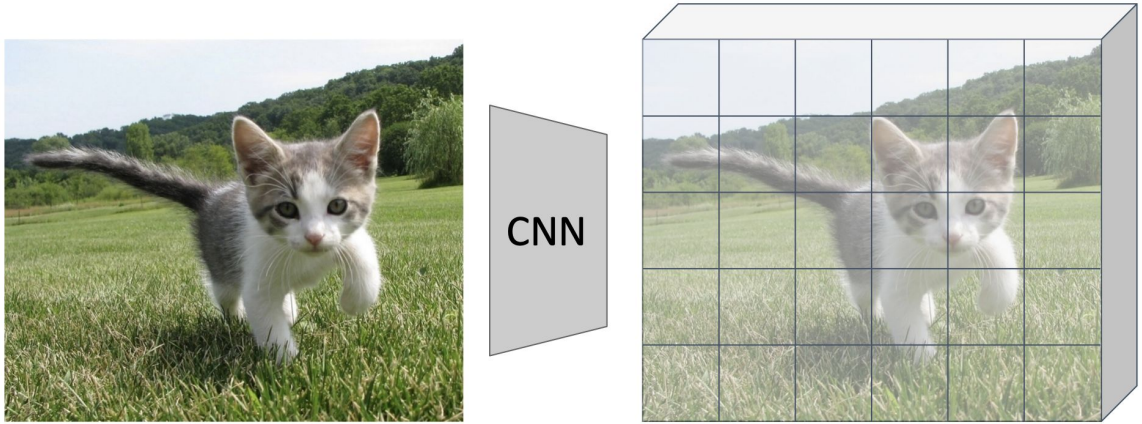


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Faster R-CNN: 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)

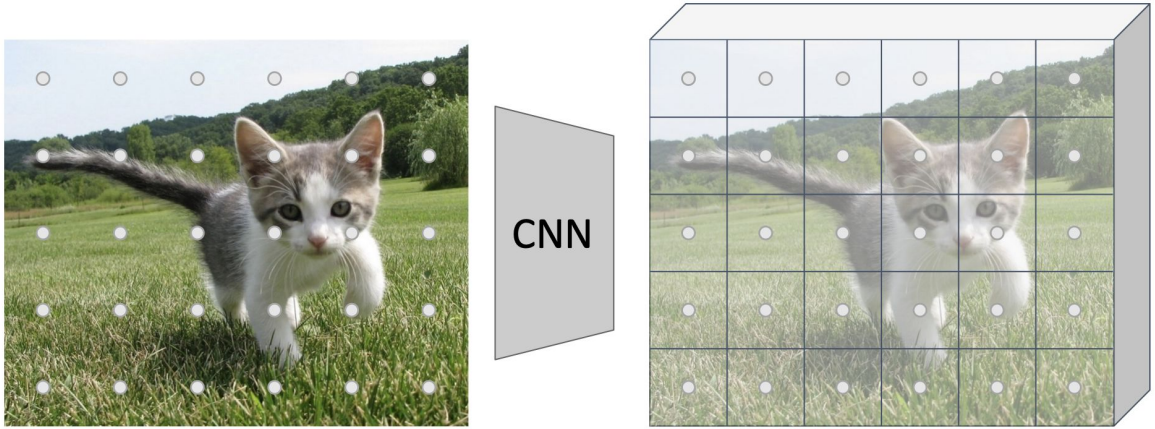
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Faster R-CNN: 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)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

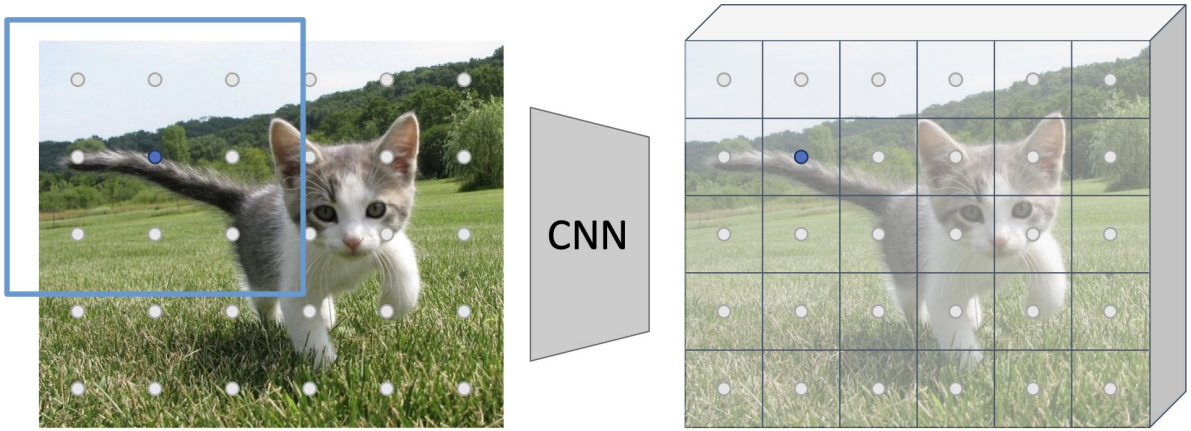
Object Detection

Faster R-CNN: 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)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

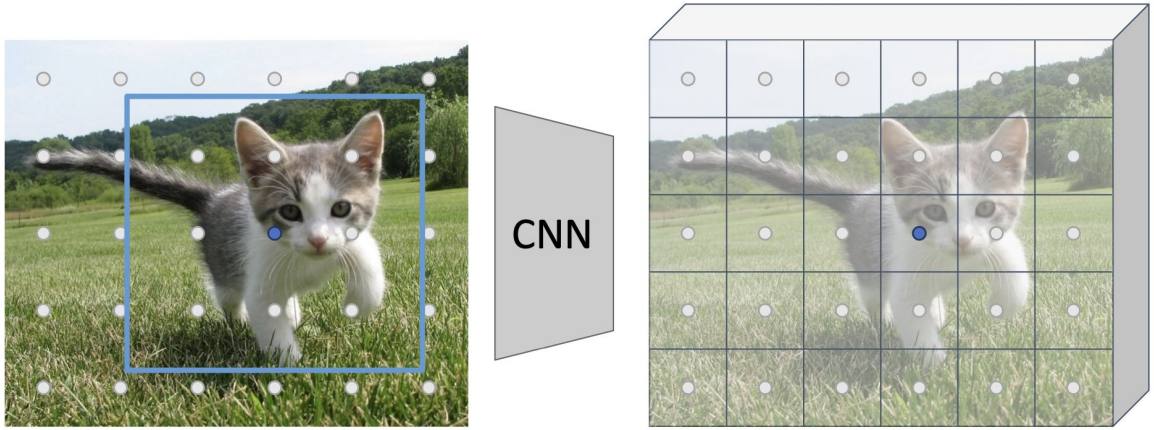
Object Detection

Faster R-CNN: 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)

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

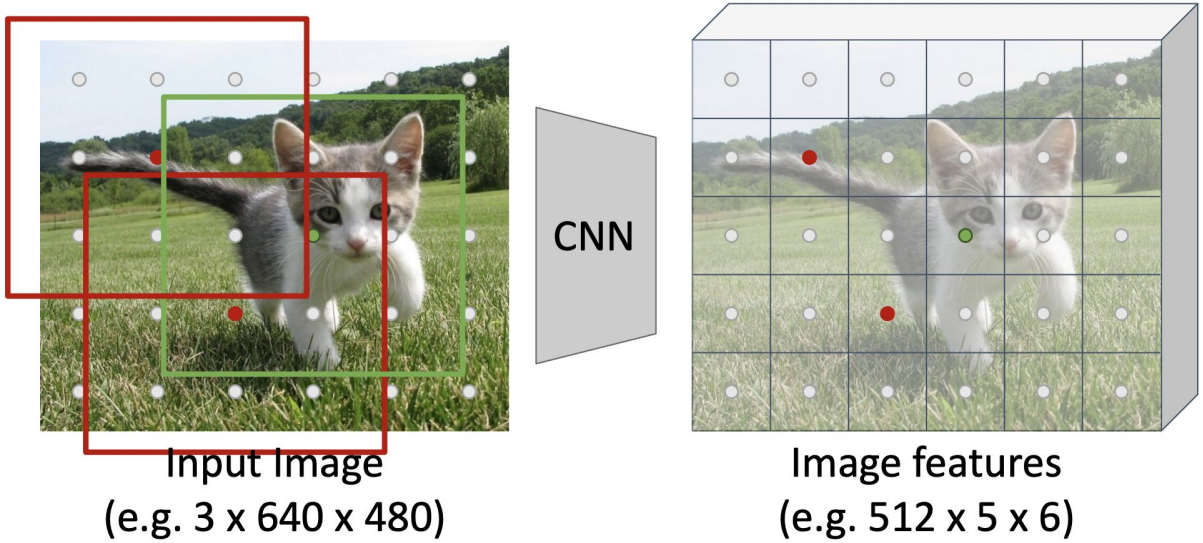
Object Detection

Faster R-CNN: 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



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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

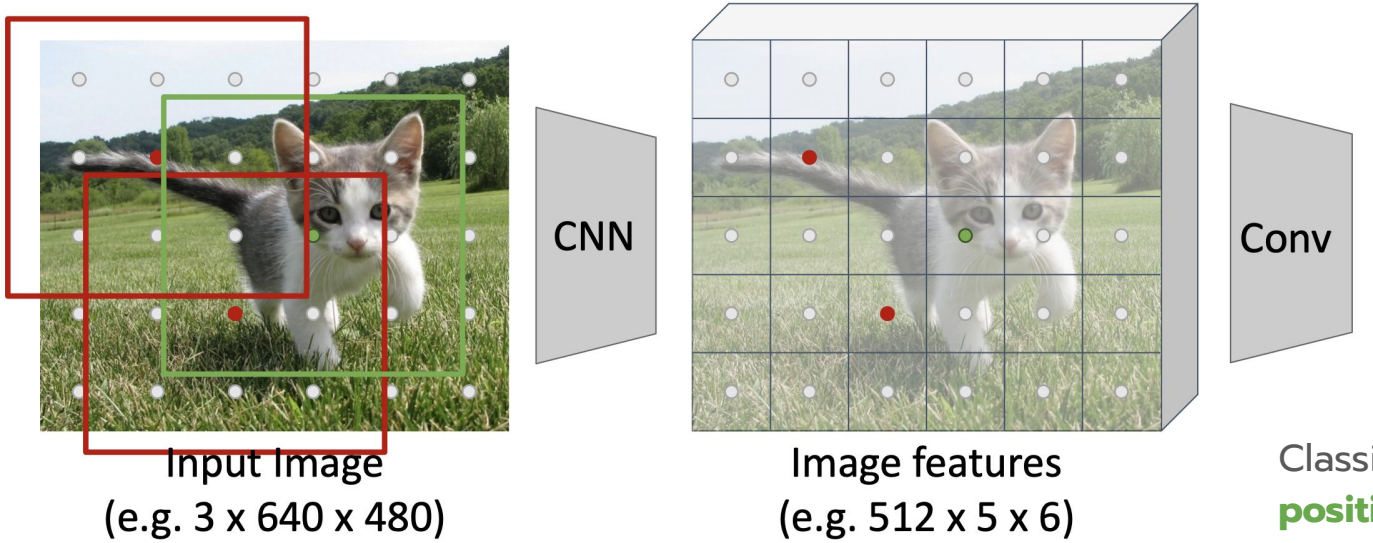
Object Detection

Faster R-CNN: 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

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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

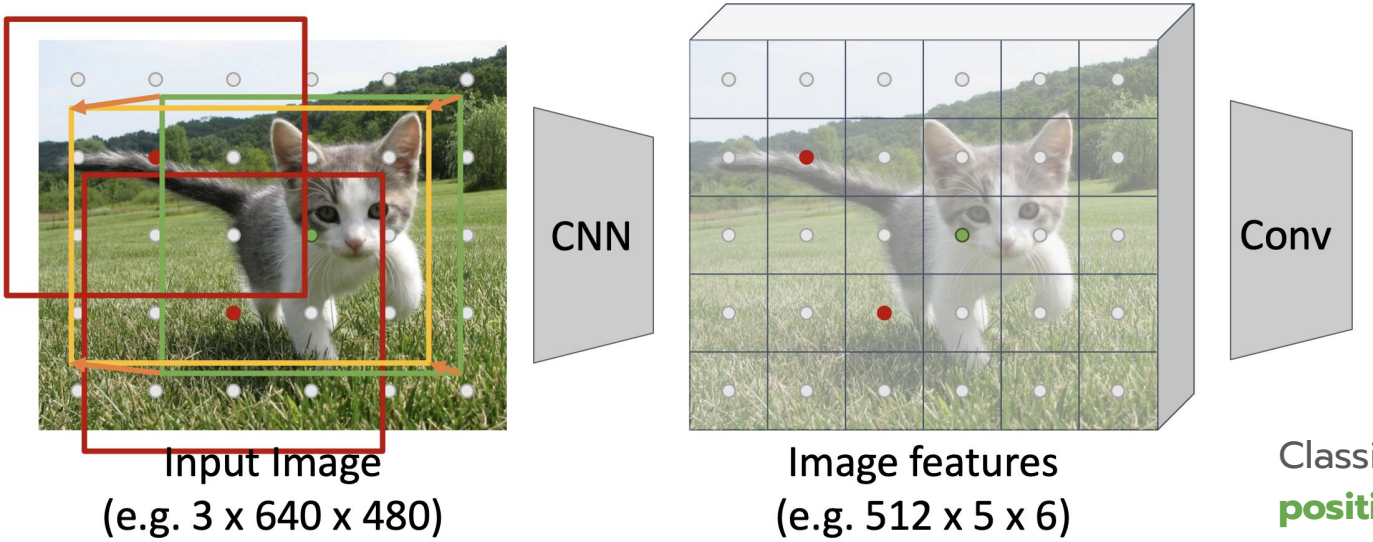
Object Detection

Faster R-CNN: Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input

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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

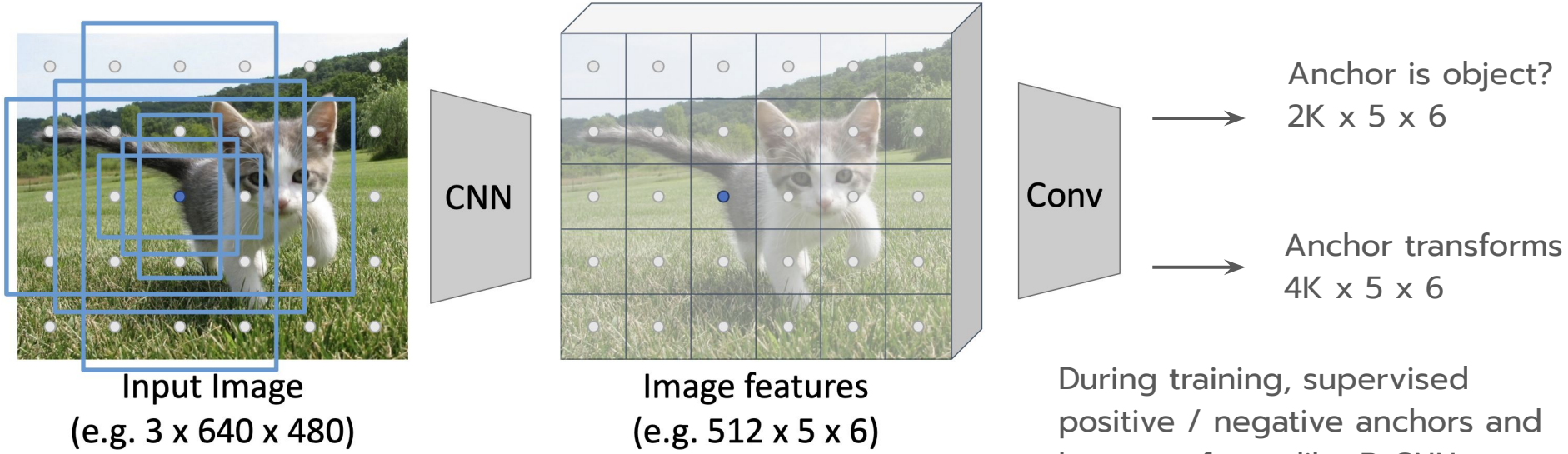
Object Detection

Faster R-CNN: 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



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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

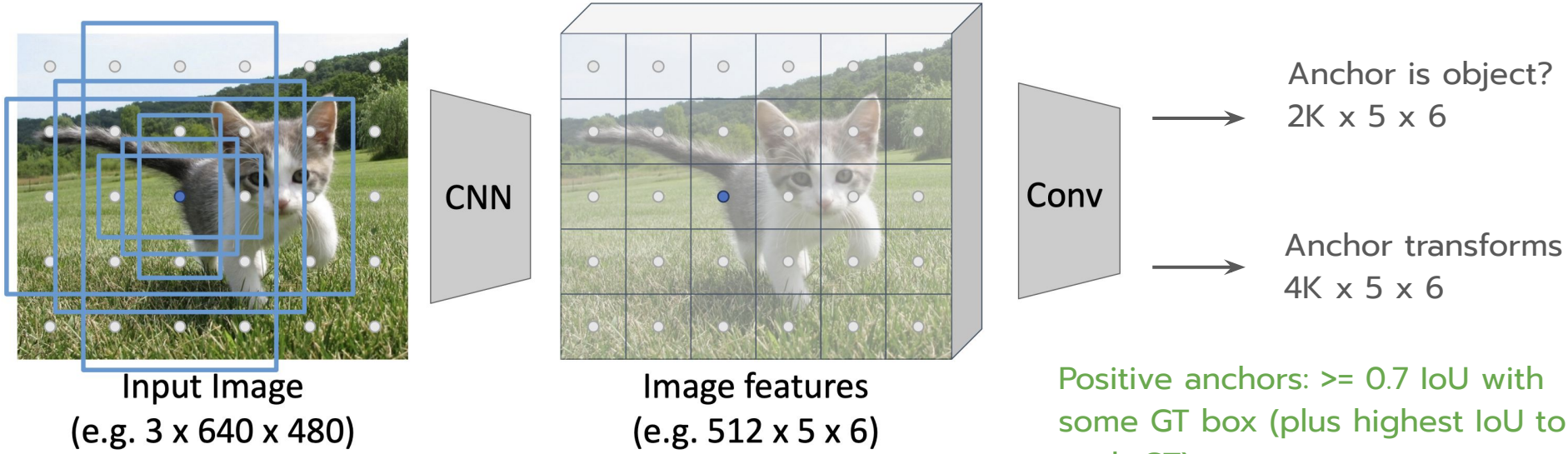
Object Detection

Faster R-CNN: 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



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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

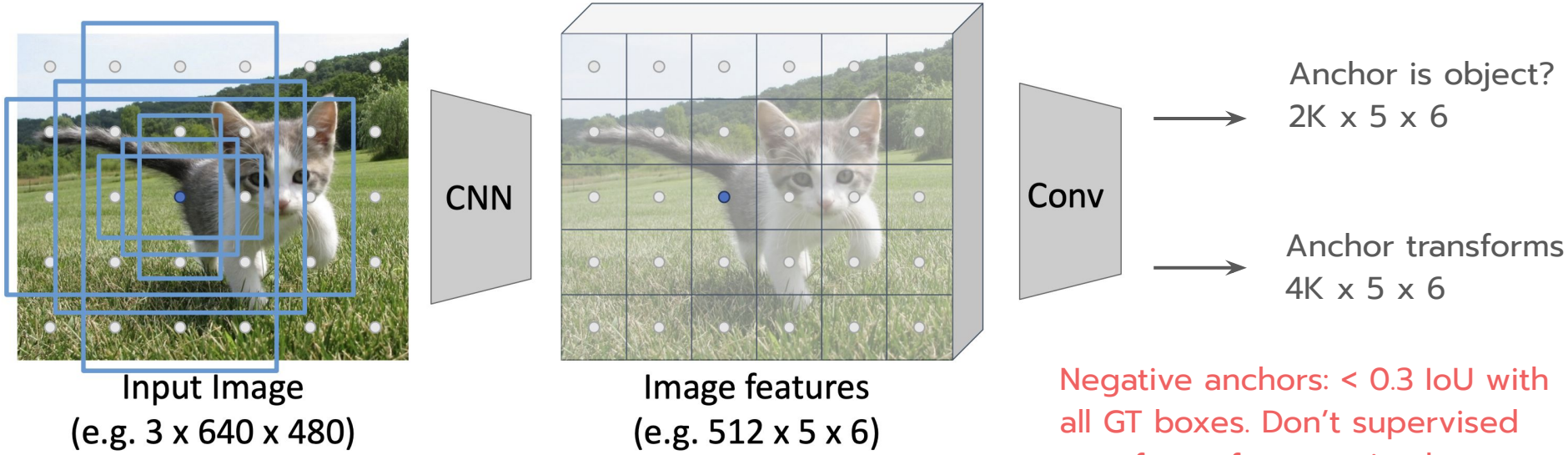
Object Detection

Faster R-CNN: 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



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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

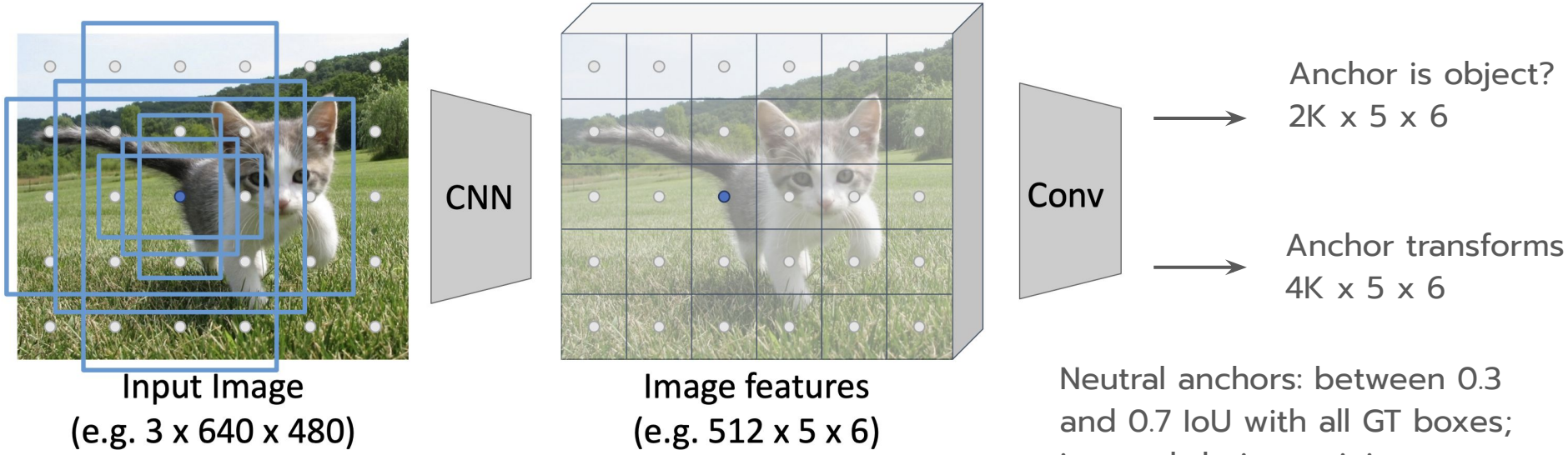
Object Detection

Faster R-CNN: 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



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

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

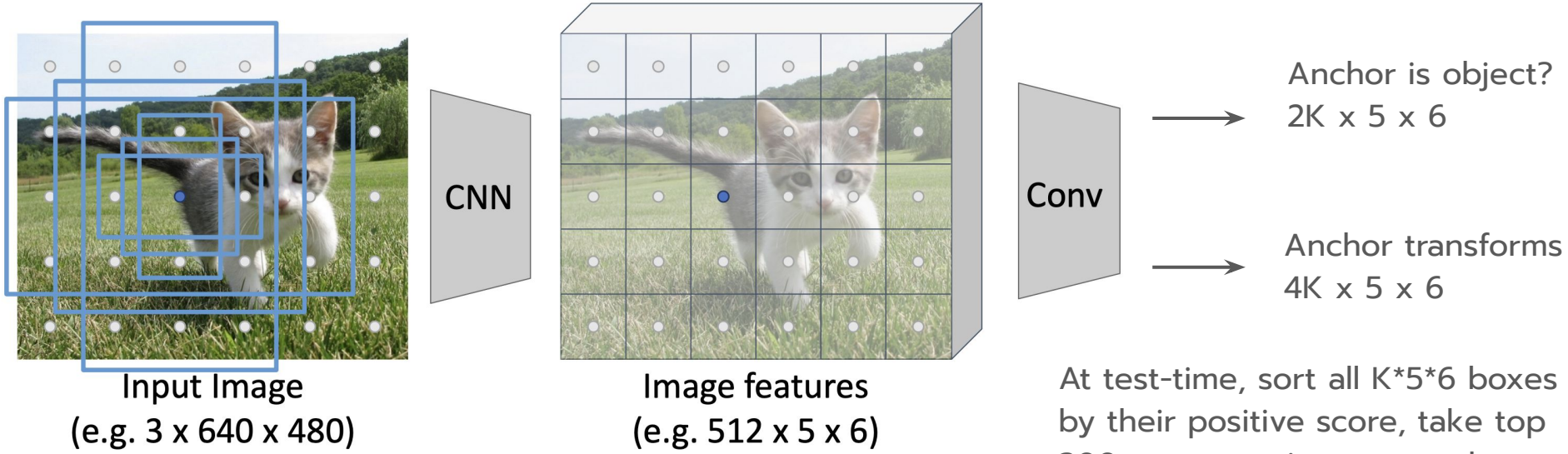
Object Detection

Faster R-CNN: Region Proposal Network (RPN)

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input

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



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

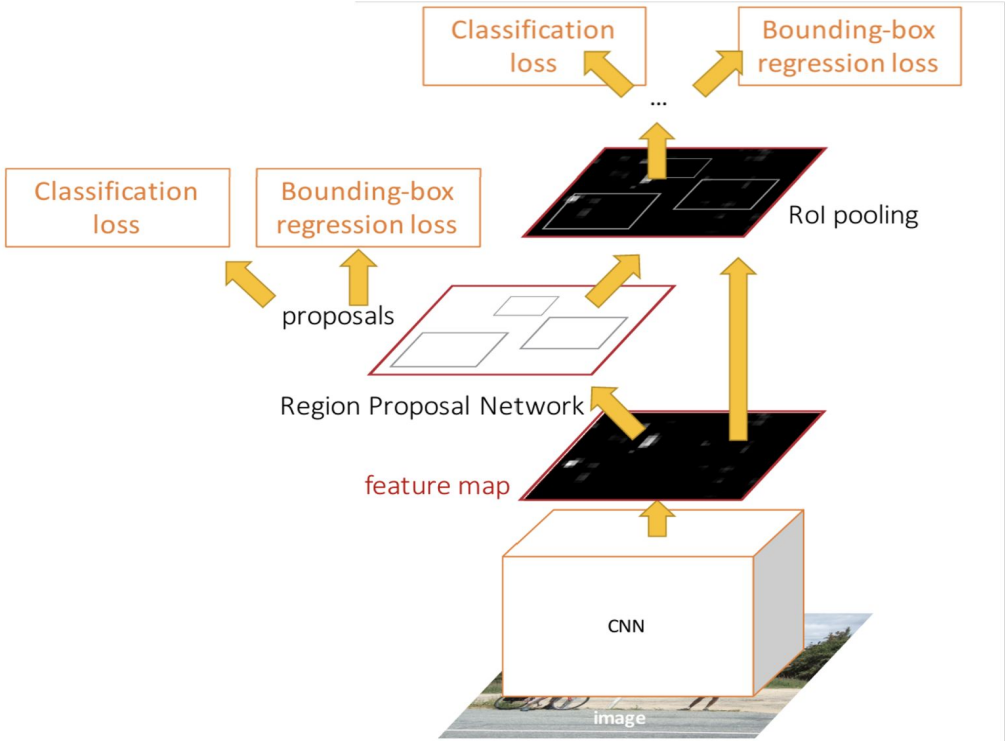
Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Faster R-CNN: Learnable Region Proposals

Jointly train with 4 losses:

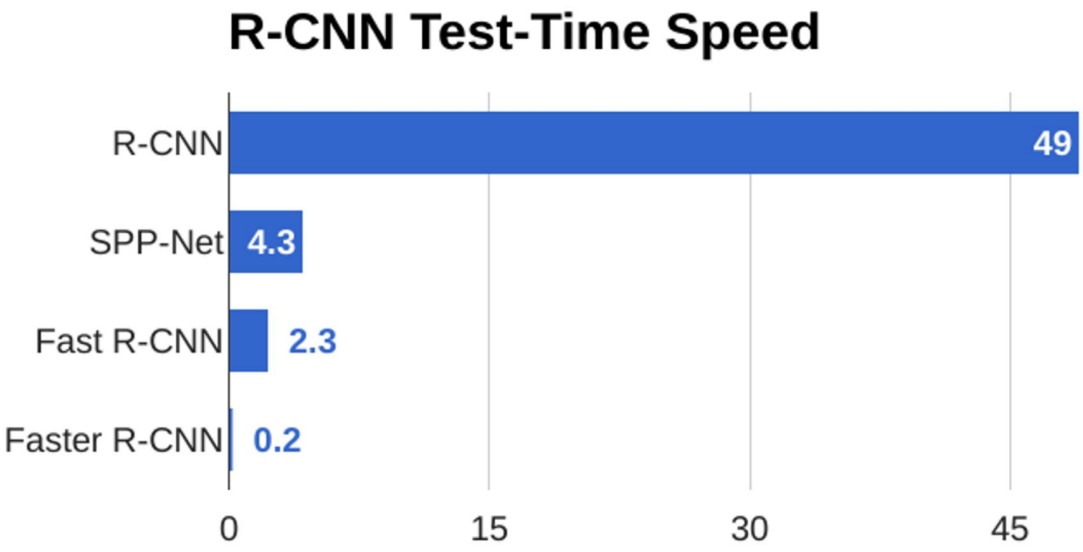
1. **RPN classification:** anchor box is object / not an object.
2. **RPN regression:** predict transform from anchor box to proposal box.
3. **Object classification:** classify proposals as background / object class.
4. **Object regression:** predict transform from proposal box to object box.



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Faster R-CNN: Learnable Region Proposals



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

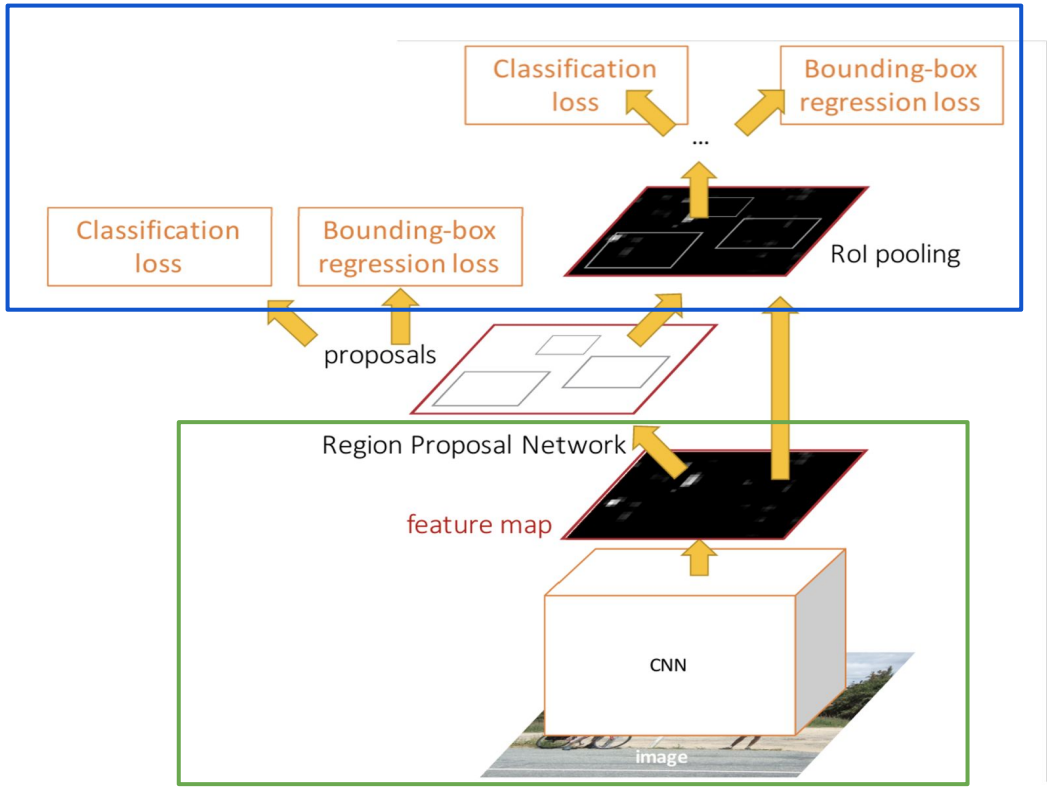
Object Detection

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



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

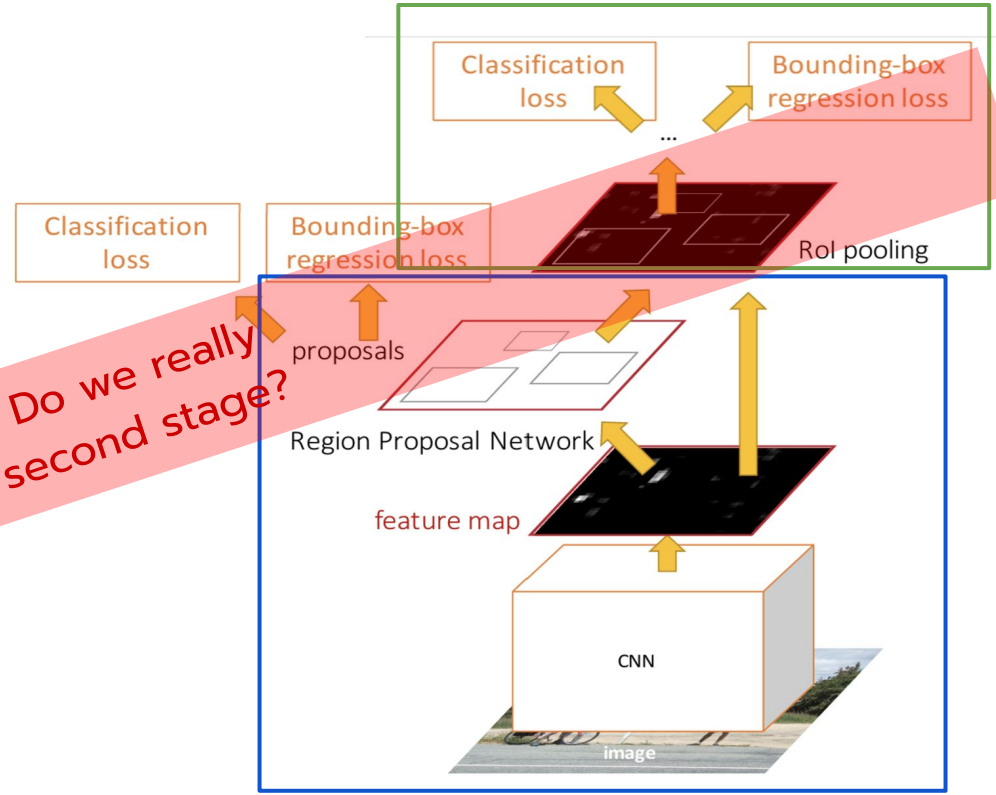
Faster R-CNN: Learnable Region Proposals

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

Question: Do we really need the second stage?



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

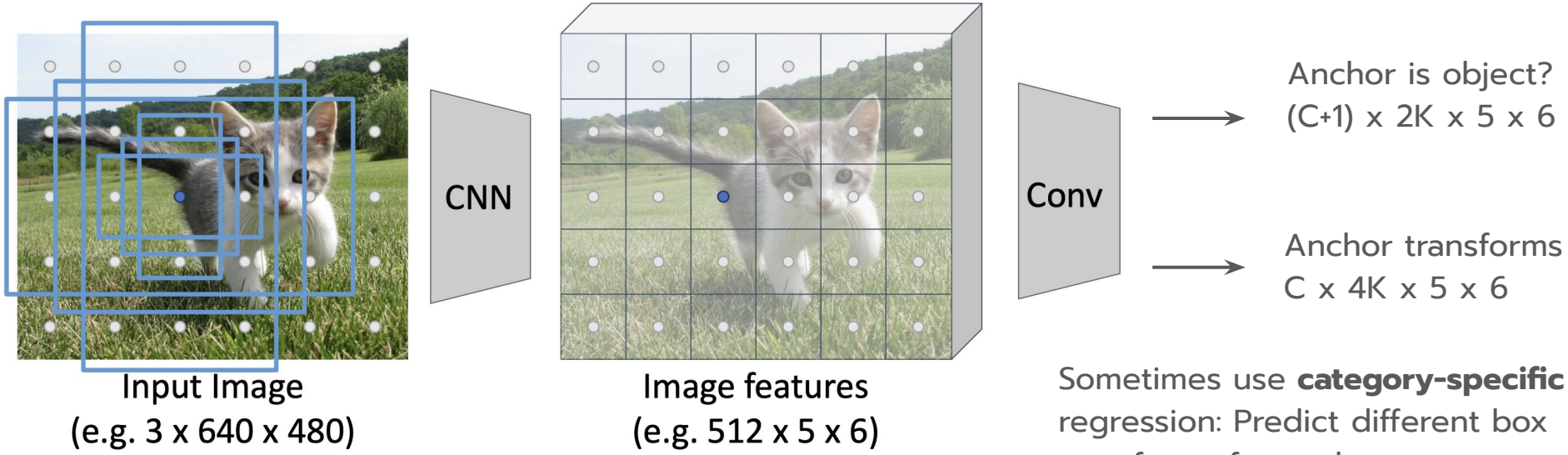
Single-Stage Object Detection (SSD)

RPN: Classify each anchor as object/not object

Single-Stage Detector: Classify each object as one of ***C*** categories (or background).

Run backbone CNN to get features aligned to input image

Each feature corresponds to a point in the input



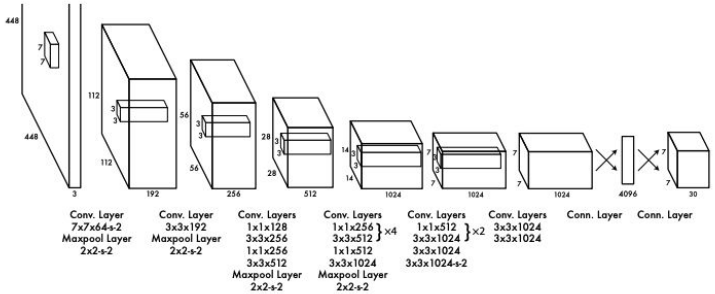
Sometimes use **category-specific** regression: Predict different box transforms for each category

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Object Detection

Single-Stage Object Detectors: YOLO

Architecture



Loss Function

$$L = L_{loc} + L_{cls}$$

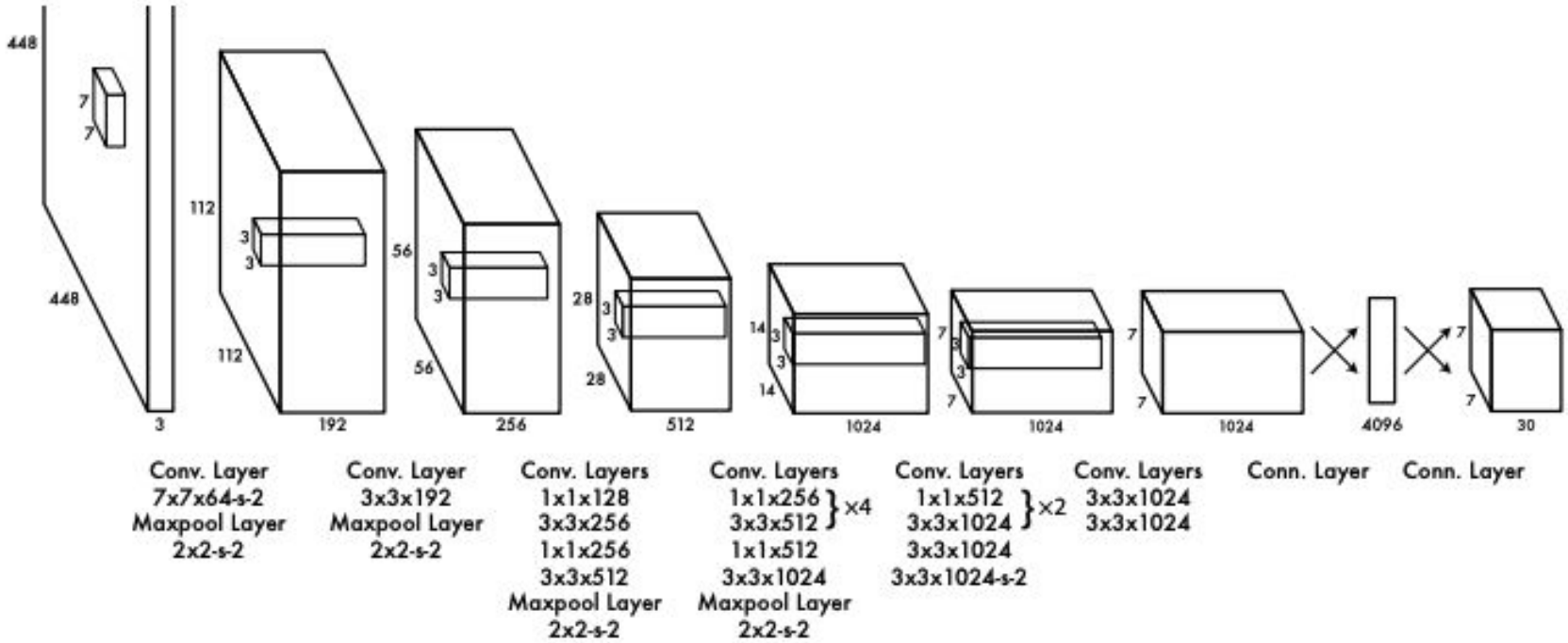
Slide adopted from: https://www.youtube.com/watch?v=svn9-xV7wjk&t=169s

Cherdsak Kingkan

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

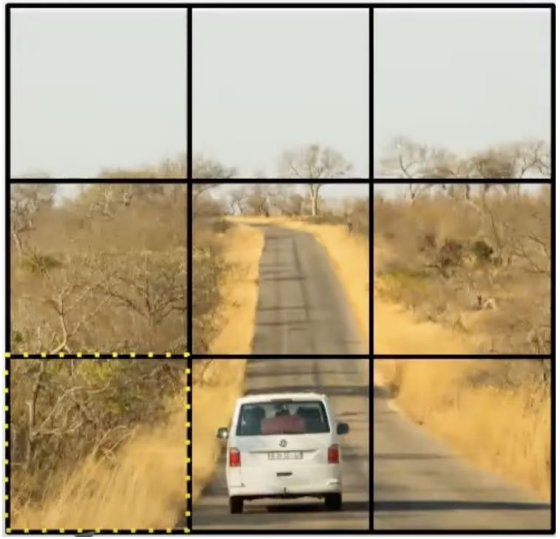
Single-Stage Object Detectors: YOLO



Object Detection

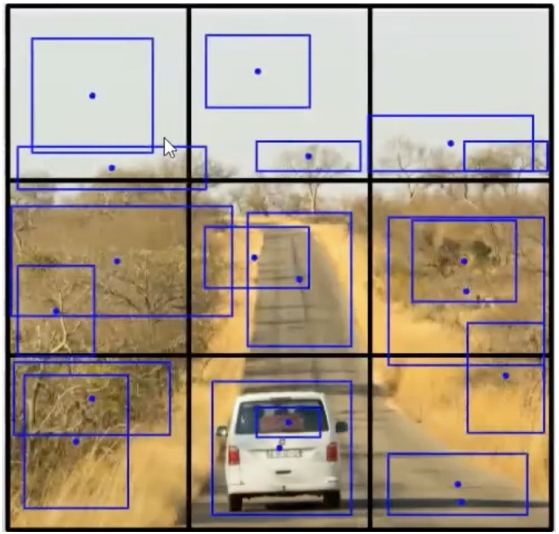
Single-Stage Object Detectors: YOLO - overview

1. Divide the image into cells with an *S x S* grid.



S = 3

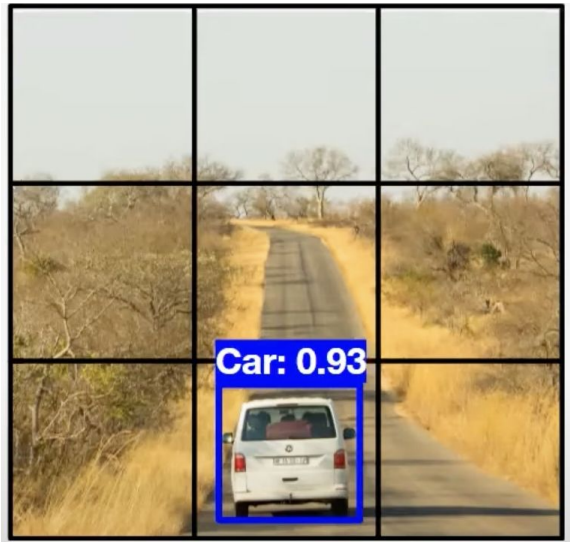
2. Each Cell predicts *B* bounding box



B = 2

Cell is responsible for detecting an object if the object's bounding box falls within the cell

3. Return bounding box above confidence threshold.



All other bounding boxes confidence prob. less than the threshold (say 0.9) so they are suppressed.

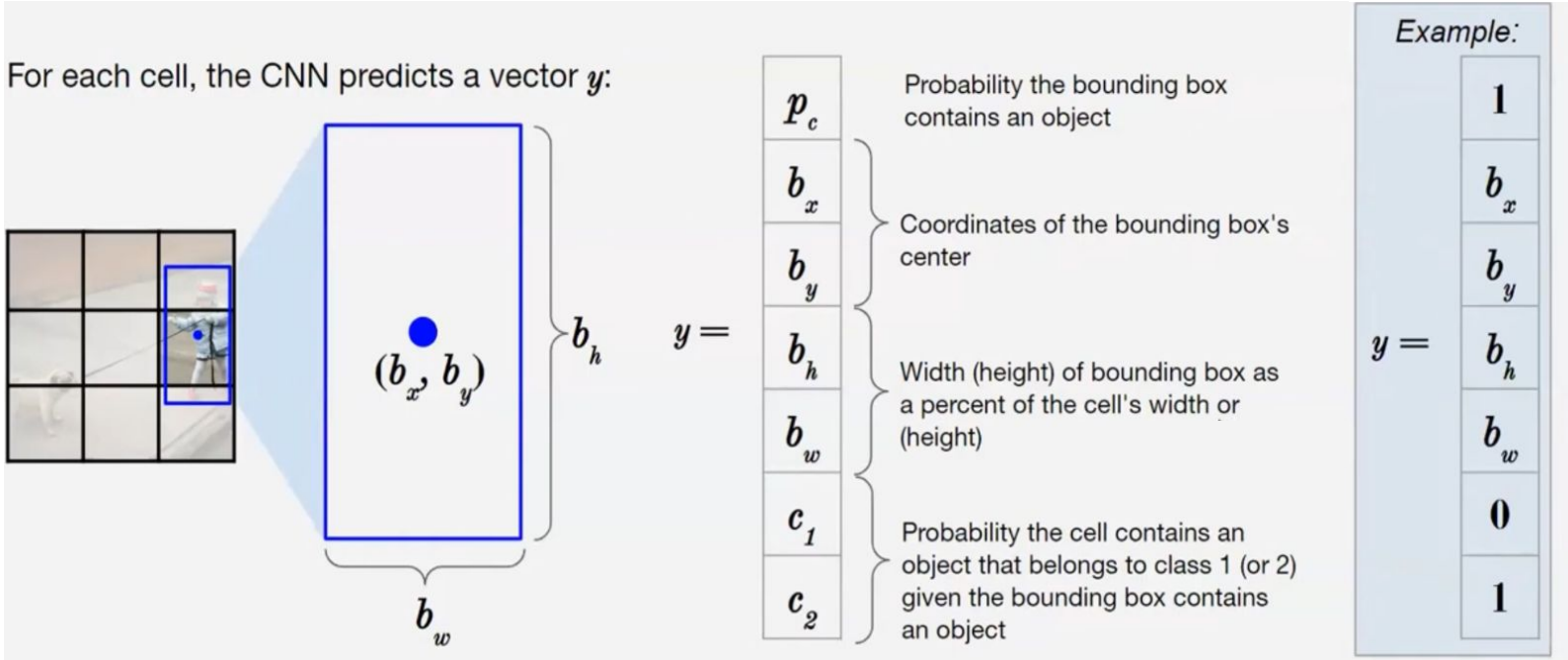
Slide adopted from: SCET Berkely, https://www.youtube.com/watch?v=2hAiJe8ITsE

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview

Let's use a simple example where there are 3x3 cells (**S=3**), each cell predicts 1 bounding box (**B=1**) and objects are either dog = 1 or human = 2.



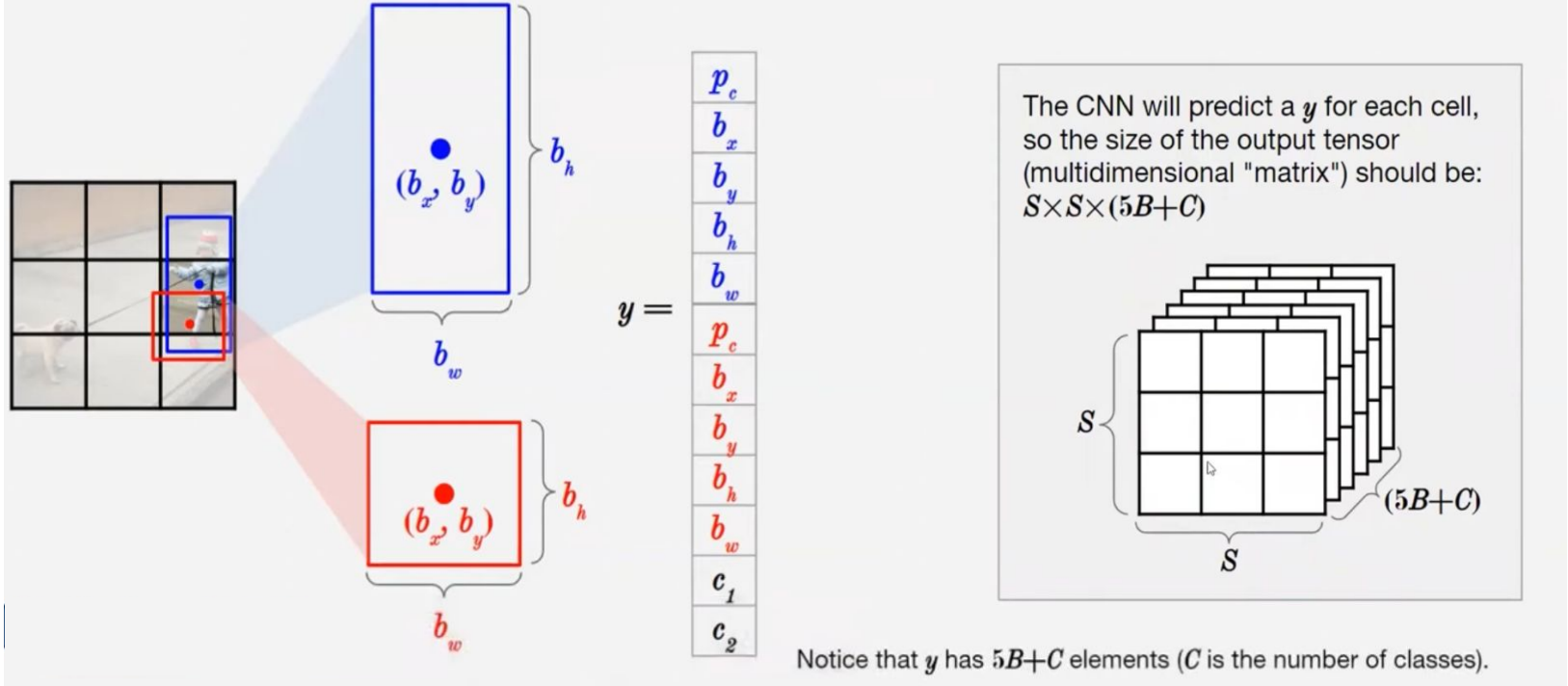
Slide adopted from: SCET Berkely, https://www.youtube.com/watch?v=2hAiJe8ITsE

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview

What happens if we predict multiple bounding boxes per cell (**B > 1**)? We simply augment *y*

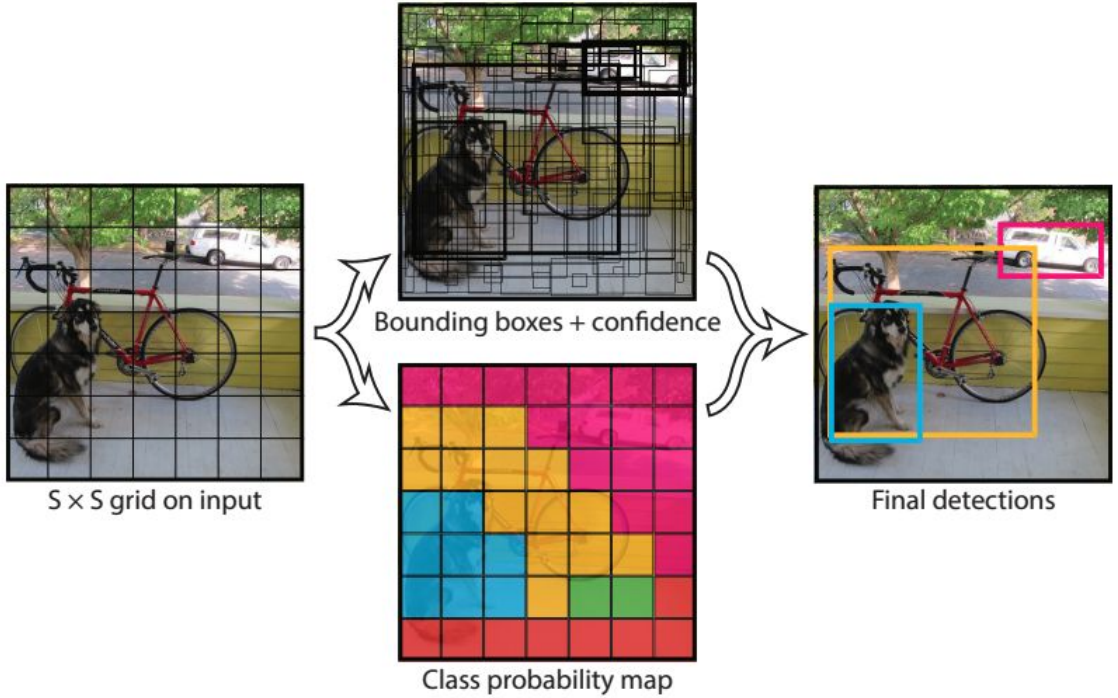


Slide adopted from: SCET Berkely, https://www.youtube.com/watch?v=2hAiJe8ITsE

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview

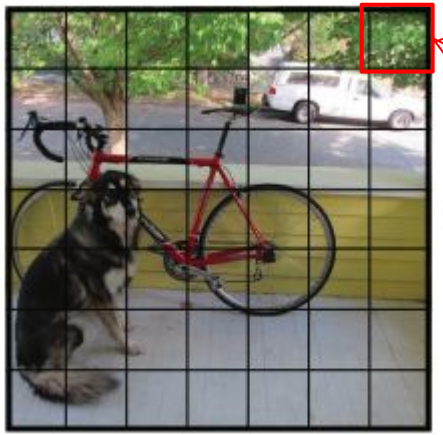


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview



S × S grid on input

$(B\text{x}5 + n)$

Output vector length

$$[p(c_1), p(c_2), \ldots, p(c_n])]$$

Class probabilities

$$[x_1, y_1, \sqrt{w_1}, \sqrt{h_1}, C_1]$$

$$[x_2, y_2, \sqrt{w_2}, \sqrt{h_2}, C_2]$$

...

$$[x_B, y_B, \sqrt{w_B}, \sqrt{h_B}, C_B]$$

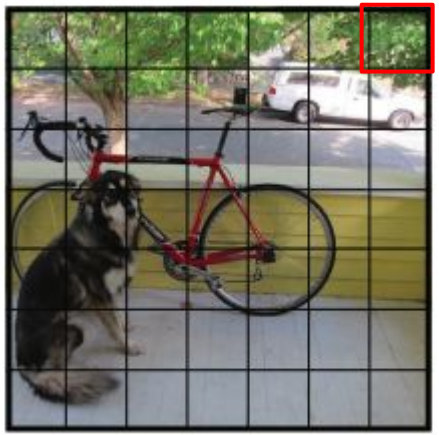
Bounding box predictions

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview



S × S grid on input

S = 7

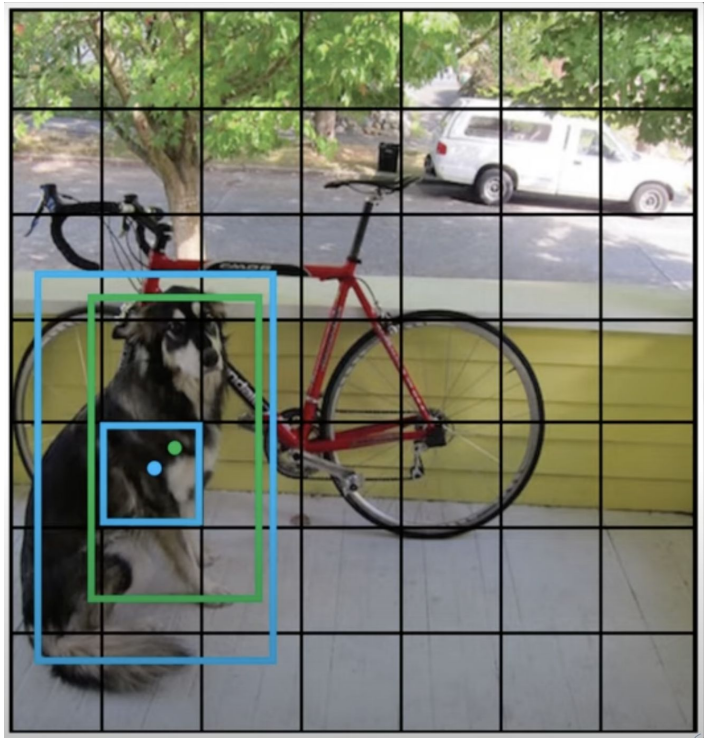
B = 2

n = 20

$(B\text{x}5 + n) = 30$

Object Detection

Single-Stage Object Detectors: YOLO - How it is trained



We are trying to predict a blue bounding box which is the ground truth with
class probabilities $[p(car) = 0, p(dog) = 1, \ldots, p(bicycle) = 0]$
and the center of the bb is at blue grid cell with
$[x, y, \sqrt{w}, \sqrt{h}, C]$

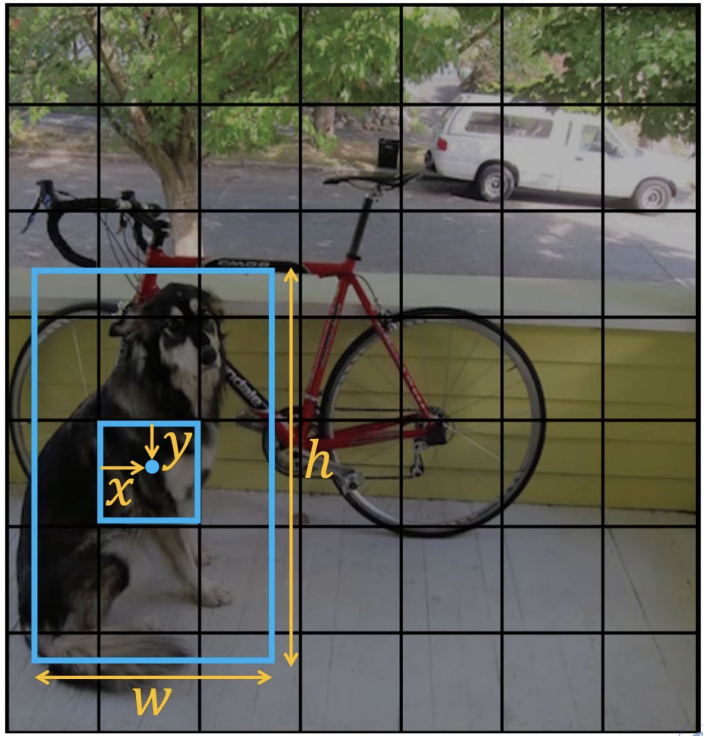
The model predicts the class probabilities along with 5 parameter of the green bb.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview



How the ground truth box coordinates are calculated

- x and y are calculated w.r.t. top left corner of each grid cell (proportion to the grid cell dimension)
- w and h are calculated w.r.t. the whole image dimension

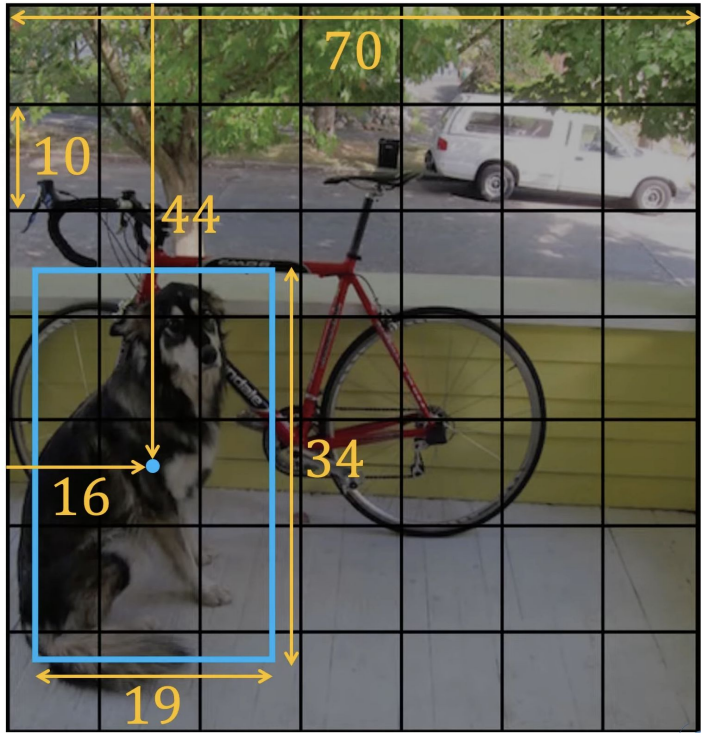
$$0 \leq x, y, w, h \leq 1$$

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview



How **the ground truth box coordinates** are calculated

$$x = \frac{(16 \mod 10)}{10} = 0.6$$

$$y = \frac{(44 \mod 10)}{10} = 0.4$$

$$w = \frac{19}{70} = 0.27$$

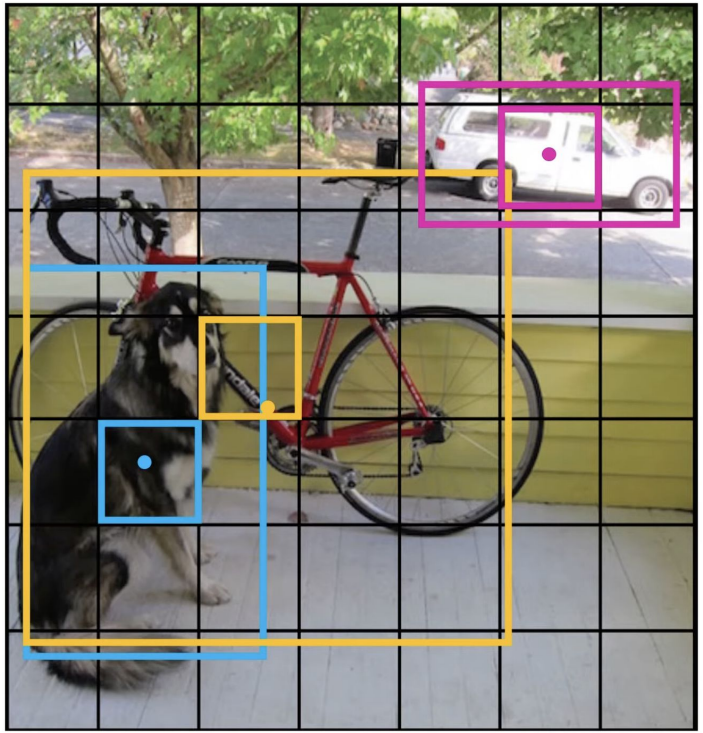
$$h = \frac{34}{70} = 0.49$$

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - How it is trained

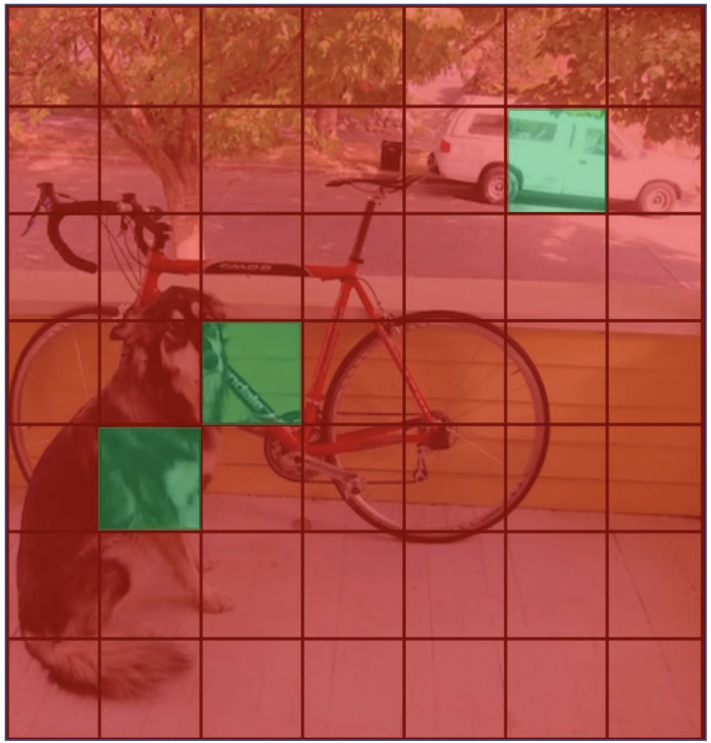


Ground truth Confidence

Q: Does an object appear in this grid cell or not ?

= 0

= IoU(pred,true)

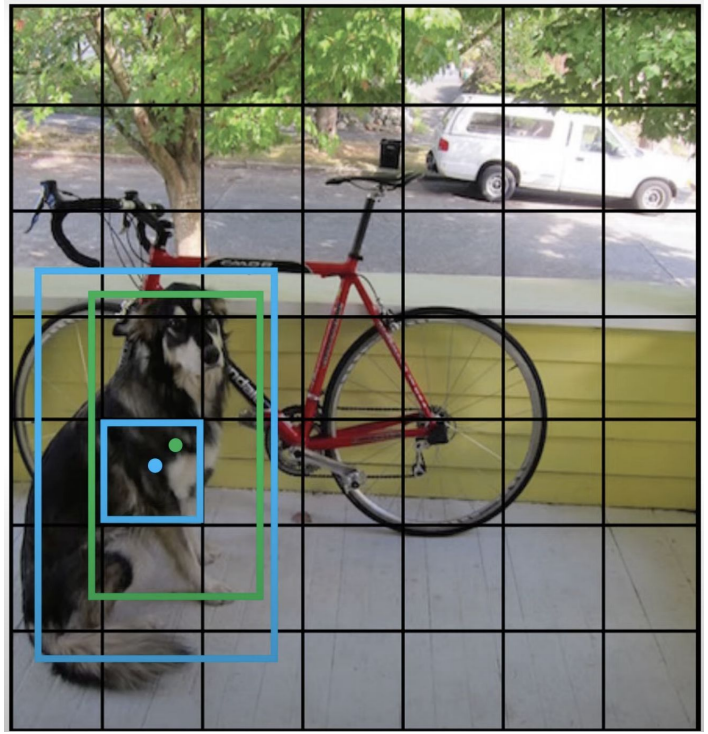


Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview



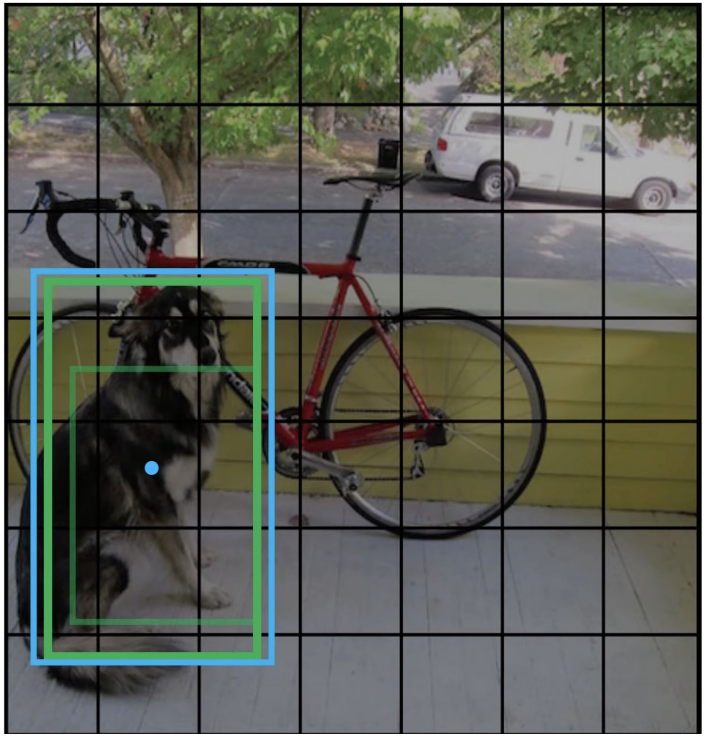
We know that each grid cell can produce **multiple predicted boxes**, which one should we use ?

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview



We know that each grid cell can produce **multiple predicted boxes**, which one should we use ?

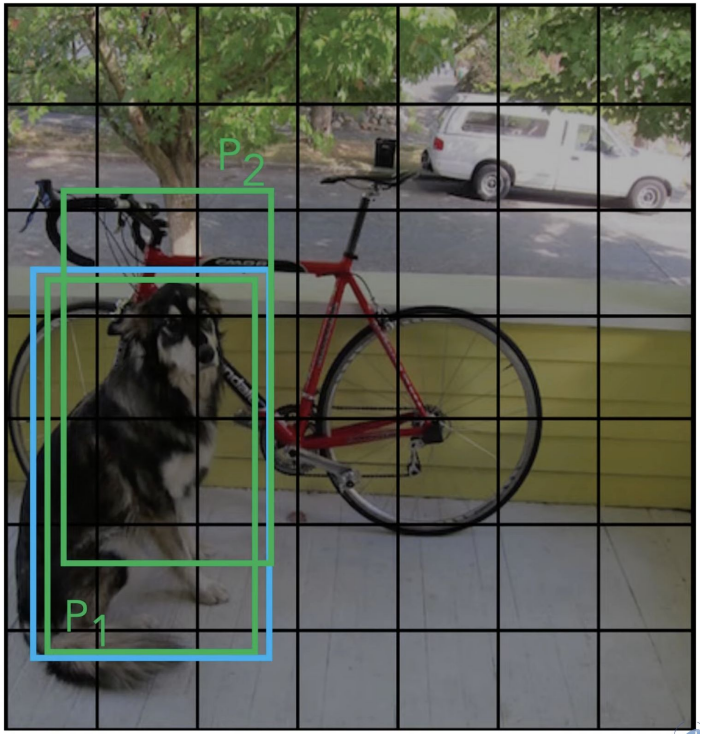
During **training**, we will choose the one with highest **IoU** score.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - overview



We know that each grid cell can produce multiple predicted boxes, which one should we use ?

During **training**, we will choose the one with highest **IoU** score.

During **inference,** we will **Non-max suppression (NMS)**

If IoU(P_1, P_2) > Threshold:
P = argmax(C(P_1), C(P_2))

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - Loss Function

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

$$L = L_{loc} + L_{cls}$$

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

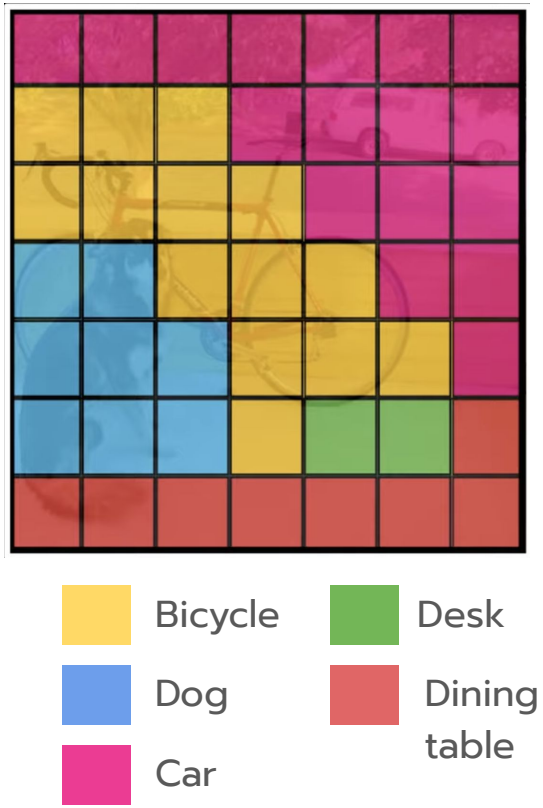
Single-Stage Object Detectors: YOLO - Loss Function

$$L_{cls} = \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2$$

MSE of class probabilities for the grids

$$\mathbb{1}_i^{obj} = \begin{cases} 1 & \text{if } obj \text{ in box } i, \\ 0 & \text{otherwise.} \end{cases}$$

The class loss only deals with cases where **an object has been mistaken for another object**, and not where there is in fact a background.



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - Loss Function

$$L_{loc} = L_{conf} + L_{coord}$$

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

Squared error is used to obtain good values for the class probabilities and all the bounding box parameters. So, **object detection is reduced to regression problem**.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - Loss Function

MSE for center point coordinates of the Bounding Boxes (BB), when there is an object.

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

MSE for *w* and *h* for BB. The square root is taken of *w* and *h* to partially **tackle** the fact that **small deviations** in **large** boxes **matter less** than in **small boxes,** when there is an object.

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

MSE for confidence scores of BB that **do** and **do-not** contain objects in them.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

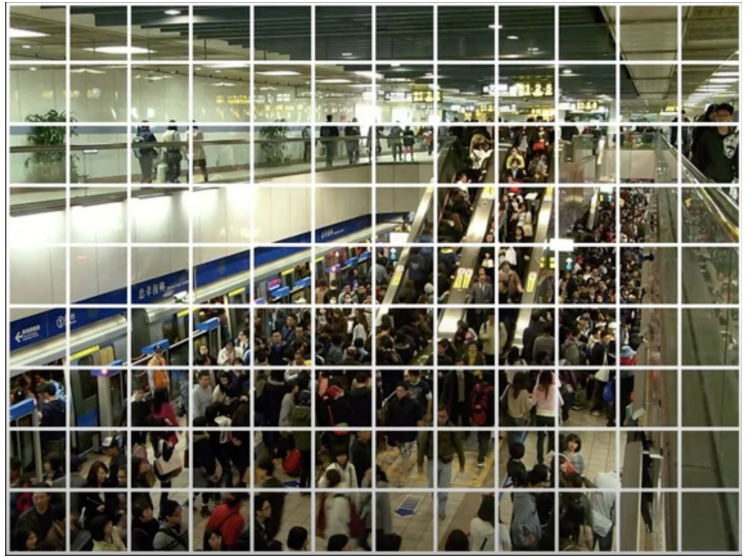
Single-Stage Object Detectors: YOLO - Limitations

- Each grid cell can contain only one class

Object Detection

Single-Stage Object Detectors: YOLO - Limitations

- Each grid cell can contain only one class
- Each grid cell can contain only (finite) *B* bounding boxes



Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - Limitations

- Each grid cell can contain only one class
- Each grid cell can contain only (finite) *B* bounding boxes
- Bounding box prediction struggles with unusual aspect ratios etc.

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - Limitations

- Each grid cell can contain only one class
- Each grid cell can contain only (finite) *B* bounding boxes
- Bounding box prediction struggles with unusual aspect ratios etc.
- Same error in small box is worse than in large box

$$
\begin{aligned}
&\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
&+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \boxed{\left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]} \\
&+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
&+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2
\end{aligned}
$$

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

Single-Stage Object Detectors: YOLO - Limitations

- Each grid cell can contain only one class
- Each grid cell can contain only (finite) *B* bounding boxes
- Bounding box prediction struggles with unusual aspect ratios etc.
- Same error in small box is worse than in large box
- Many grid cells do not contain an image
  - Training confidence scores pushed to zero
  - Coordinate loss contribution is low

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

5.0

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

0.5

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

Slide adopted from: Justin Johnson, EECS 498-007 / 598-005: Deep Learning for Computer Vision, University of Michigan, https://web.eecs.umich.edu/~justincj/teaching/eecs498/WI2022/

Redmon et al. "*You only look once: unified, real-time object detection*", 2015

Object Detection

- Two-Stage/Proposal
  - R-CNN
  - Fast R-CNN
  - Faster R-CNN
- Single-Stage/ Proposal-Free
  - SSD
  - YOLO

Slide adopted from: https://www.youtube.com/watch?v=svn9-xV7wjk&t=169s

Redmon et al. "*You only look once: unified, real-time object detection*", 2015