Lecture 13: Object Detection

A3 Grades, Midterm Grades

We are working on grading these this week

(Course staff needs spring break too!)

Big Problem: A4 Not Ready

A4 covers object detection (this week’s lectures);
won’t be ready until ~midweek
This messes up the schedule for the rest of the semester

Big Problem: A4 Not Ready

A4 covers object detection (this week's lectures);
won't be ready until ~midweek
This messes up the schedule for the rest of the semester

Option 1: Push back deadlines for A4, A5, and A6; they will end up compressed, with about 1.5 weeks for each of A4, A5, A6, mini-project

Big Problem: A4 Not Ready

A4 covers object detection (this week’s lectures);
won’t be ready until ~midweek
This messes up the schedule for the rest of the semester

Option 1: Push back deadlines for A4, A5, and A6; they will end up compressed, with about 1.5 weeks for each of A4, A5, A6, mini-project

Option 2: Cancel mini-project. Two full weeks for each of A4, A5, and A6. Points previously allocated to mini-project will be re-allocated to homework and midterm. A6 will become longer.

Big Problem: A4 Not Ready

A4 covers object detection (this week's lectures);
won't be ready until ~midweek
This messes up the schedule for the rest of the semester

Option 1: Push back deadlines for A4, A5, and A6; they will end up compressed, with about 1.5 weeks for each of A4, A5, A6, mini-project

Option 2: Cancel mini-project. Two full weeks for each of A4, A5, and A6. Points previously allocated to mini-project will be re-allocated to homework and midterm. A6 will become longer.

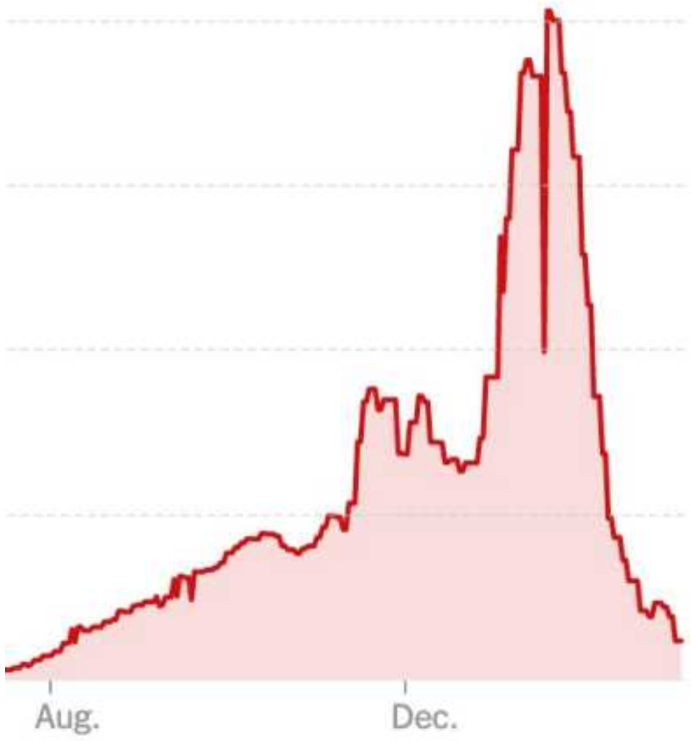
I will send out a poll via Piazza tonight

Lecture Format

COVID cases have fallen dramatically since the start of the semester

How would people feel about in-person lecture starting next week?

Will include question in the poll to be sent tonight



Source: https://www.nytimes.com/interactive/2021/us/michigan-covid-cases.html

Last Time: Deep Learning Software

Static Graphs vs **Dynamic Graphs**

PyTorch vs **TensorFlow**

So Far: Image Classification

This image is CC0 public domain

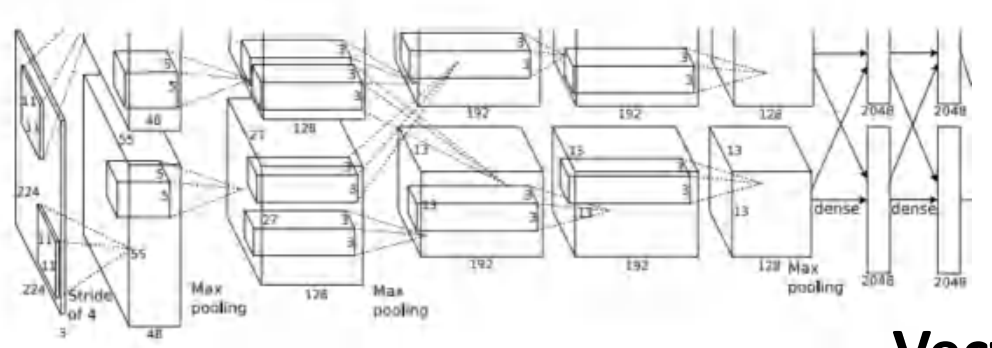


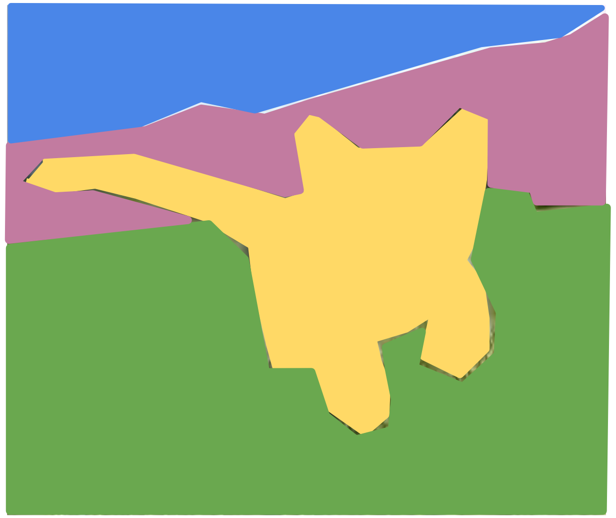
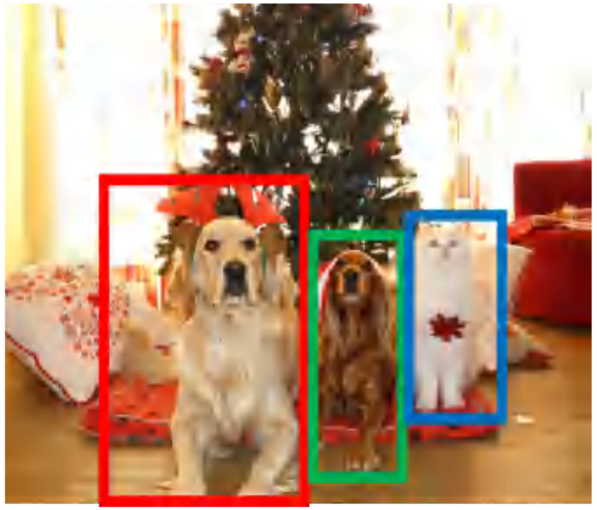
Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector: 4096

Fully-Connected: 4096 to 1000

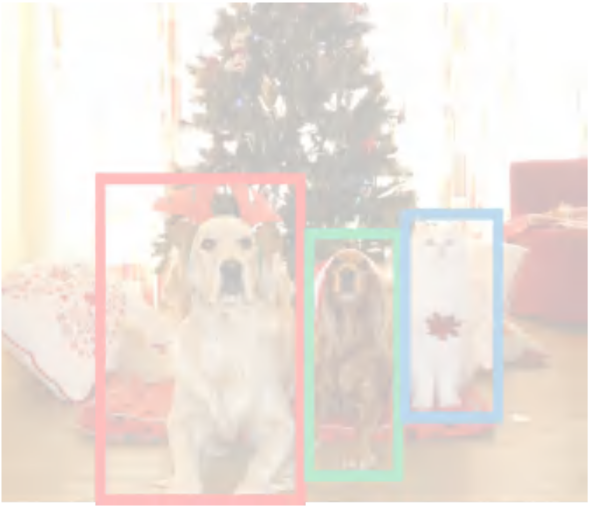
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

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	

This image is CC0 public domain

Classification: Transferring to New 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

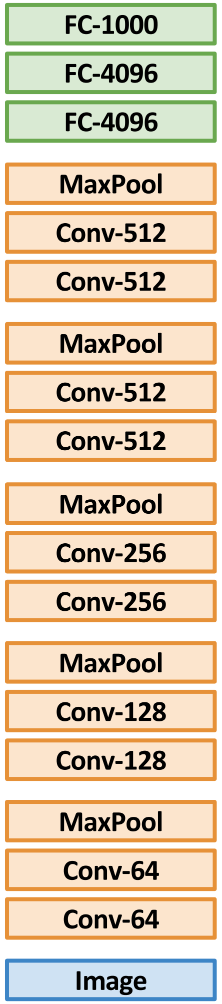
Multiple Objects

This image is CC0 public domain

Transfer Learning: Generalizing to New Tasks

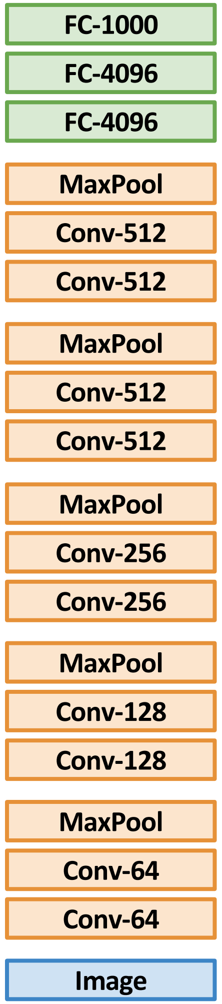
Transfer Learning

1. Train on ImageNet

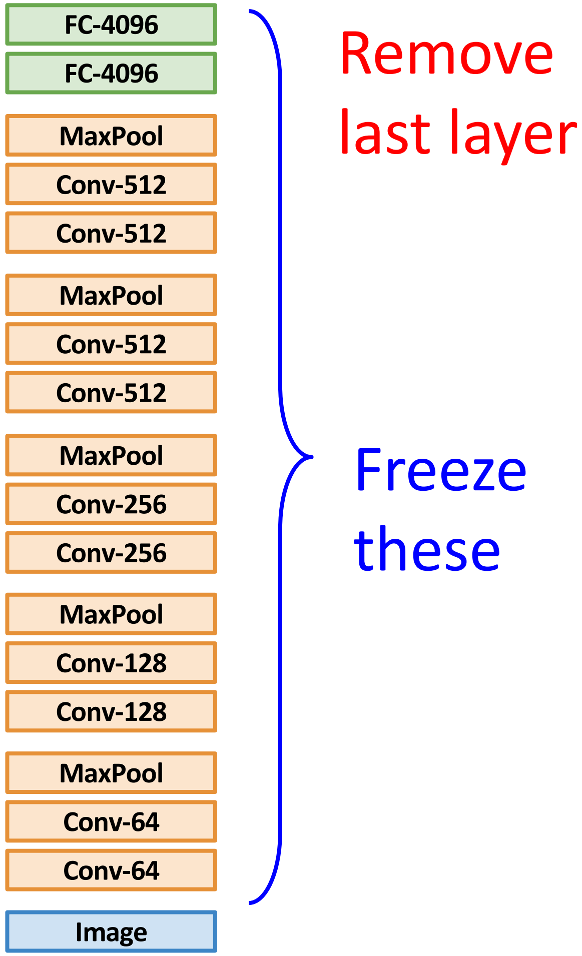


Transfer Learning: Feature Extraction

1. Train on ImageNet

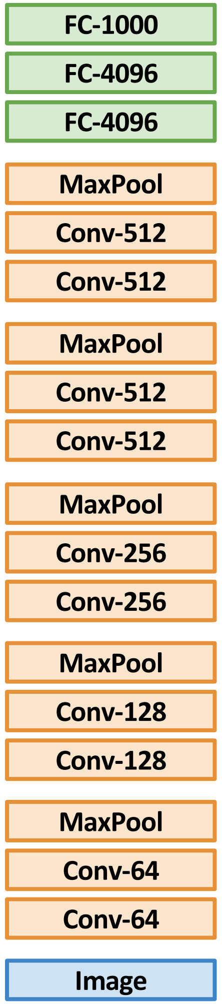


2. Extract features with CNN, train linear model

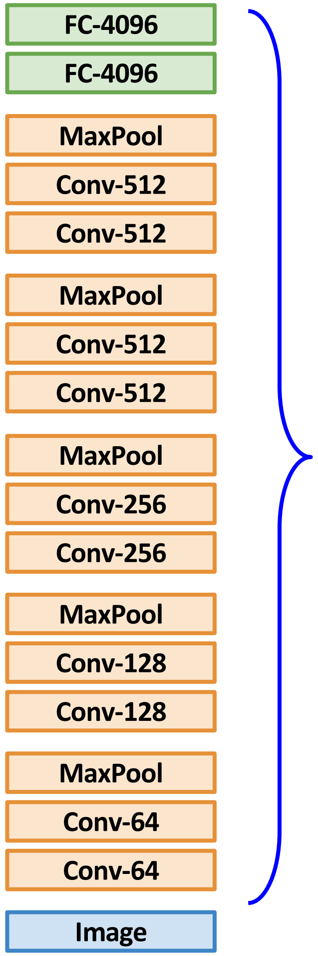


Transfer Learning: Feature Extraction

1. Train on ImageNet



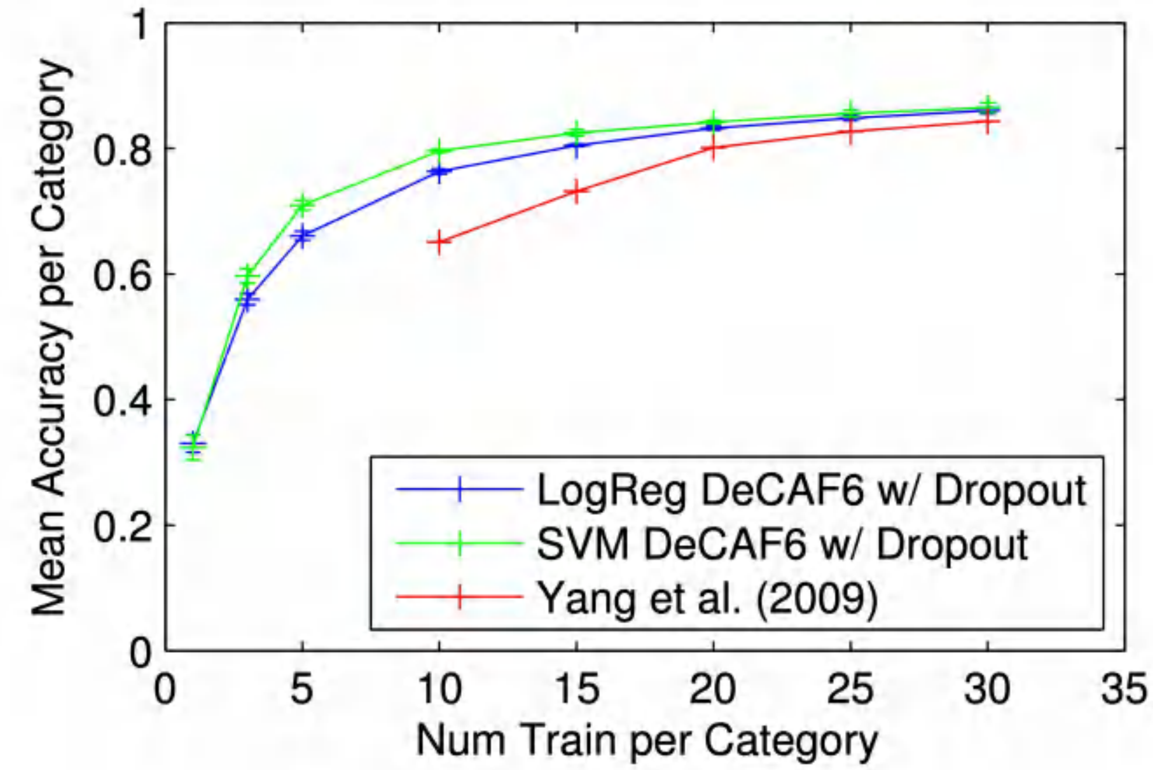
2. Extract features with CNN, train linear model



Remove last layer

Freeze these

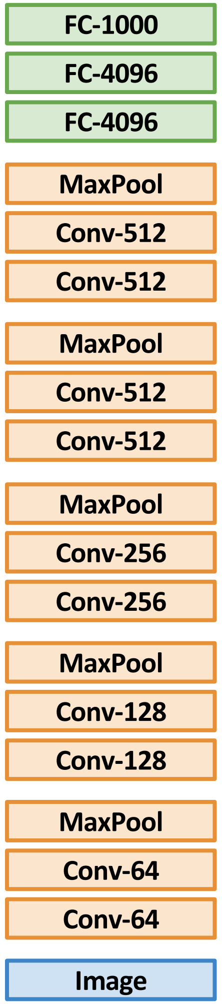
Classification on Caltech-101



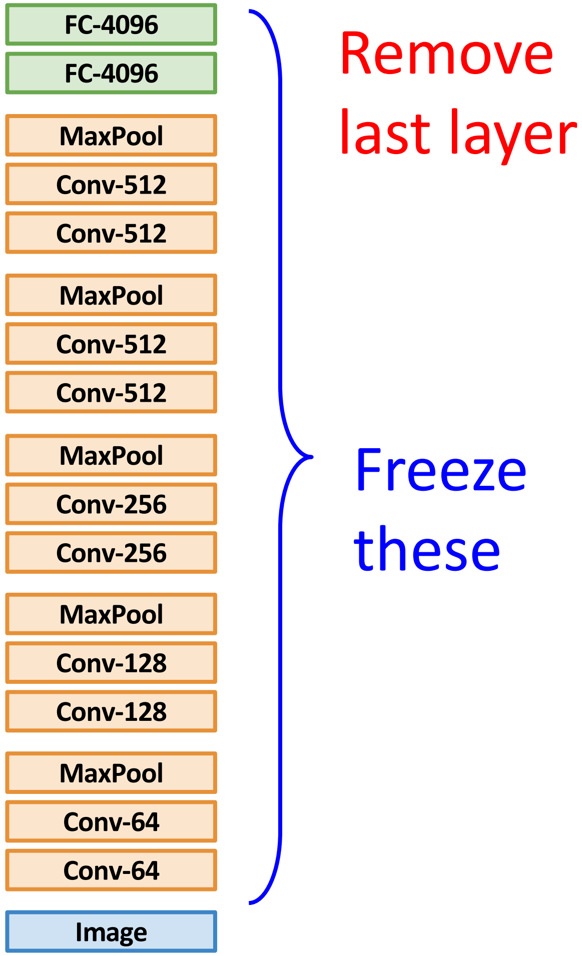
Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014

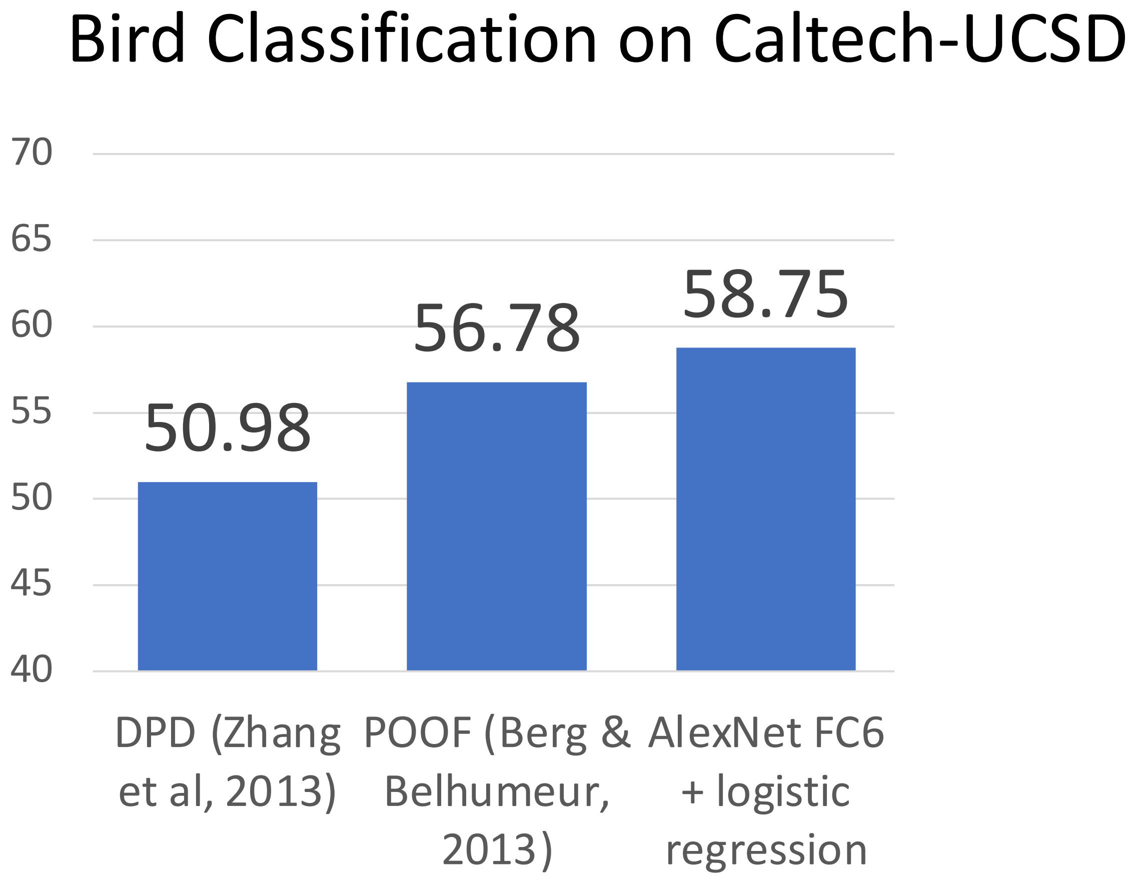
Transfer Learning: Feature Extraction

1. Train on ImageNet



2. Extract features with CNN, train linear model

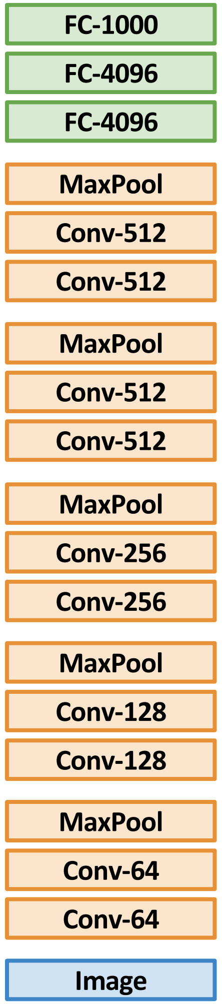




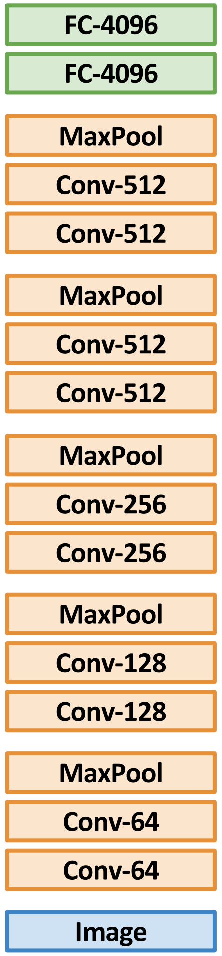
Donahue et al, “DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition”, ICML 2014

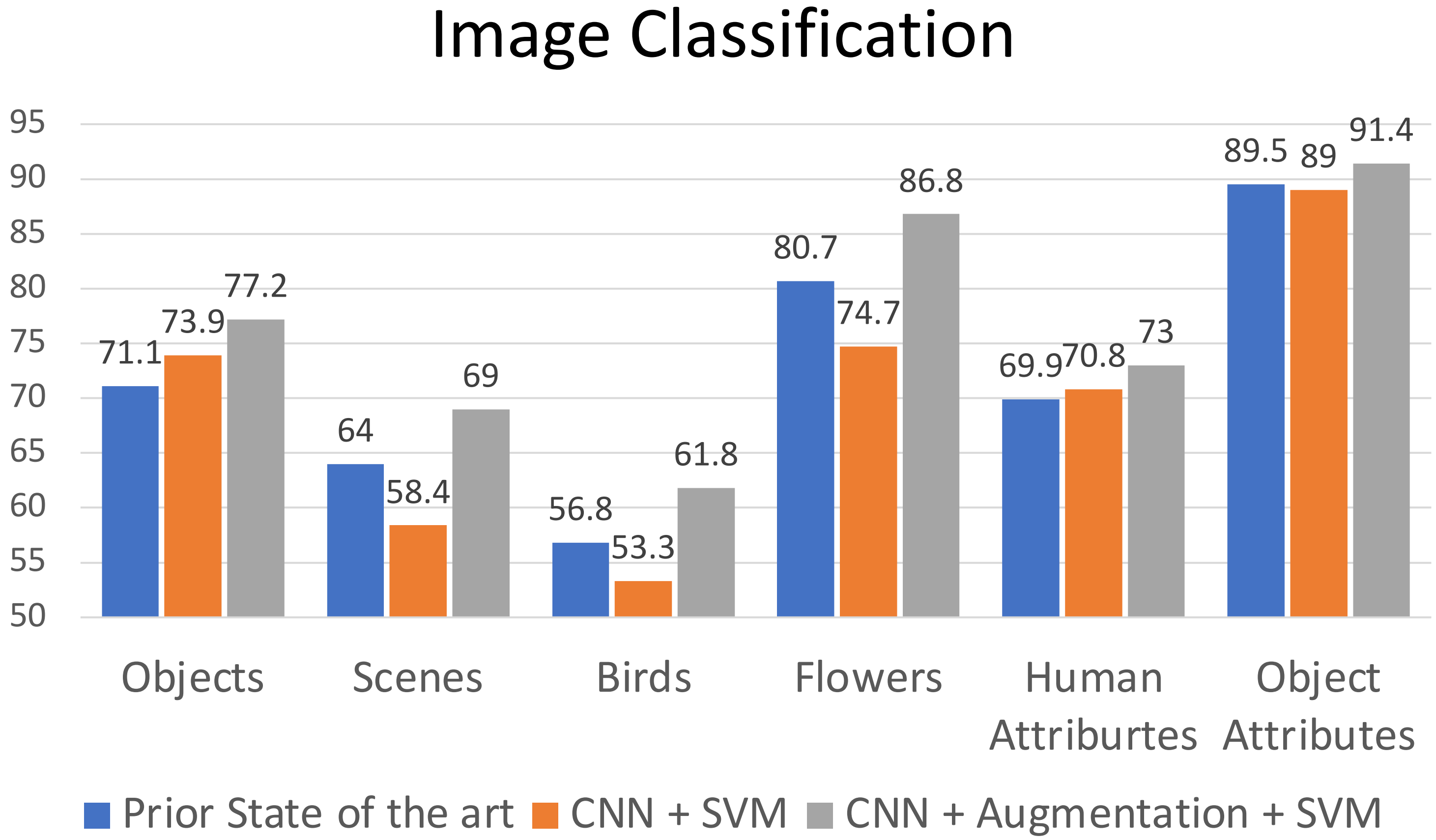
Transfer Learning: Feature Extraction

1. Train on ImageNet



2. Extract features with CNN, train linear model

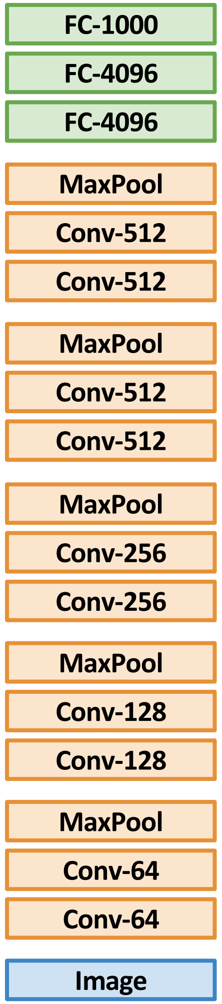




Razavian et al, “CNN Features Off-the-Shelf: An Astounding Baseline for Recognition”, CVPR Workshops 2014

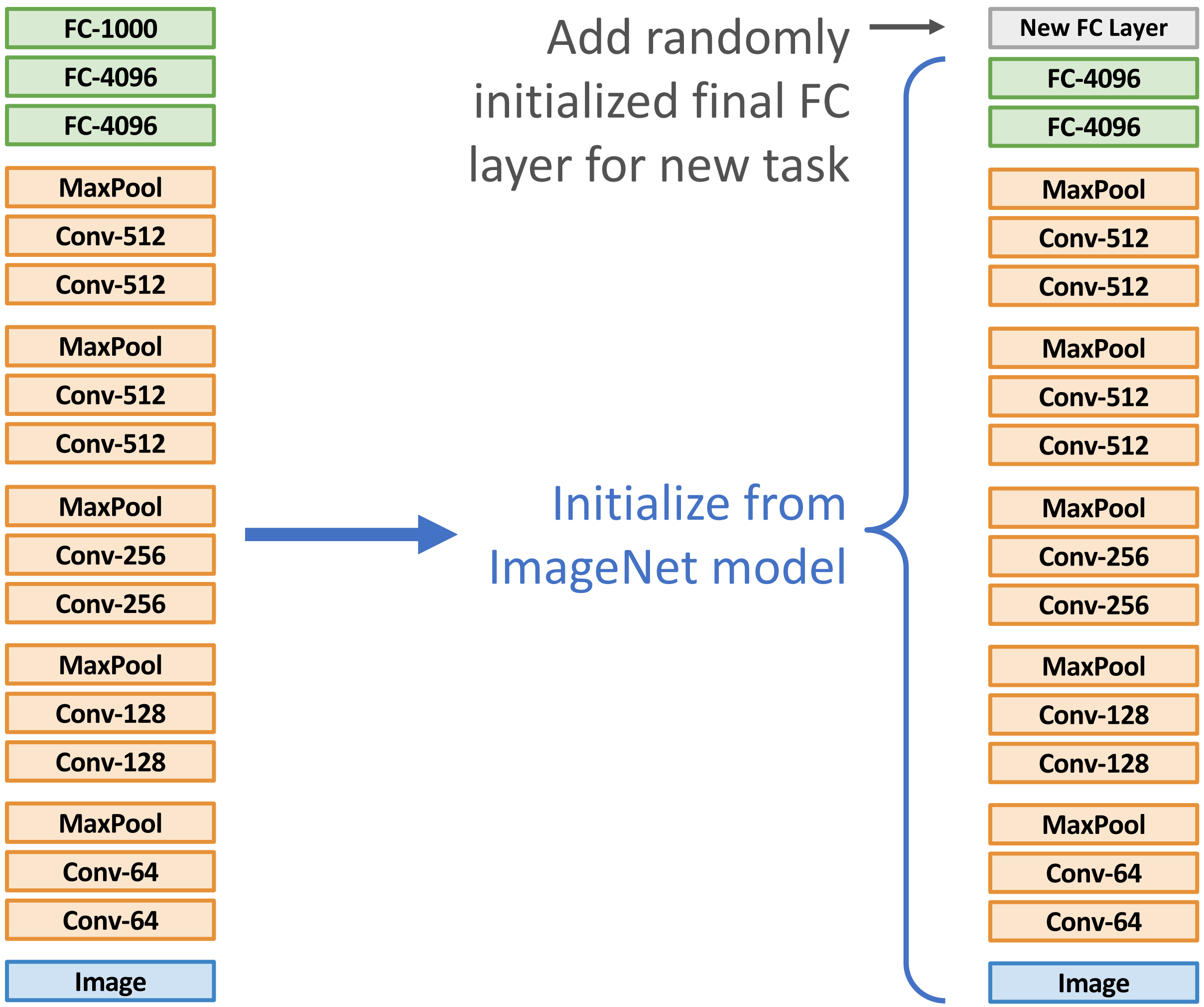
Transfer Learning: Fine-Tuning

1. Train on ImageNet



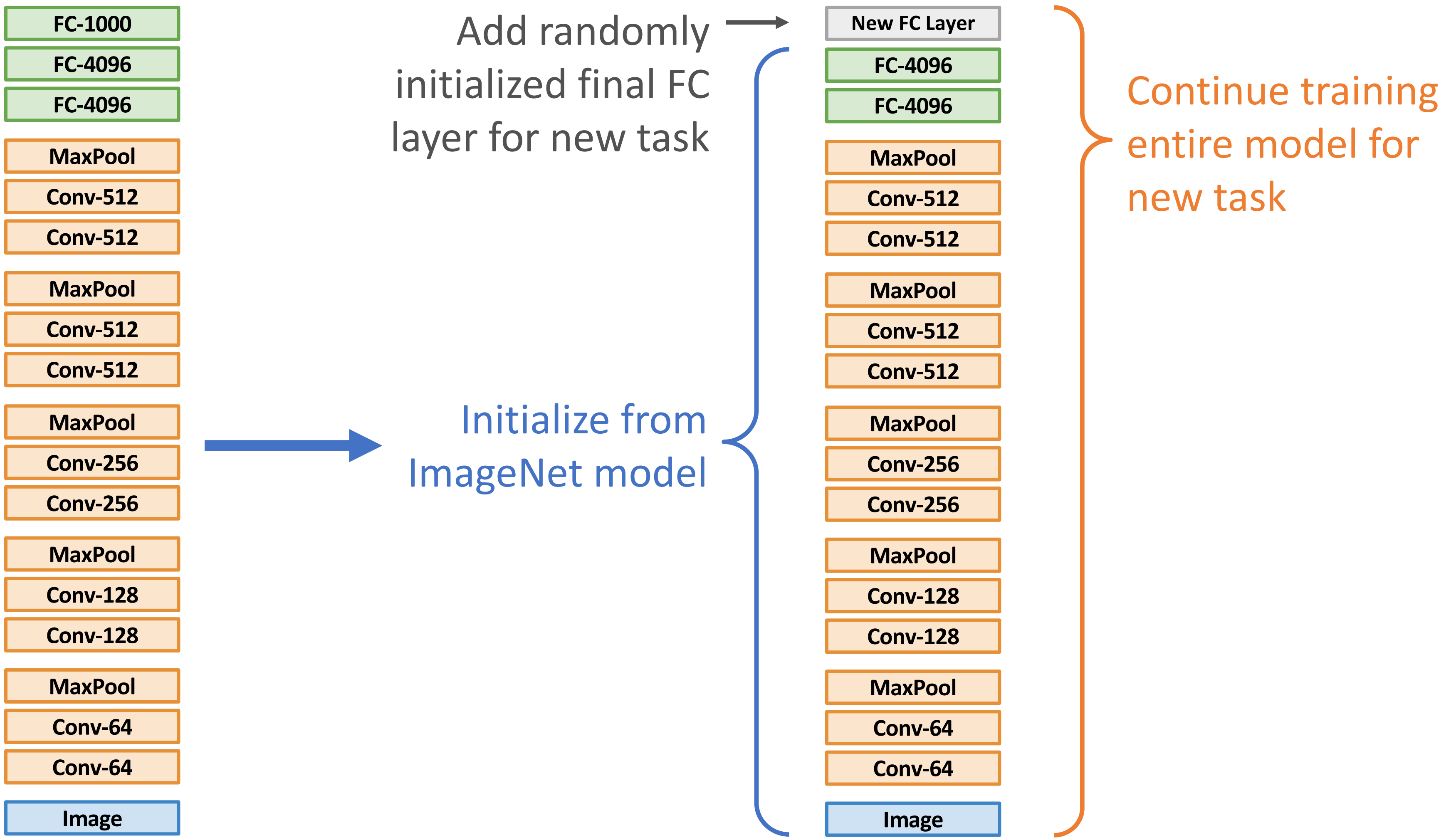
Transfer Learning: Fine-Tuning

1. Train on ImageNet



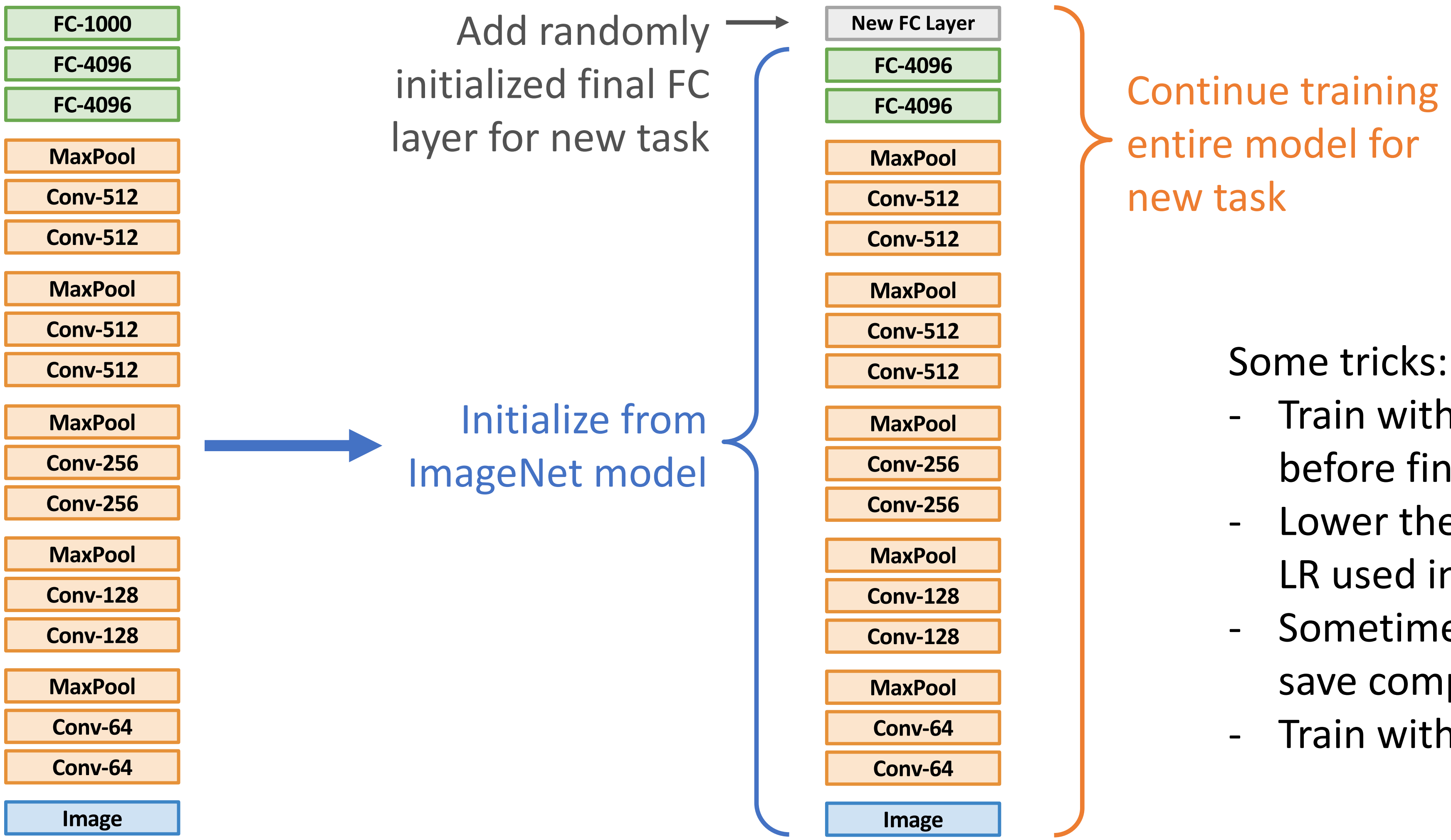
Transfer Learning: Fine-Tuning

1. Train on ImageNet



Transfer Learning: Fine-Tuning

1. Train on ImageNet

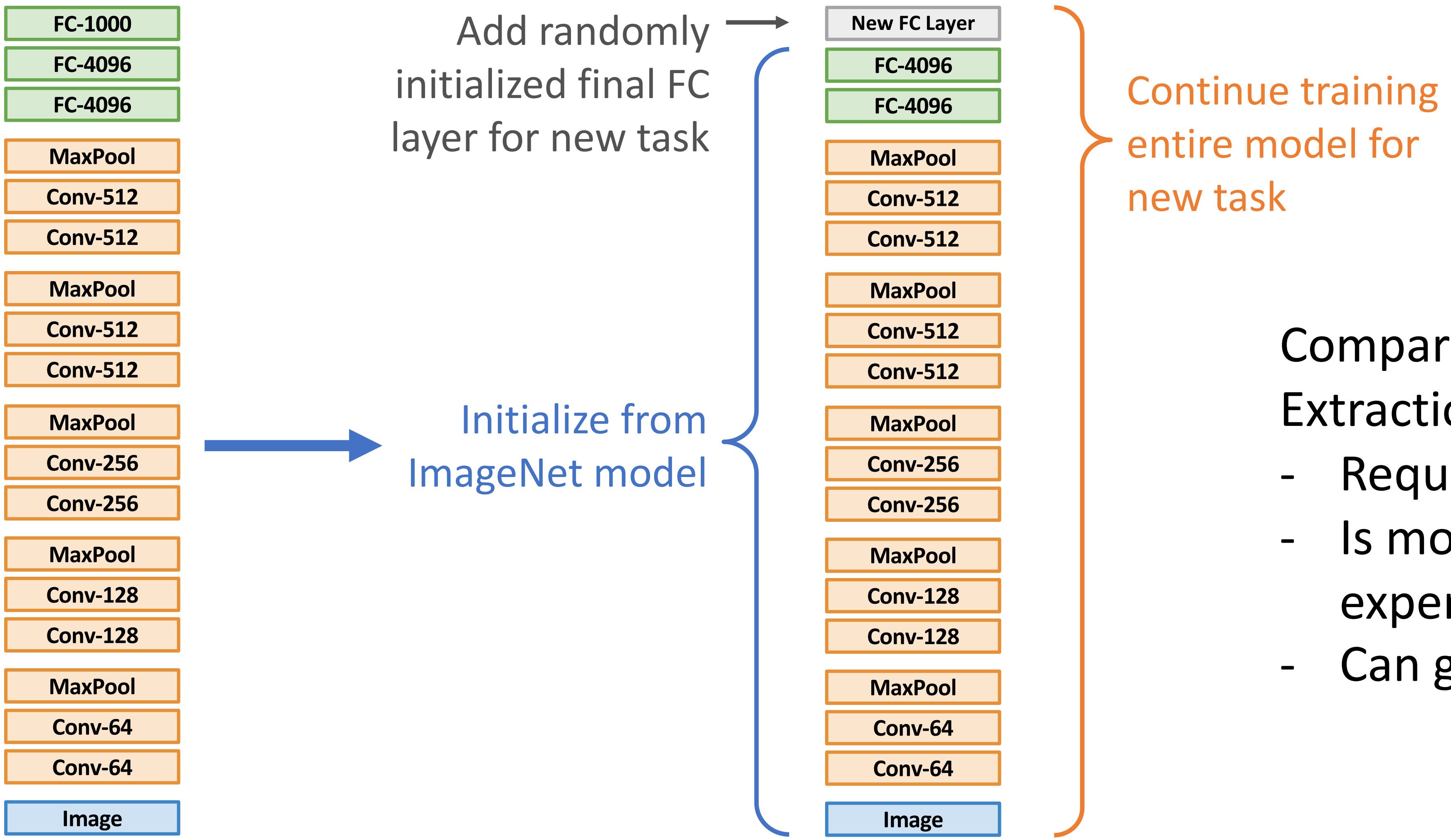


Some tricks:

- Train with feature extraction first before fine-tuning
- Lower the learning rate: use ~1/10 of LR used in original training
- Sometimes freeze lower layers to save computation
- Train with BatchNorm in “test” mode

Transfer Learning: Fine-Tuning

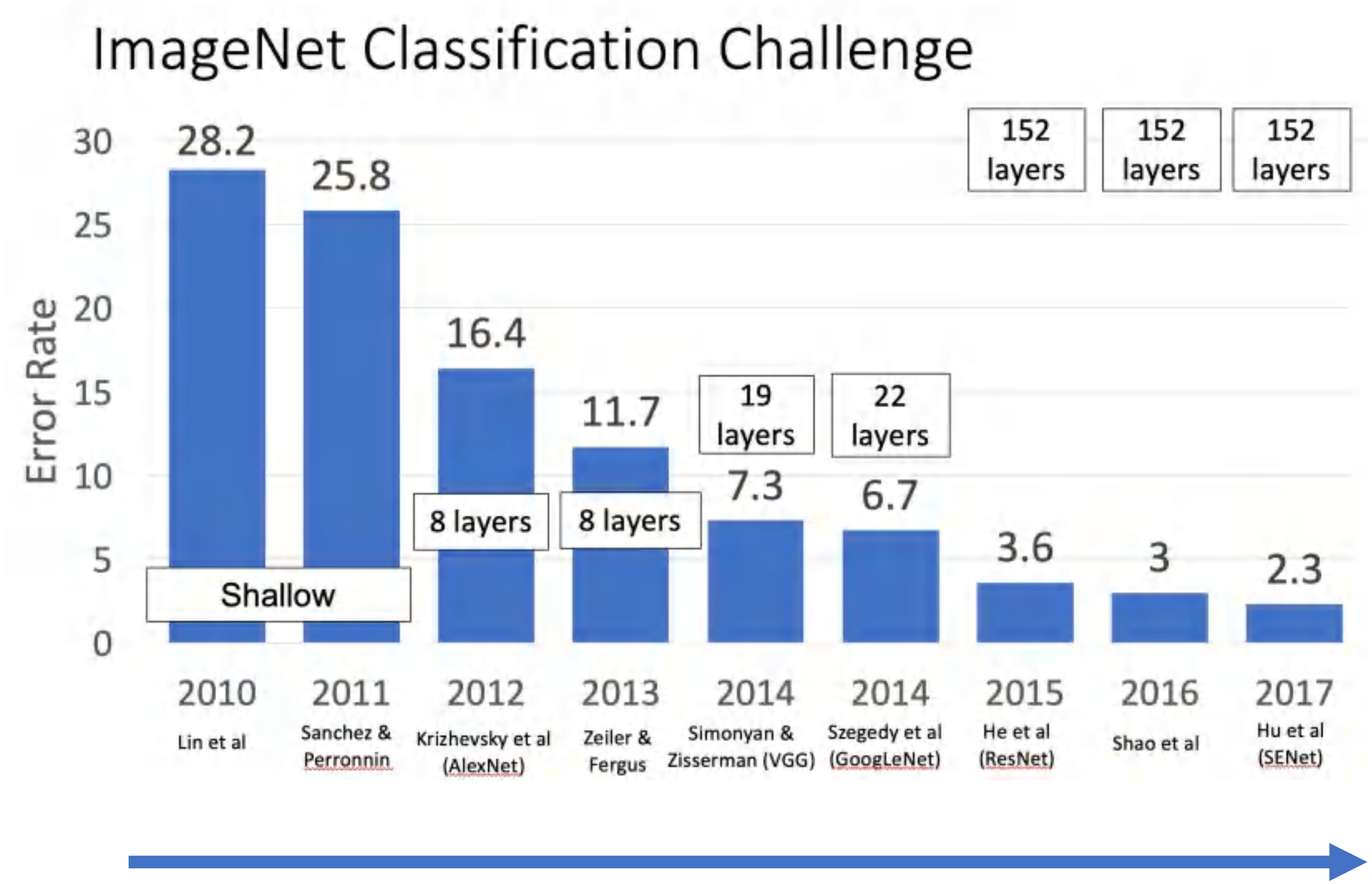
1. Train on ImageNet



Compared with Feature Extraction, Fine-Tuning:

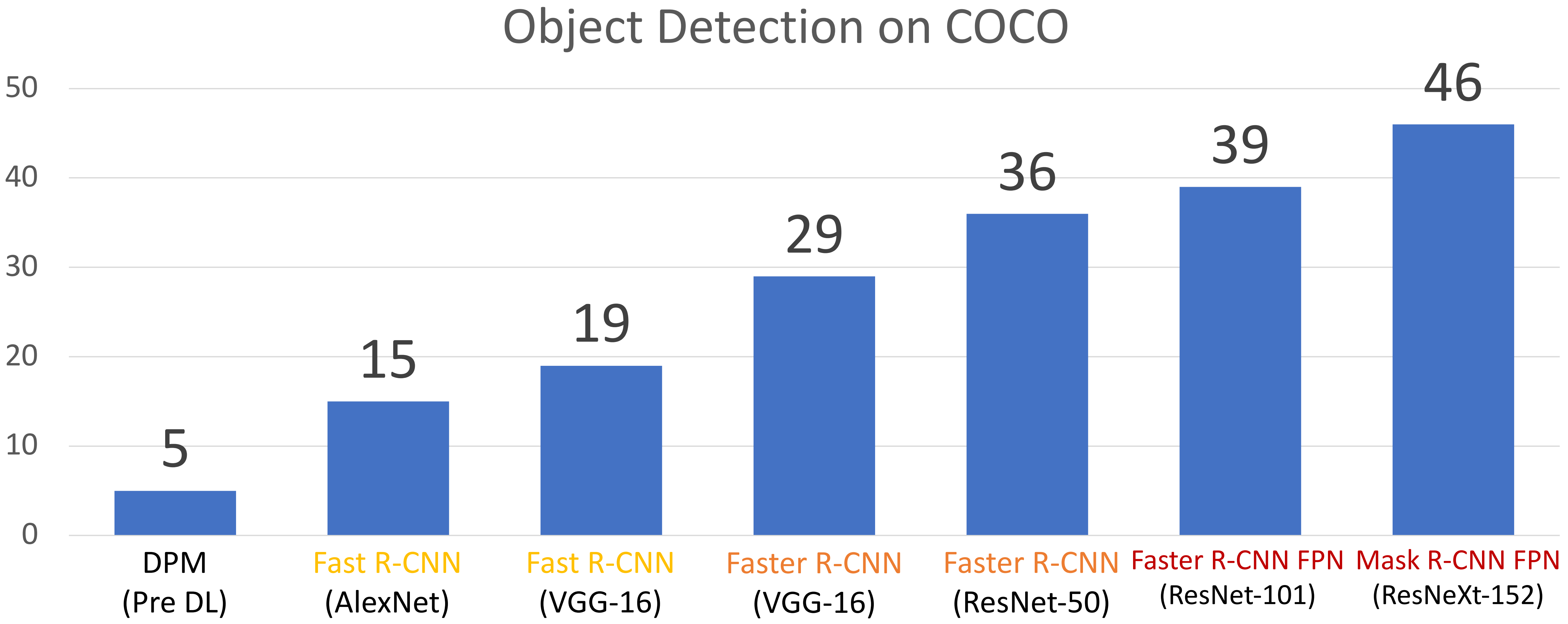
- Requires more data
- Is more computationally expensive
- Can give higher accuracies

Transfer Learning: Architecture Matters!



Improvements in CNN architectures lead to improvements in many downstream tasks thanks to transfer learning!

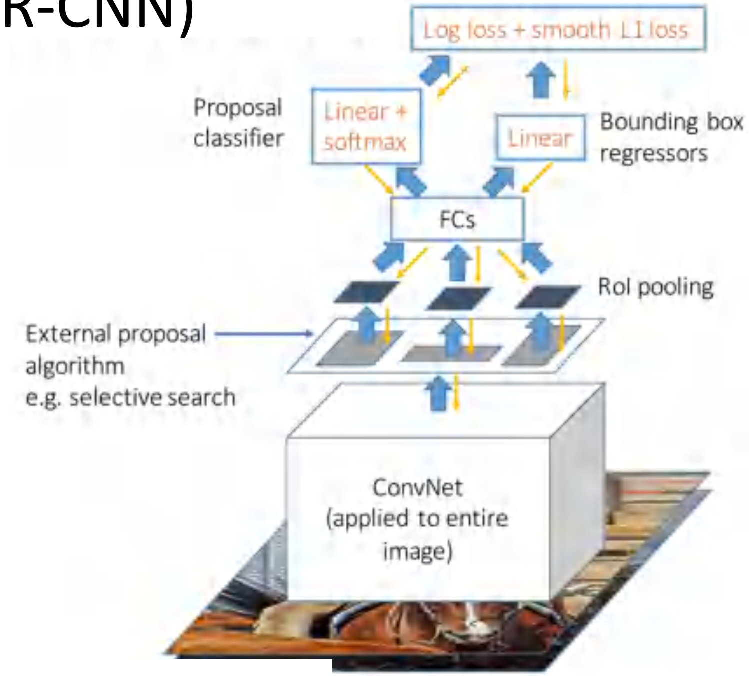
Transfer Learning: Architecture Matters!

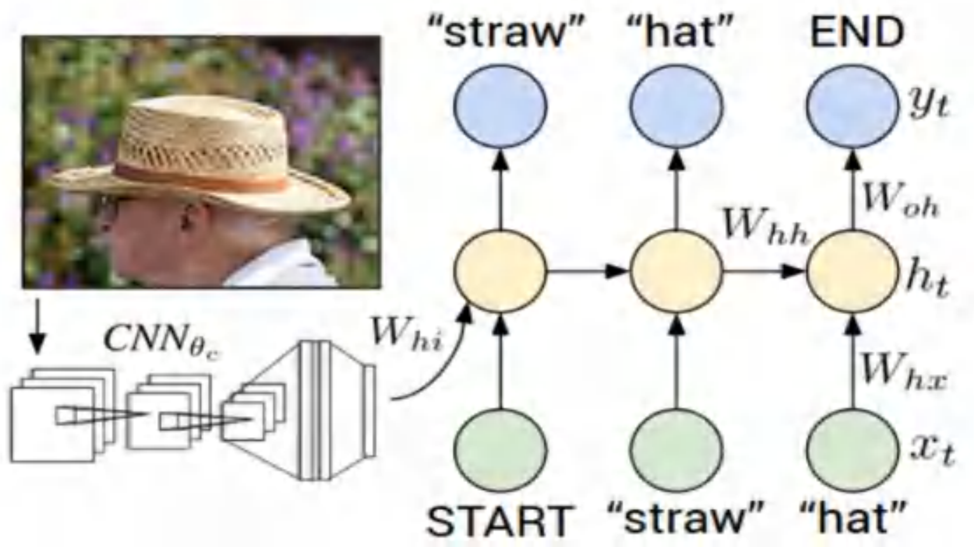


Ross Girshick, "The Generalized R-CNN Framework for Object Detection", ICCV 2017 Tutorial on Instance-Level Visual Recognition

Transfer learning is pervasive! It's the norm, not the exception

Object Detection
(Fast R-CNN)



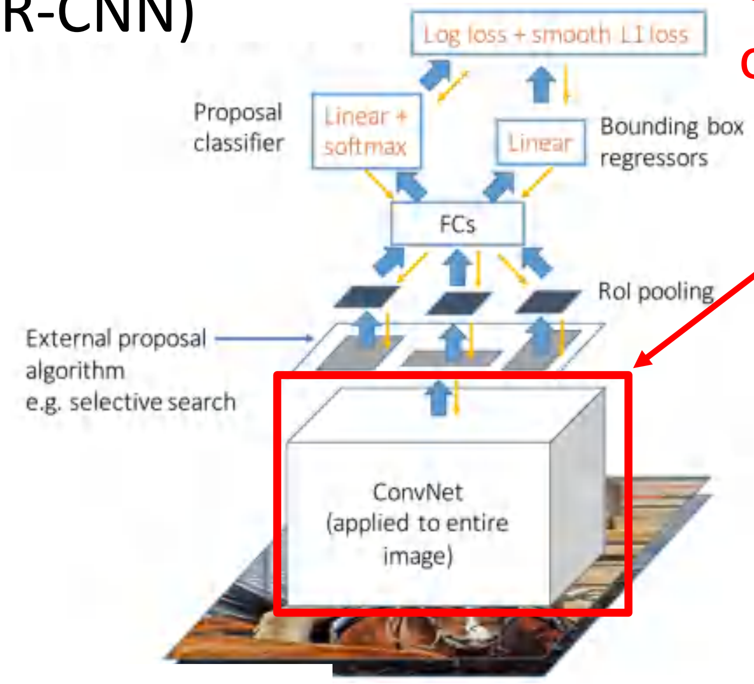


Girshick, "Fast R-CNN", ICCV 2015
Figure copyright Ross Girshick, 2015. Reproduced with permission.

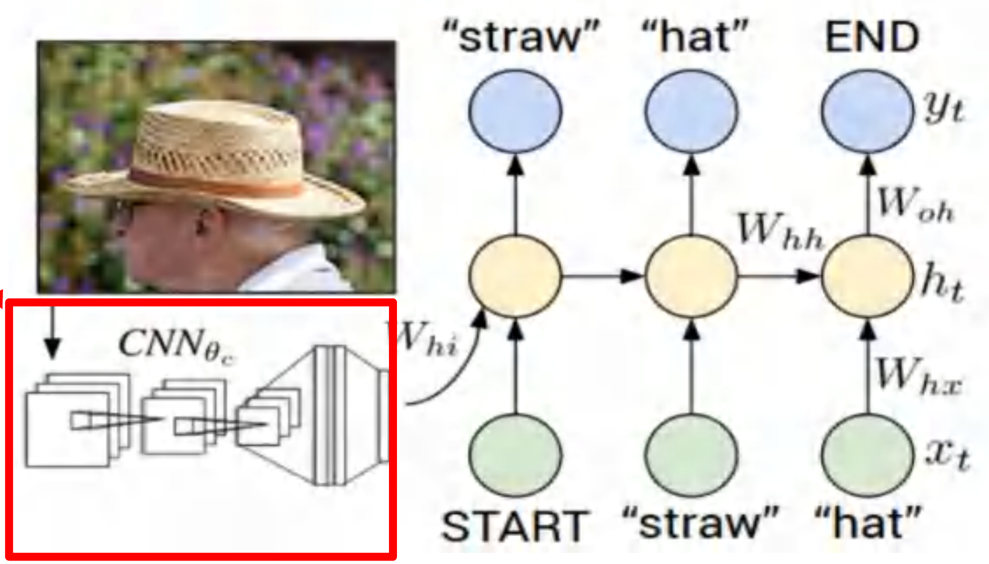
Karpathy and Fei-Fei, "Deep Visual-Semantic Alignments for Generating Image Descriptions", CVPR 2015

Transfer learning is pervasive! It's the norm, not the exception

Object Detection
(Fast R-CNN)



CNN pretrained on ImageNet



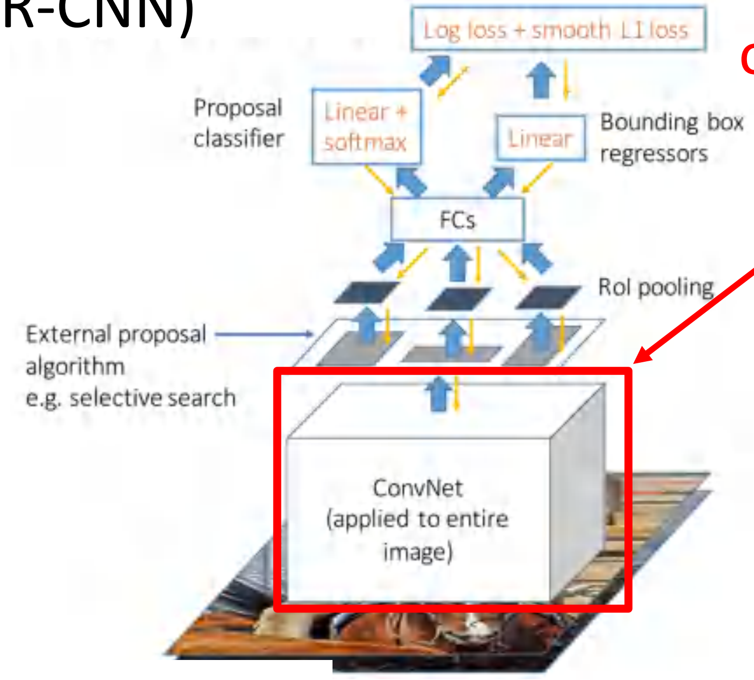
Girshick, "Fast R-CNN", ICCV 2015
Figure copyright Ross Girshick, 2015. Reproduced with permission.

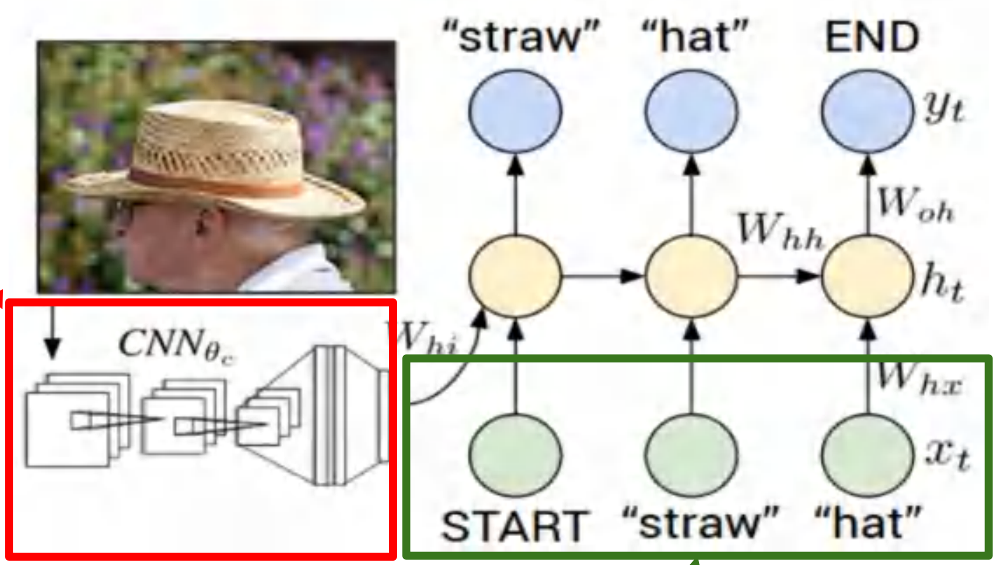
Karpathy and Fei-Fei, "Deep Visual-Semantic Alignments for Generating Image Descriptions", CVPR 2015

Transfer learning is pervasive!
It’s the norm, not the exception

Object Detection
(Fast R-CNN)

CNN pretrained
on ImageNet



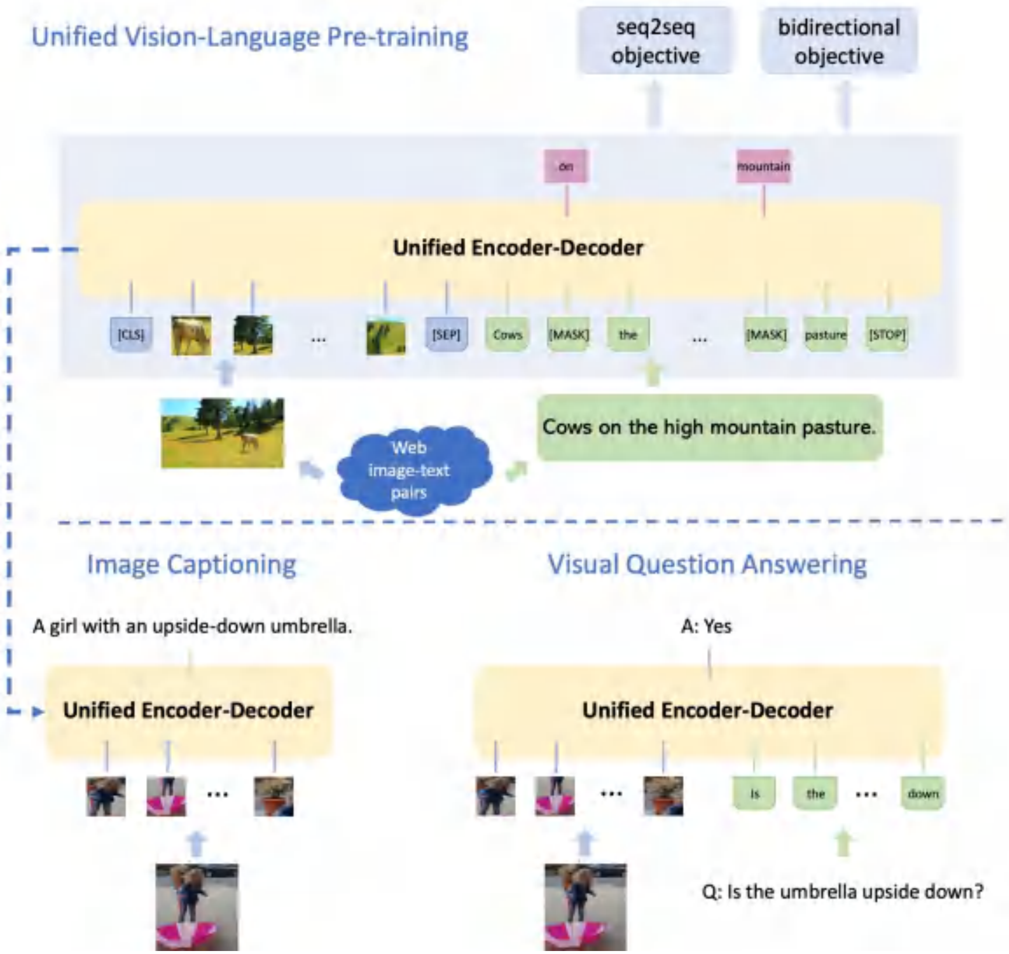


Word vectors pretrained
with word2vec

Girshick, “Fast R-CNN”, ICCV 2015
Figure copyright Ross Girshick, 2015. Reproduced with permission.

Karpathy and Fei-Fei, “Deep Visual-Semantic Alignments for Generating Image Descriptions”, CVPR 2015

Transfer learning is pervasive! It's the norm, not the exception

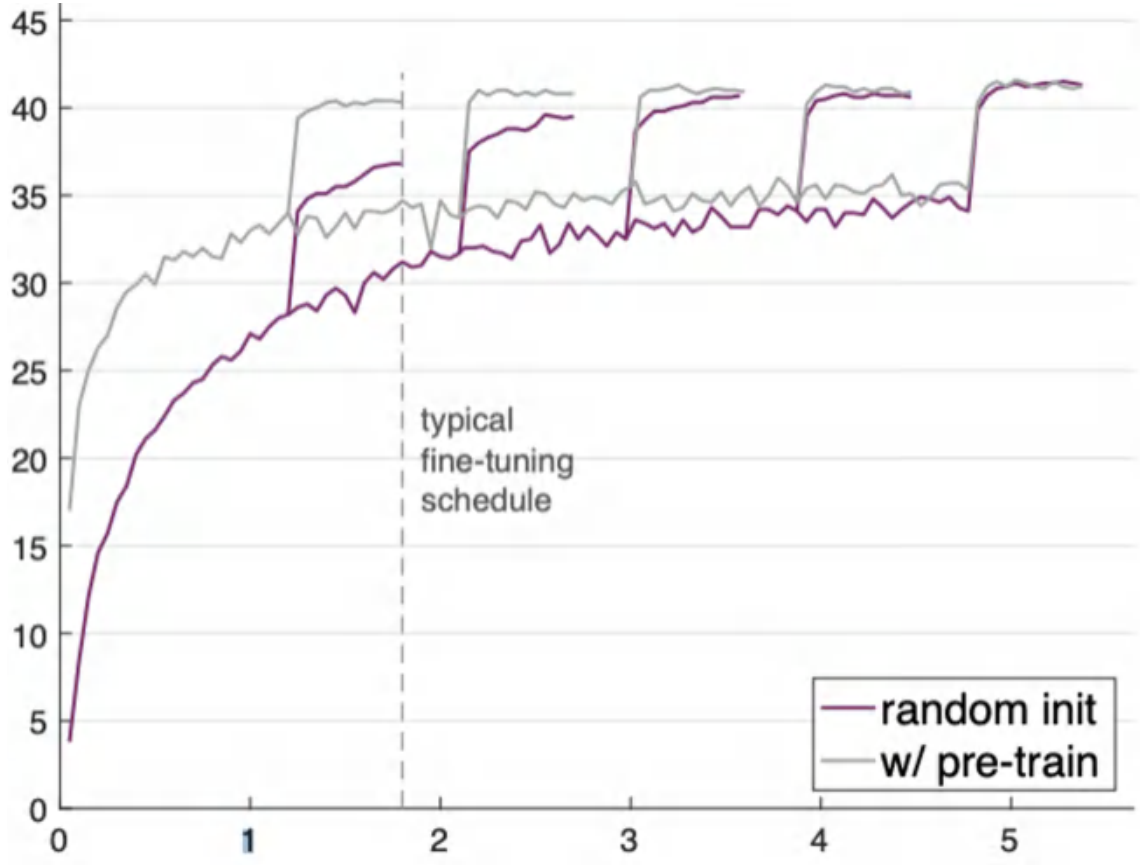


1. Train CNN on ImageNet
2. Fine-Tune (1) for object detection on Visual Genome
3. Train BERT language model on lots of text
4. Combine (2) and (3), train for joint image / language modeling
5. Fine-tune (5) for image captioning, visual question answering, etc.

Zhou et al, "Unified Vision-Language Pre-Training for Image Captioning and VQA", AAAI 2020

Transfer Learning can help you converge faster

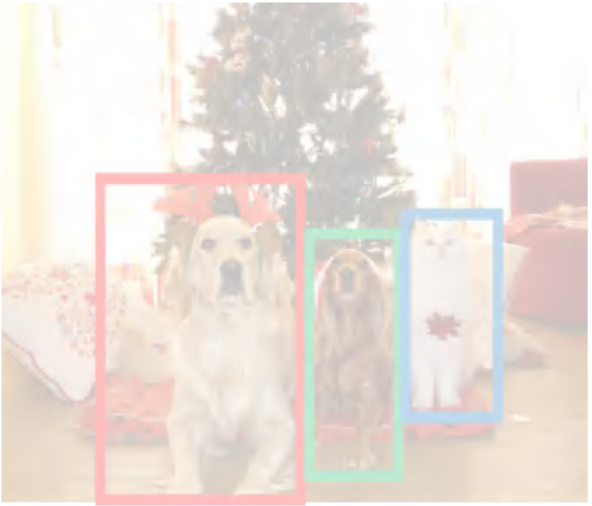
COCO object detection



If you have enough data and train for much longer, random initialization can sometimes do as well as transfer learning

He et al, "Rethinking ImageNet Pre-Training", ICCV 2019

Classification: Transferring to New 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

This image is CC0 public domain

This Week: Object Detection

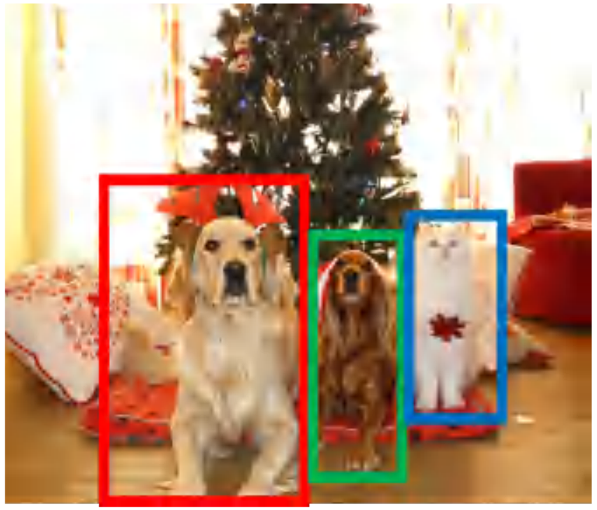
Classification

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

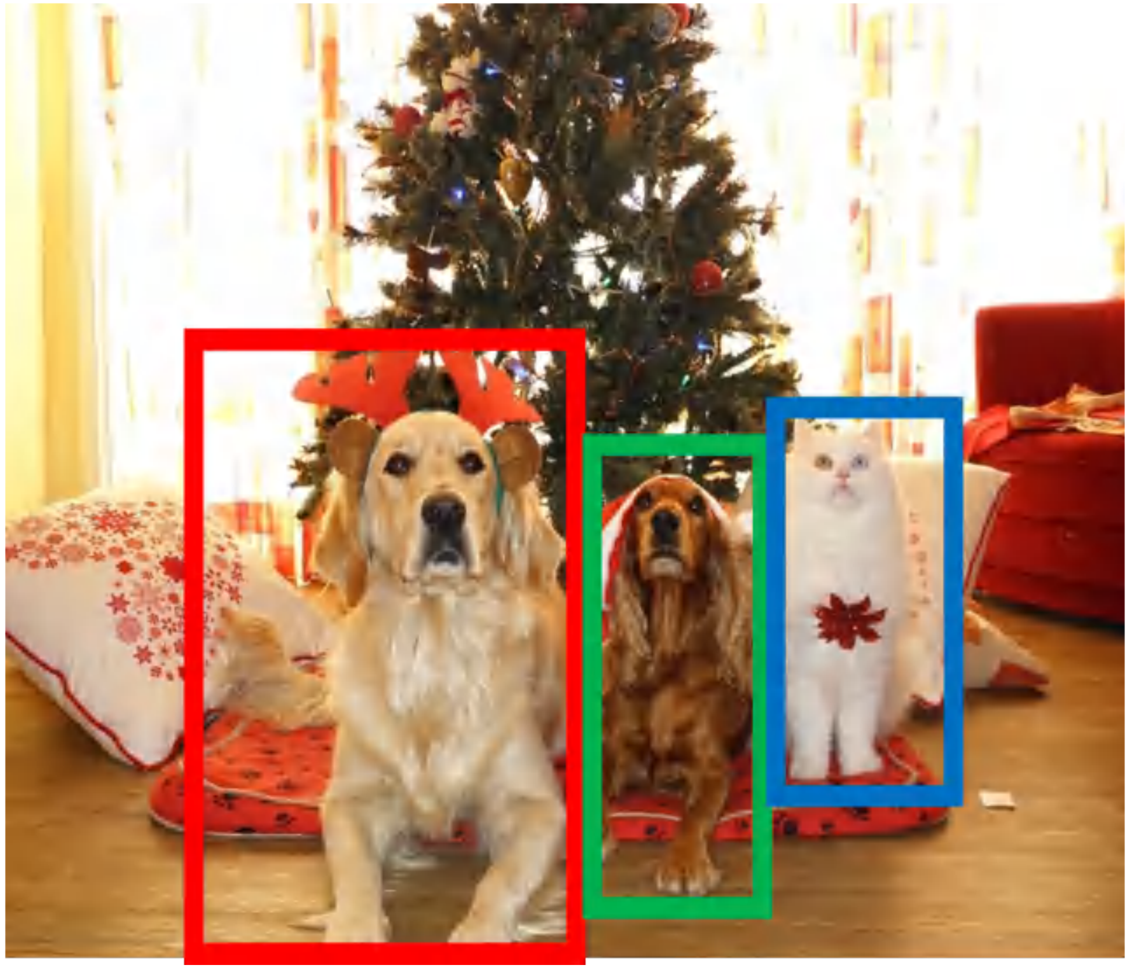
This image is CC0 public domain

Object Detection: Task Definition

Input: Single RGB Image

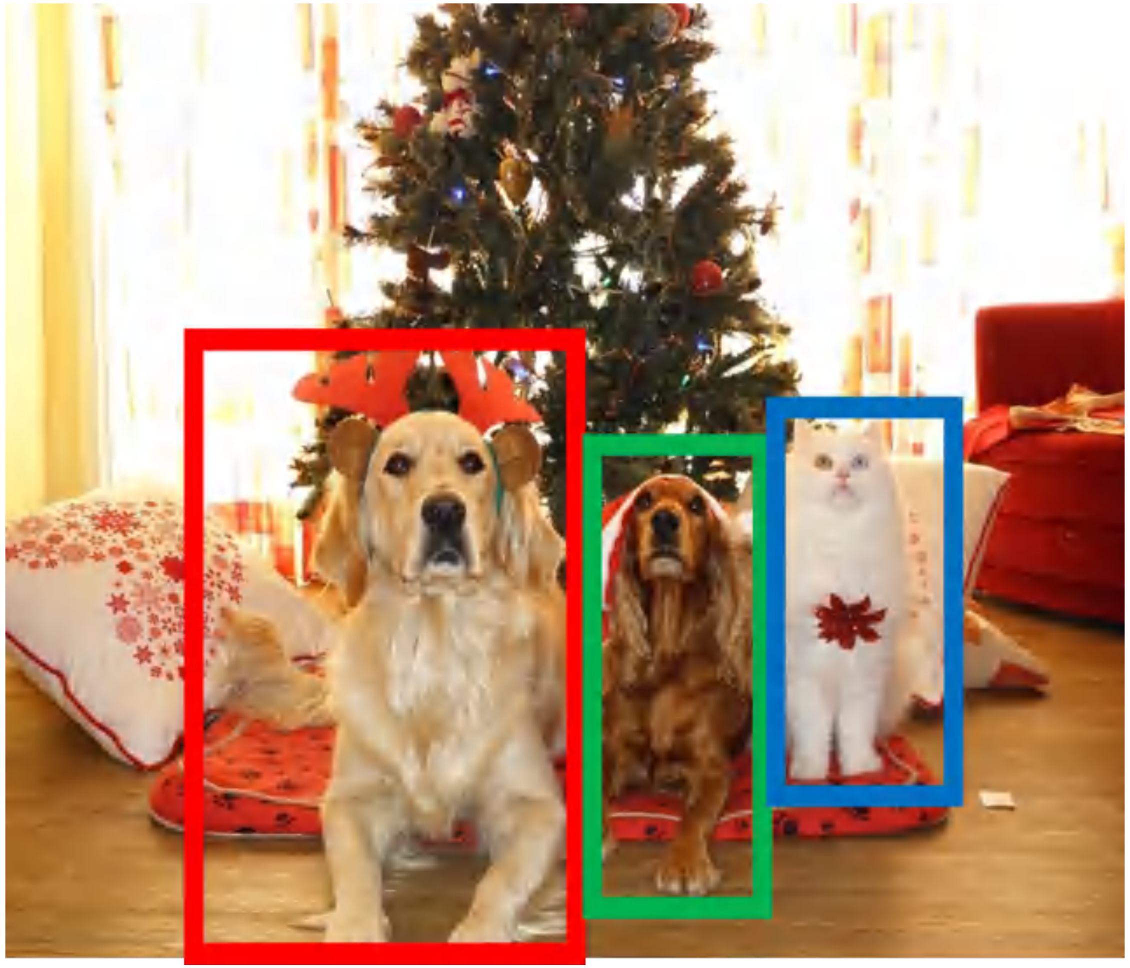
Output: A <u>set</u> of detected objects;
For each object predict:

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



Object Detection: Challenges

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



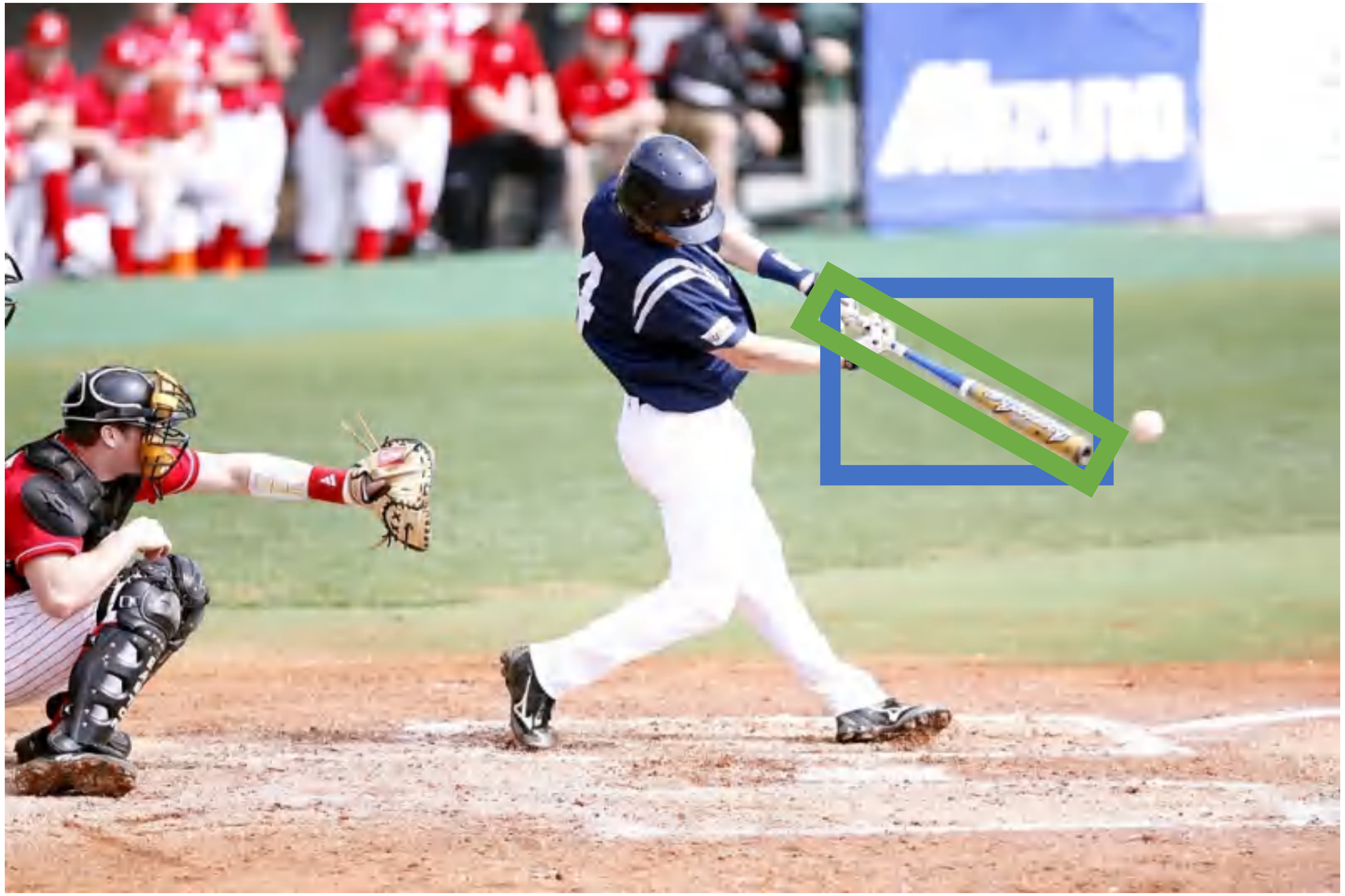
Bounding Boxes

Bounding boxes are typically *axis-aligned*

Bounding Boxes

Bounding boxes are typically *axis-aligned*

Oriented boxes are much less common



Object Detection: Modal vs Amodal Boxes

Bounding boxes (usually) cover only the visible portion of the object

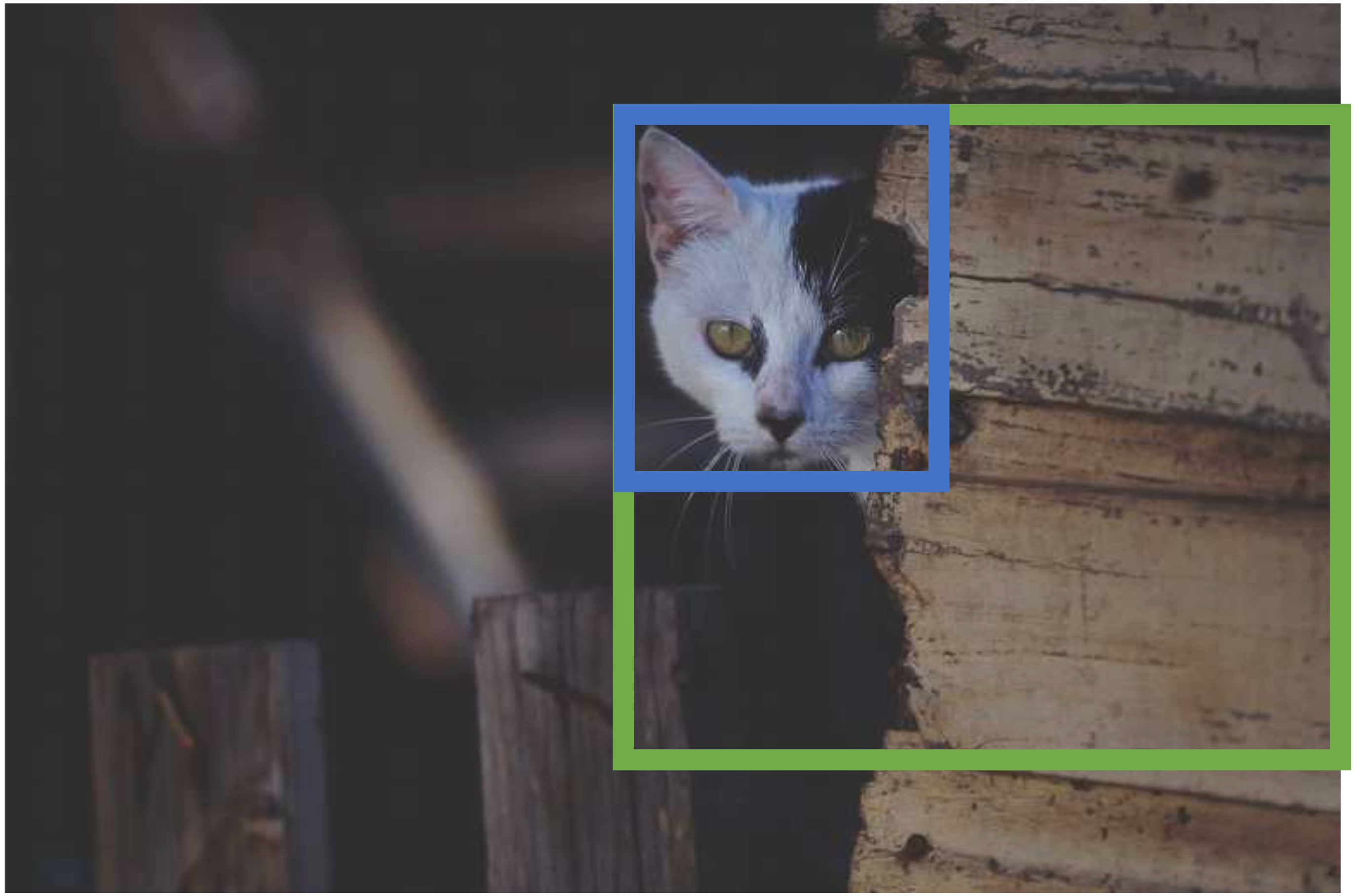
Zhu et al, "Semantic Amodal Segmentation", CVPR 2017

This image is CC0 Public Domain

Object Detection: Modal vs Amodal Boxes

Bounding boxes (usually) cover only the visible portion of the object

<u>Amodal detection:</u> box covers the entire extent of the object, even occluded parts



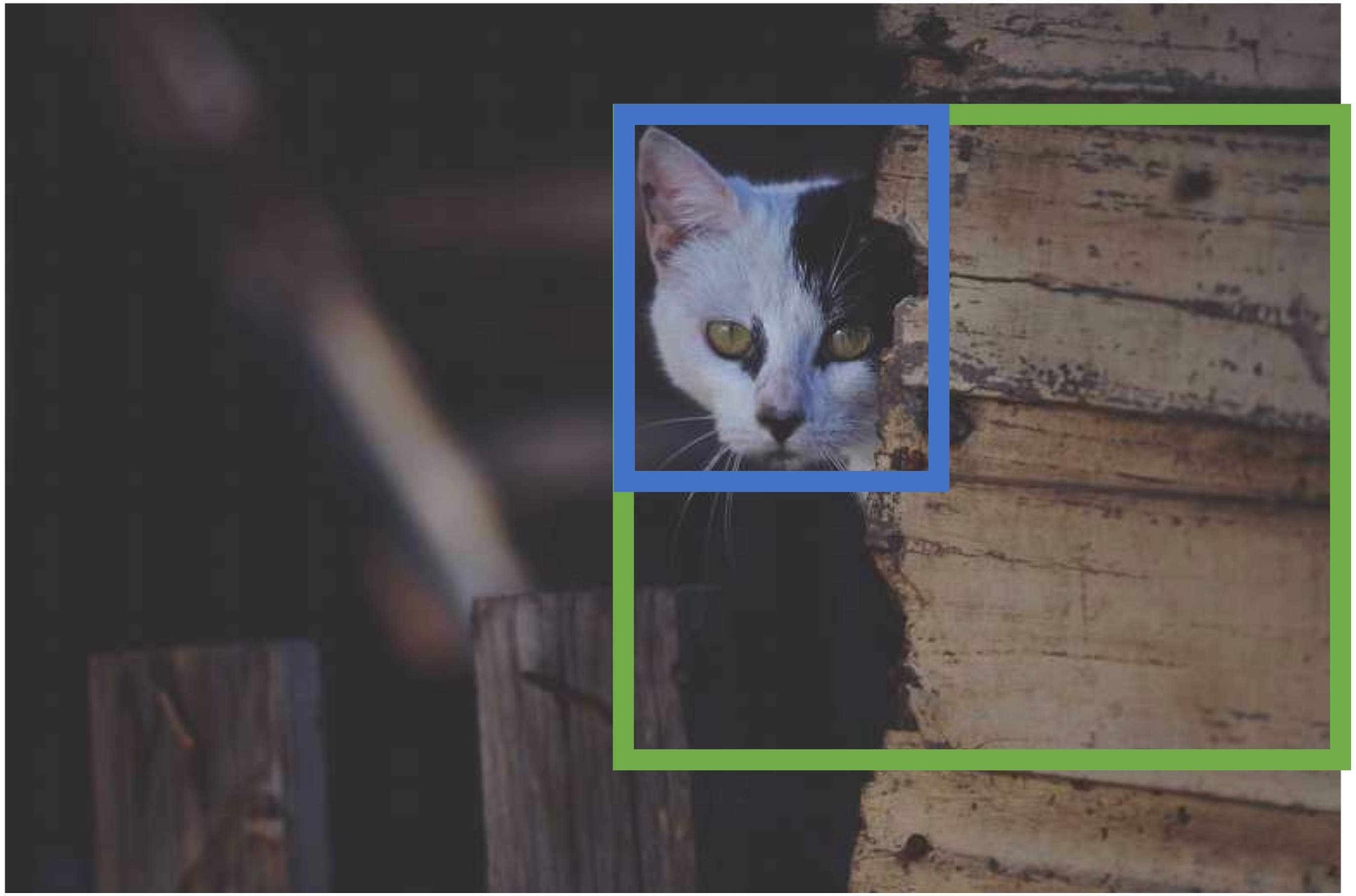
Zhu et al, "Semantic Amodal Segmentation", CVPR 2017

This image is CC0 Public Domain

Object Detection: Modal vs Amodal Boxes

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

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

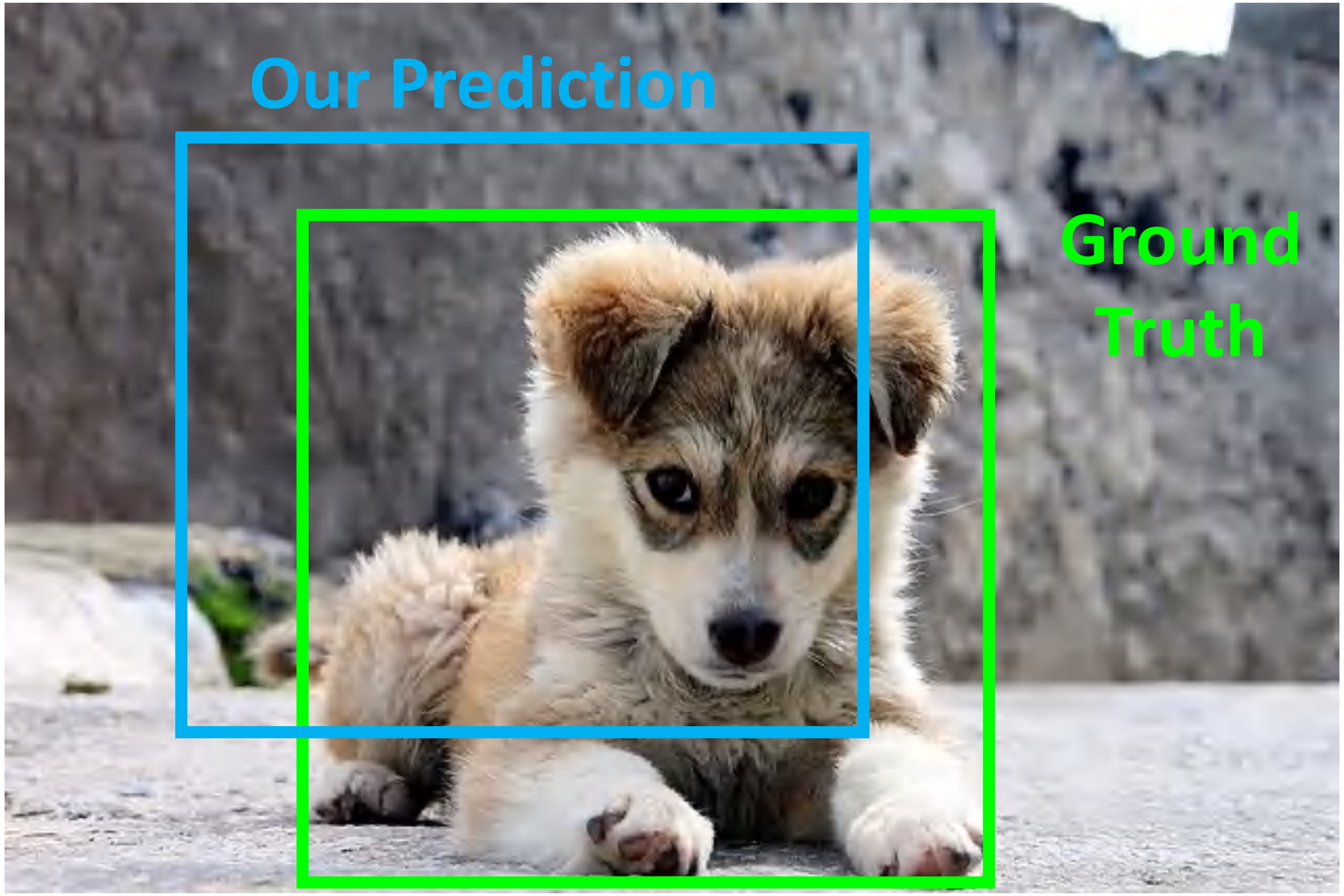


Zhu et al, "Semantic Amodal Segmentation", CVPR 2017

This image is CC0 Public Domain

Comparing Boxes: Intersection over Union (IoU)

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



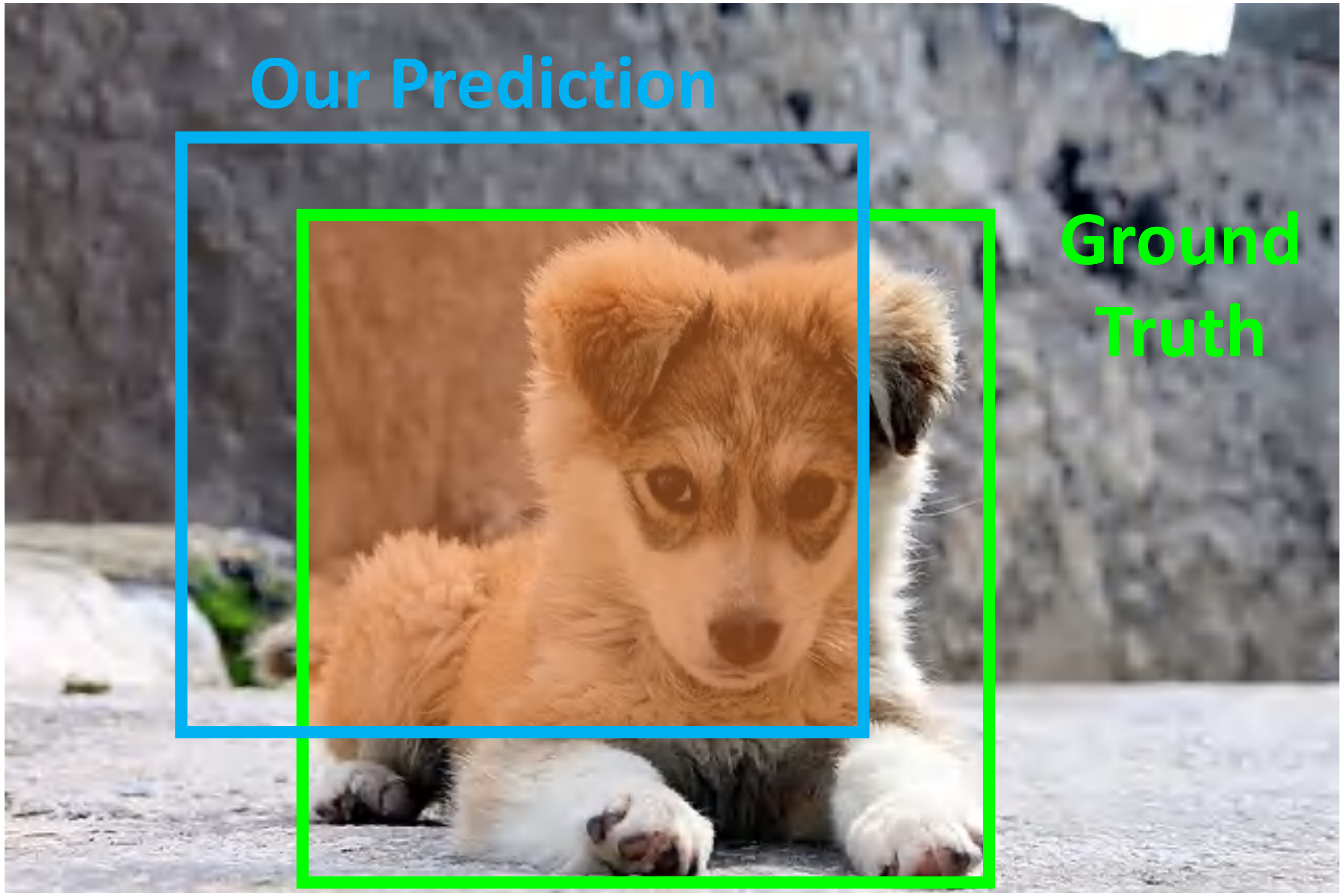
Puppy image is licensed under CC-A 2.0 Generic license. Bounding boxes and text added by Justin Johnson.

Comparing Boxes: Intersection over Union (IoU)

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

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

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



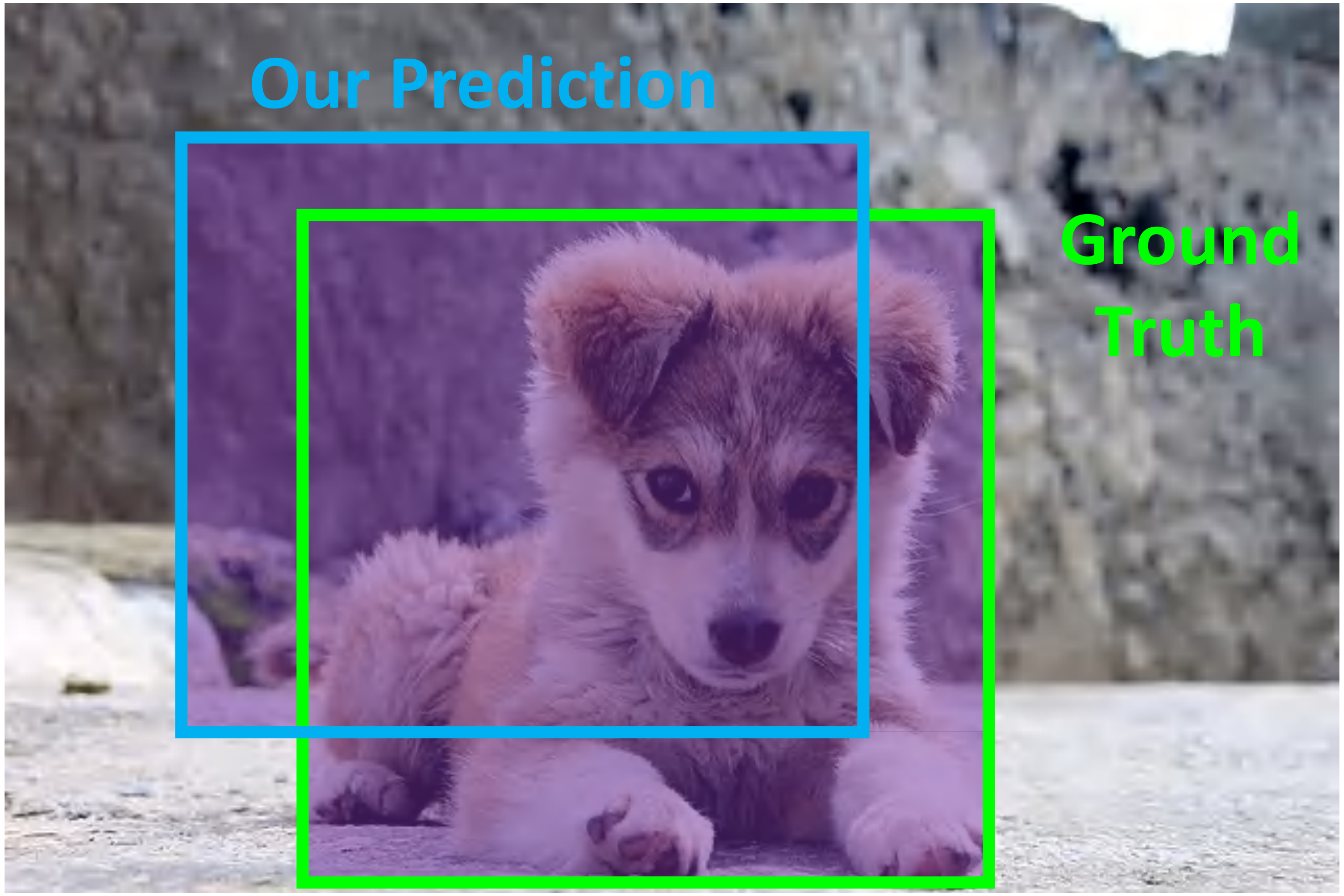
Puppy image is licensed under CC-A 2.0 Generic license. Bounding boxes and text added by Justin Johnson.

Comparing Boxes: Intersection over Union (IoU)

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

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

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



Puppy image is licensed under CC-A 2.0 Generic license. Bounding boxes and text added by Justin Johnson.

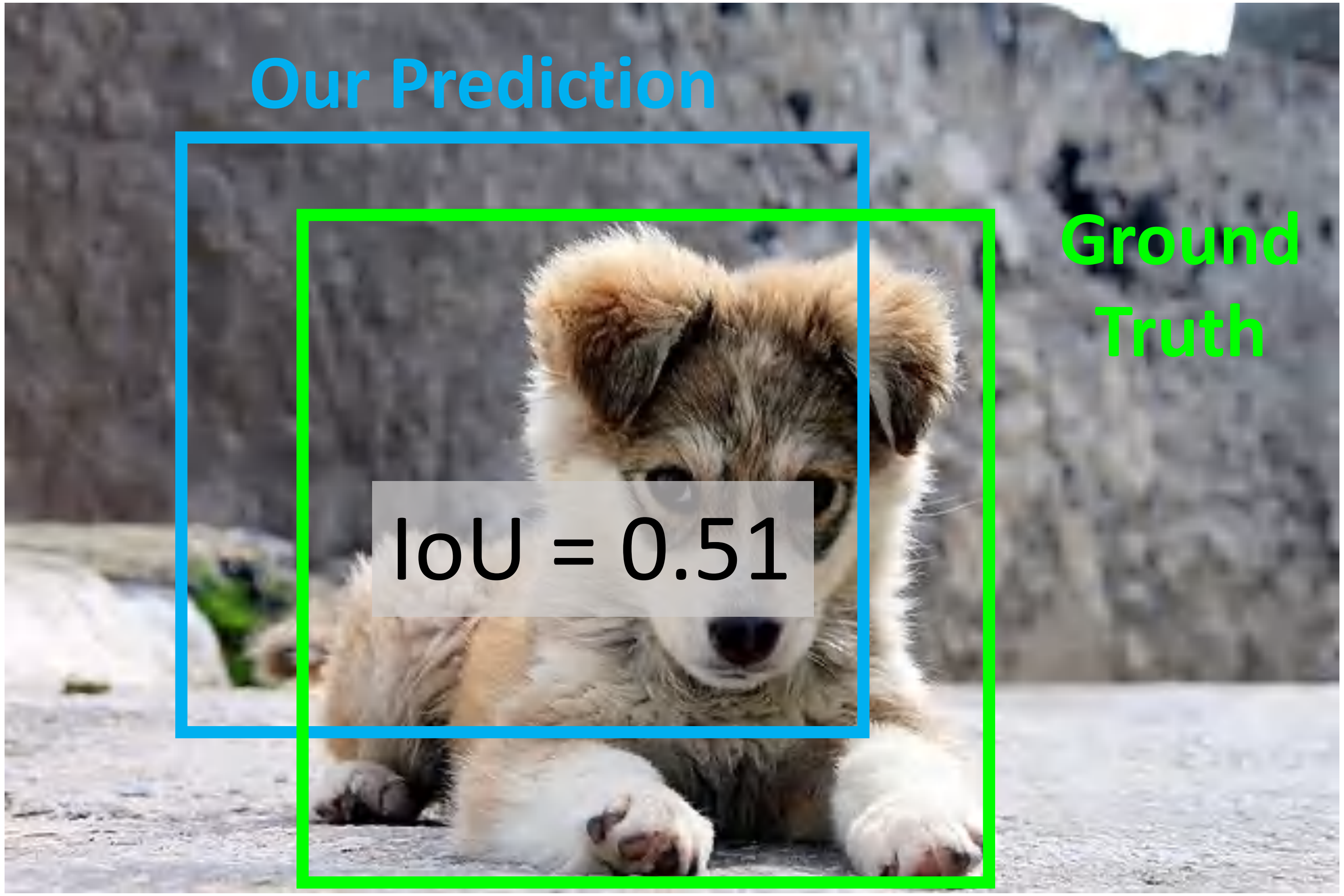
Comparing Boxes: Intersection over Union (IoU)

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

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

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

IoU > 0.5 is "decent"



Puppy image is licensed under CC-A 2.0 Generic license. Bounding boxes and text added by Justin Johnson.

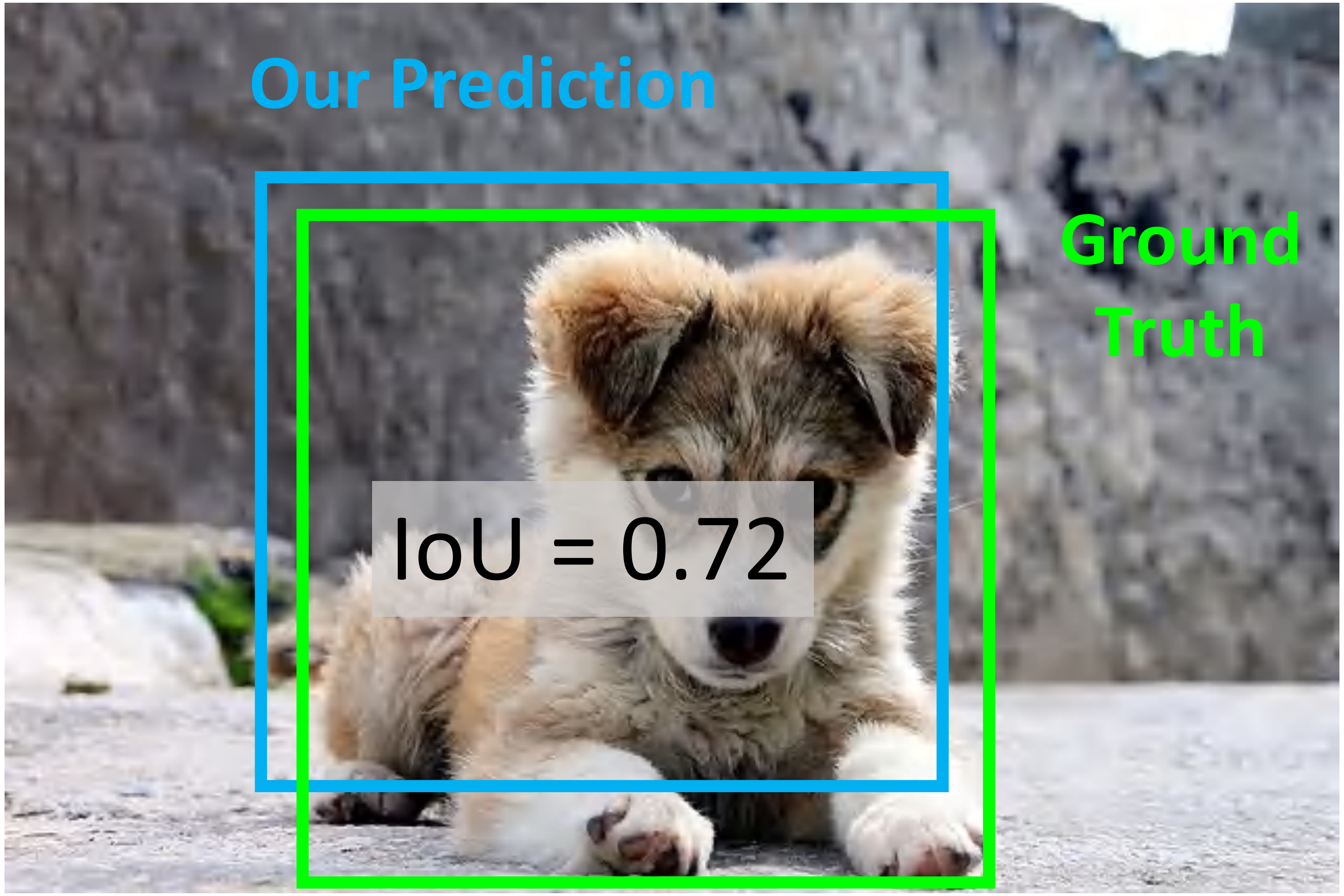
Comparing Boxes: Intersection over Union (IoU)

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

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

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

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



Puppy image is licensed under CC-A 2.0 Generic license. Bounding boxes and text added by Justin Johnson.

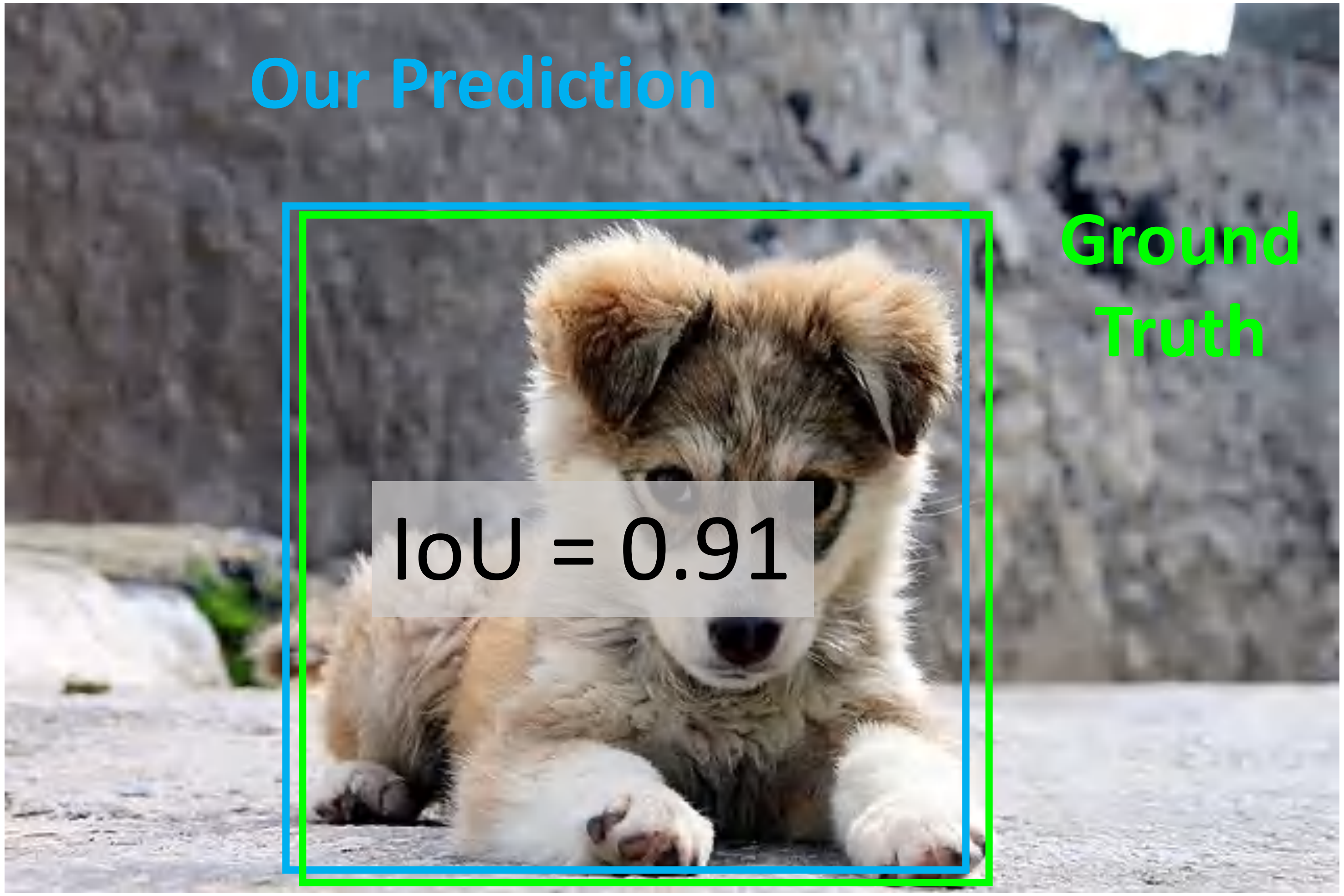
Comparing Boxes: Intersection over Union (IoU)

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

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

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

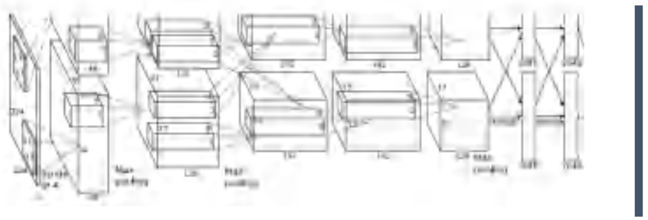
IoU > 0.5 is “decent”,
IoU > 0.7 is “pretty good”,
IoU > 0.9 is “almost perfect”



Puppy image is licensed under CC-A 2.0 Generic license. Bounding boxes and text added by Justin Johnson.

Detecting a single object

This image is CC0 public domain



Vector:
4096

Treat localization as a regression problem!

Detecting a single object "What"

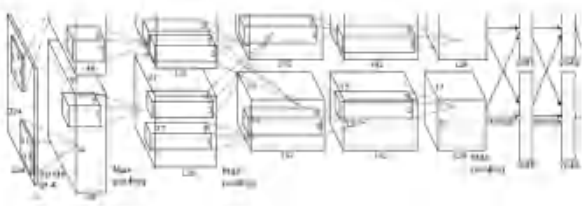
Correct label:
Cat

Fully Connected:
4096 to 1000

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

Softmax Loss

This image is CC0 public domain



Vector:
4096

Treat localization as a regression problem!

Detecting a single object

“What”

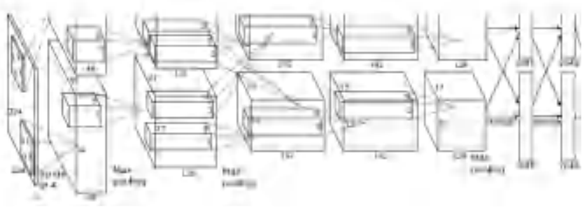
Correct label:
Cat

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

Softmax Loss

Fully Connected:
4096 to 1000

This image is CC0 public domain



Vector:
4096

Fully Connected:
4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

Correct box:
(x’, y’, w’, h’)

Treat localization as a regression problem!

“Where”

Detecting a single object

"What"

Correct label:
Cat

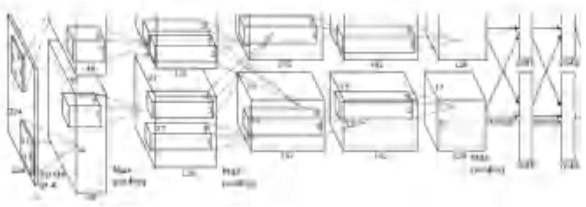
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Fully Connected:
4096 to 1000

Softmax Loss

Multitask Loss

This image is CC0 public domain



Vector:
4096

Weighted Sum → **Loss**

$$L = L_{cls} + \lambda L_{reg}$$

Fully Connected:
4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

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

Treat localization as a regression problem!

"Where"

Detecting a single object

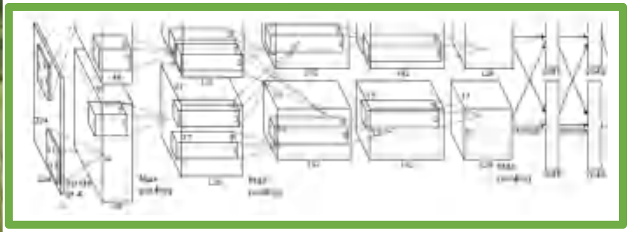
"What"

Correct label:

Cat

Often pretrained on ImageNet (Transfer learning)

This image is CC0 public domain



Vector:

4096

Fully Connected: 4096 to 1000

Class Scores

Cat: 0.9

Dog: 0.05

Car: 0.01

...

Softmax Loss

Multitask Loss

Weighted Sum

Loss

$L = L_{cls} + \lambda L_{reg}$

Fully Connected: 4096 to 4

Box Coordinates (x, y, w, h)

L2 Loss

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

Treat localization as a regression problem!

"Where"

Detecting a single object

"What"

Correct label:
Cat

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

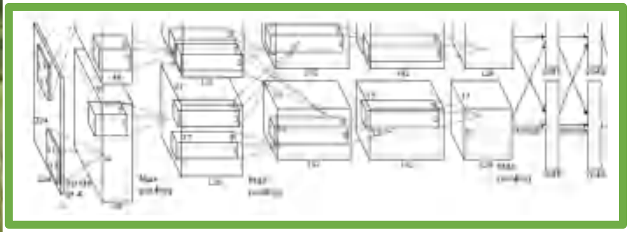
Softmax Loss

Often pretrained on ImageNet (Transfer learning)

Fully Connected: 4096 to 1000

Multitask Loss

This image is CC0 public domain



Vector: 4096

Weighted Sum → **Loss**

$$L = L_{cls} + \lambda L_{reg}$$

Fully Connected: 4096 to 4

Box Coordinates
(x, y, w, h)

L2 Loss

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

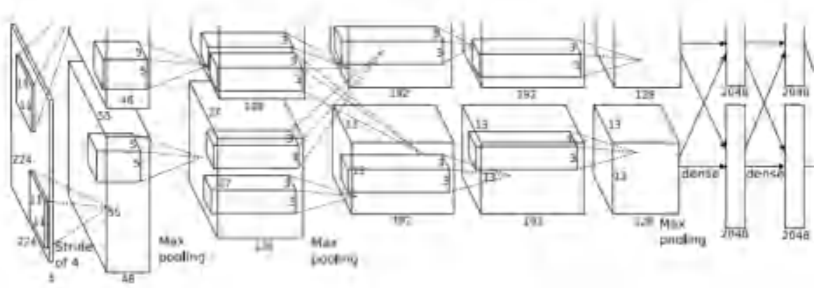
Treat localization as a regression problem!

Problem: Images can have more than one object!

"Where"

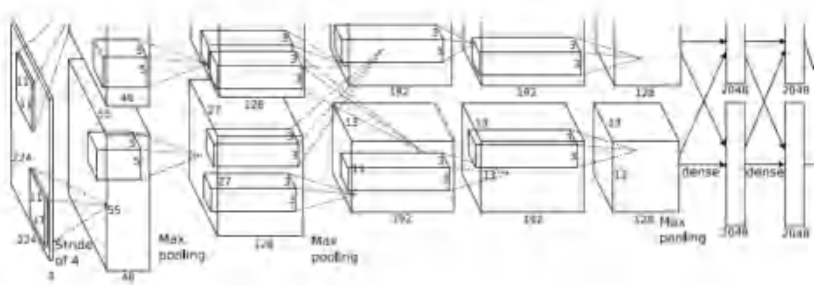
Detecting Multiple Objects

Need different numbers of outputs per image

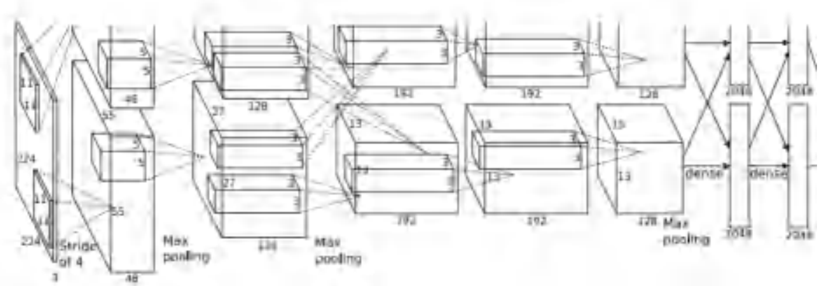
CAT: (x, y, w, h)

4 numbers

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

12 numbers

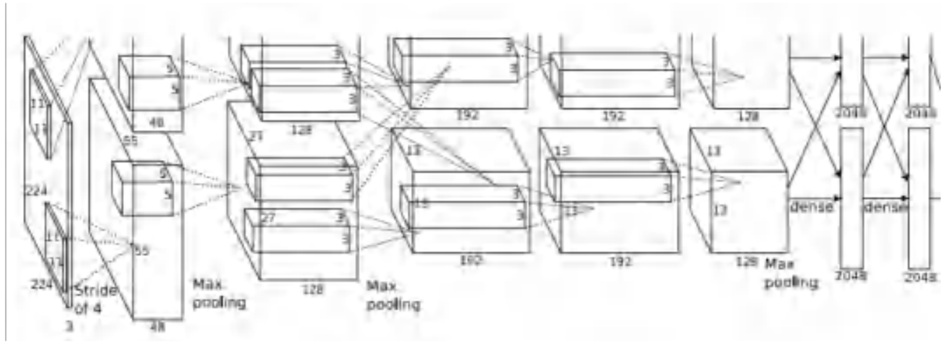
DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

Many numbers!

Duck image is free to use under the Pixabay license

Detecting Multiple Objects: **Sliding Window**

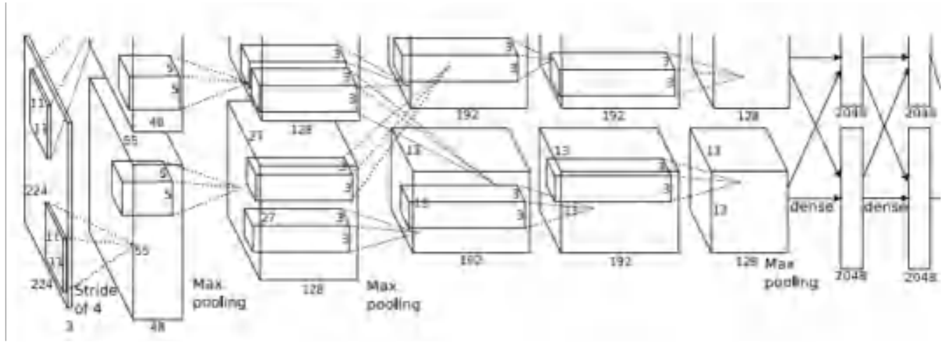
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? NO
Background? YES

Detecting Multiple Objects: **Sliding Window**

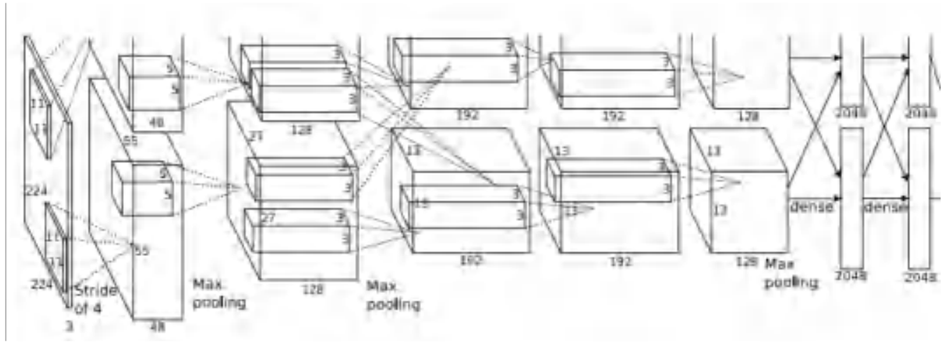
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

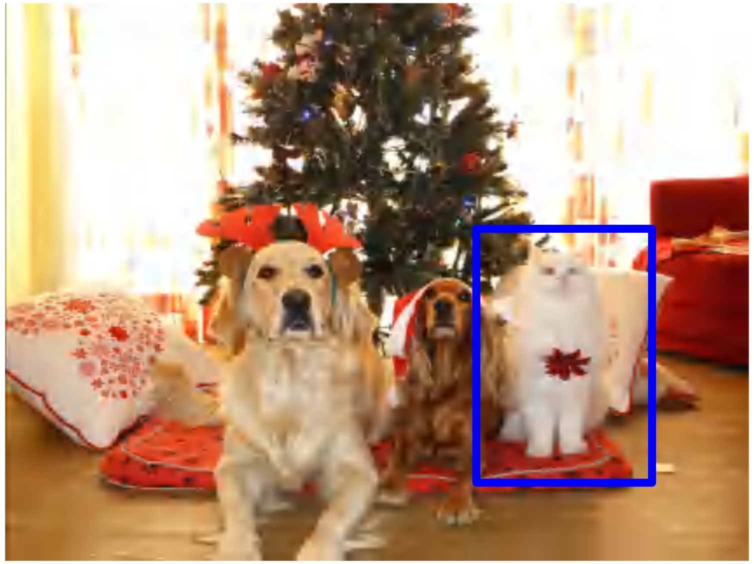
Detecting Multiple Objects: **Sliding Window**

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

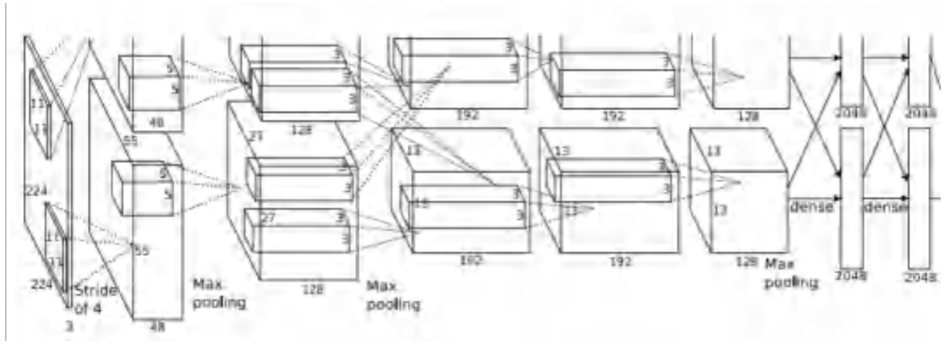


Dog? YES
Cat? NO
Background? NO

Detecting Multiple Objects: **Sliding Window**

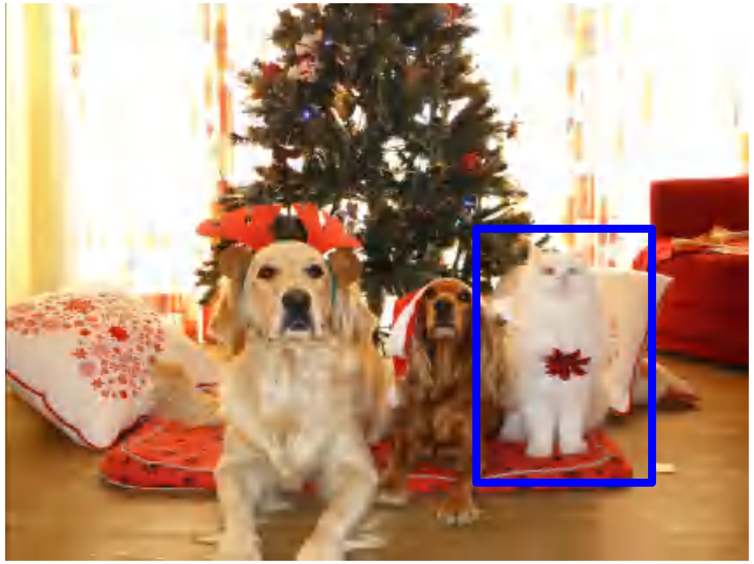


Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? YES
Background? NO

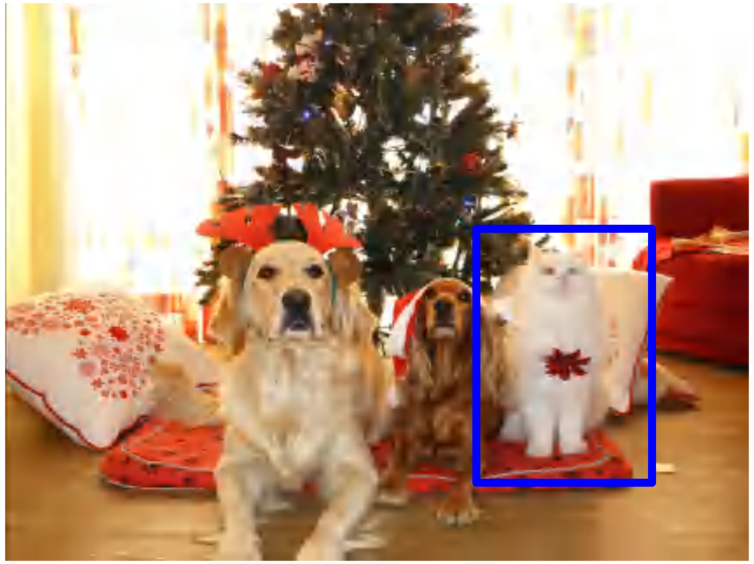
Detecting Multiple Objects: **Sliding Window**



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

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

Detecting Multiple Objects: **Sliding Window**

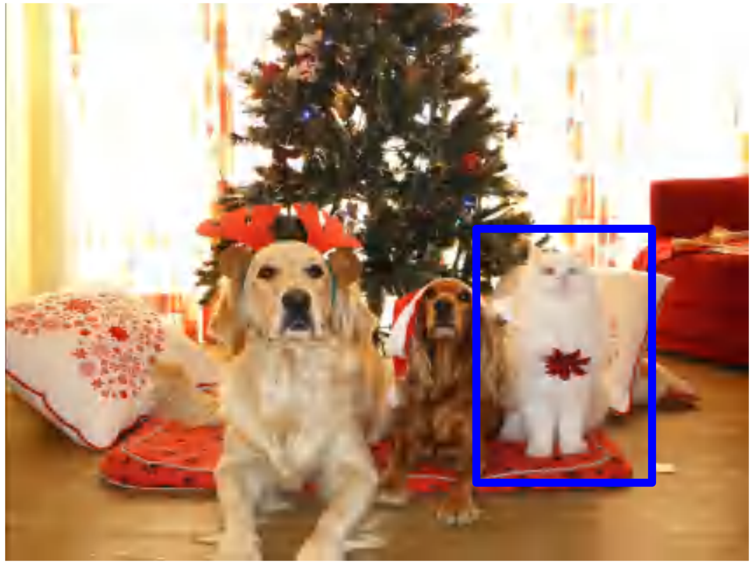


Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

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

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

Detecting Multiple Objects: **Sliding Window**



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

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

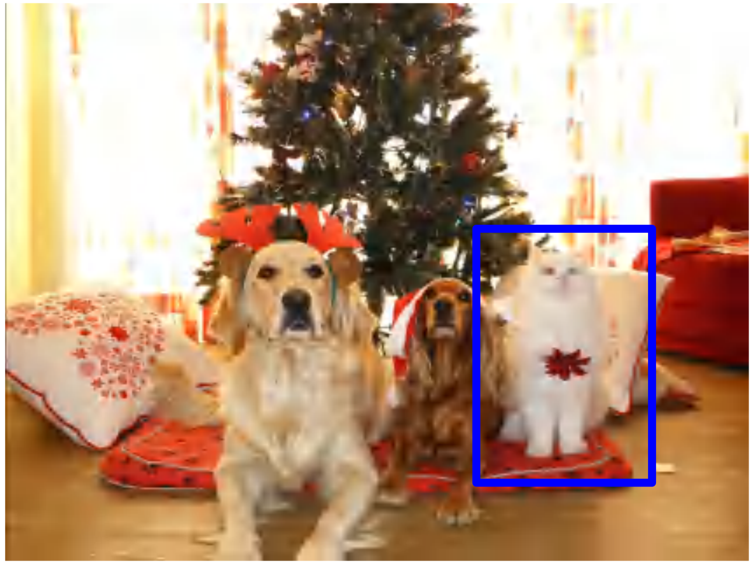
Consider a box of size h x w:
Possible x positions: W – w + 1
Possible y positions: H – h + 1
Possible positions:
(W – w + 1) * (H – h + 1)

Total possible boxes:

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

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

Detecting Multiple Objects: **Sliding Window**



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

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

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

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

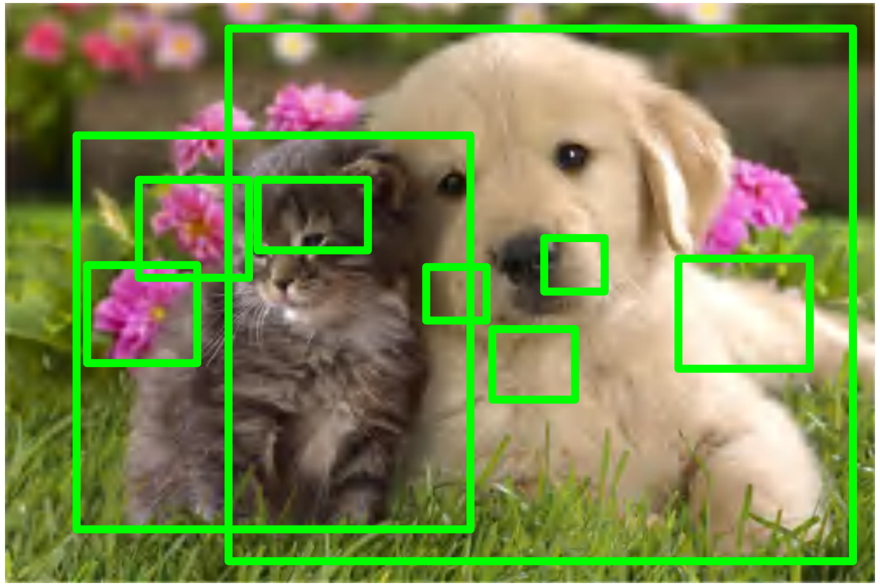
Total possible boxes:

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

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

Region Proposals

- Find a small set of boxes that are likely to cover all objects
- Often based on heuristics: e.g. look for “blob-like” image regions
- Relatively fast to run; e.g. Selective Search gives 2000 region proposals in a few seconds on CPU



Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014

R-CNN: Region-Based CNN

Input image

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

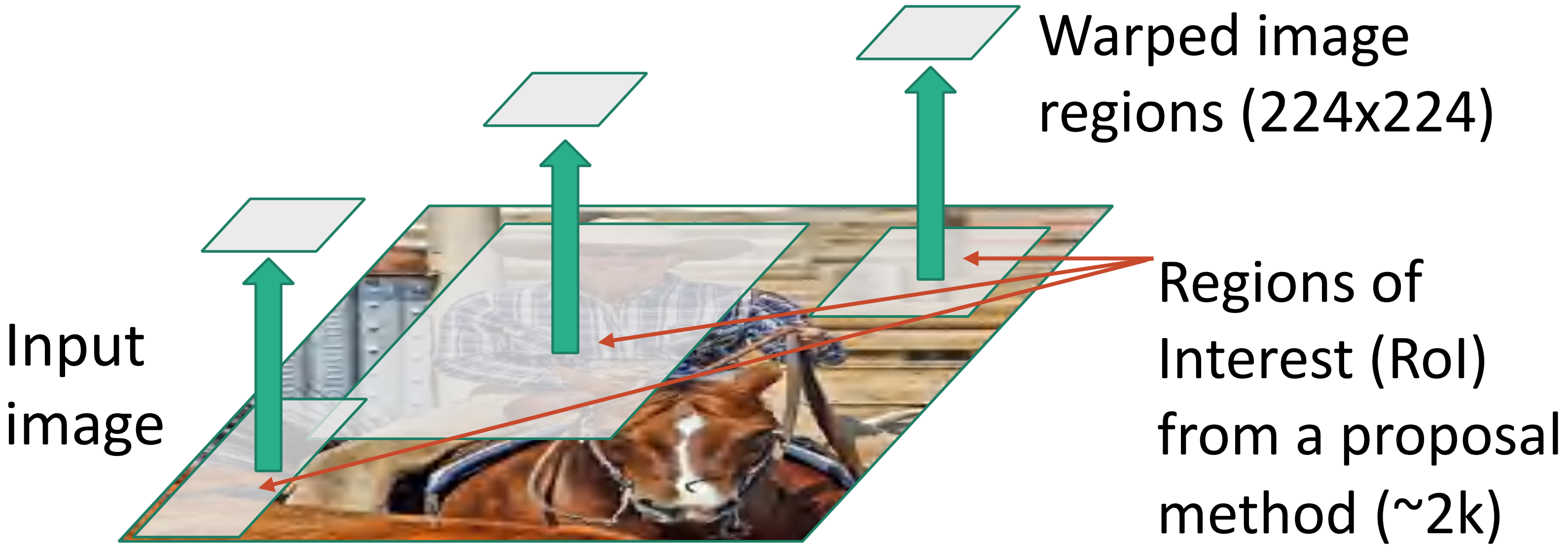
R-CNN: Region-Based CNN

Input image

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

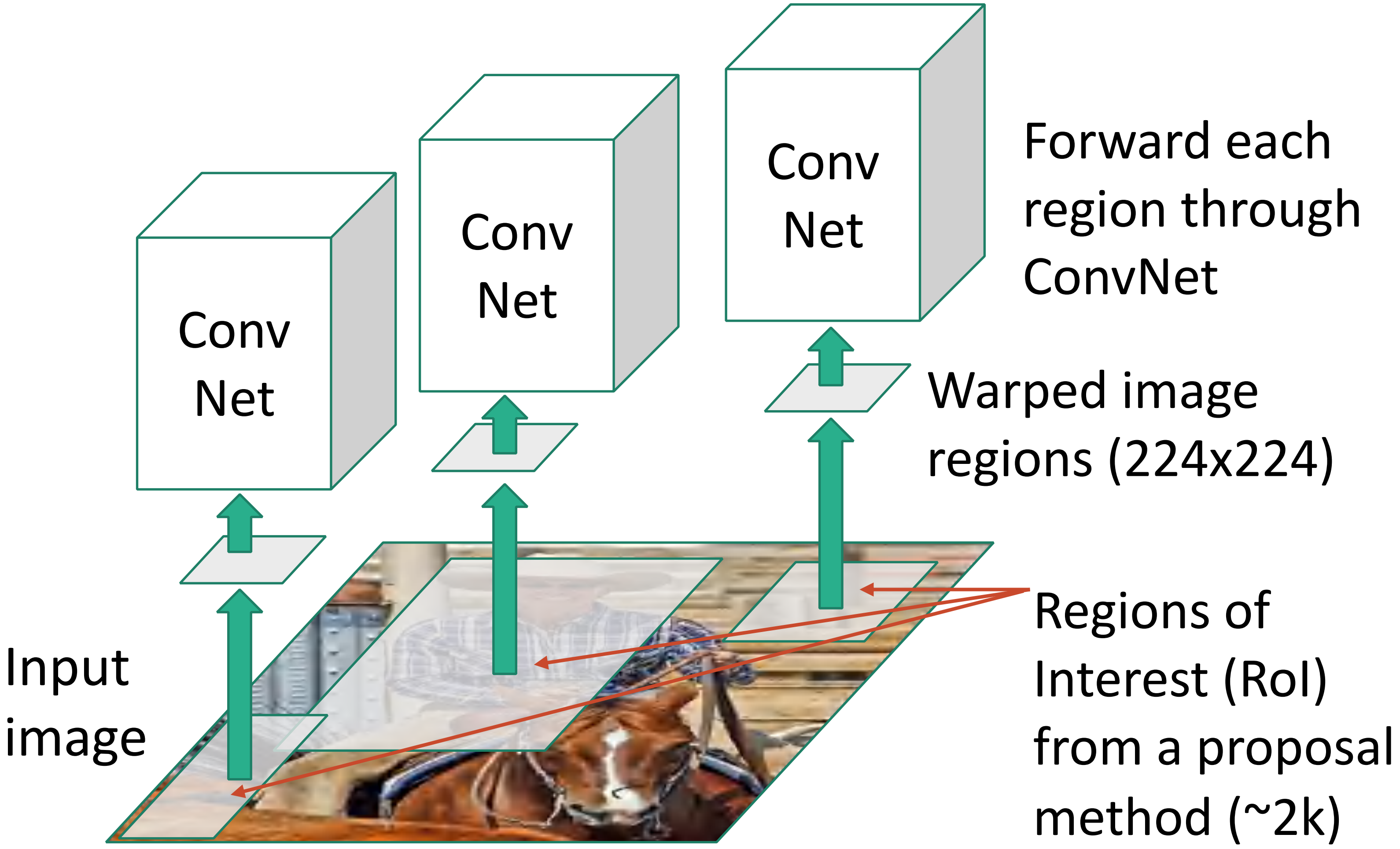
Girshick et al, “Rich feature hierarchies for accurate object detection and semantic segmentation”, CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN: Region-Based CNN



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

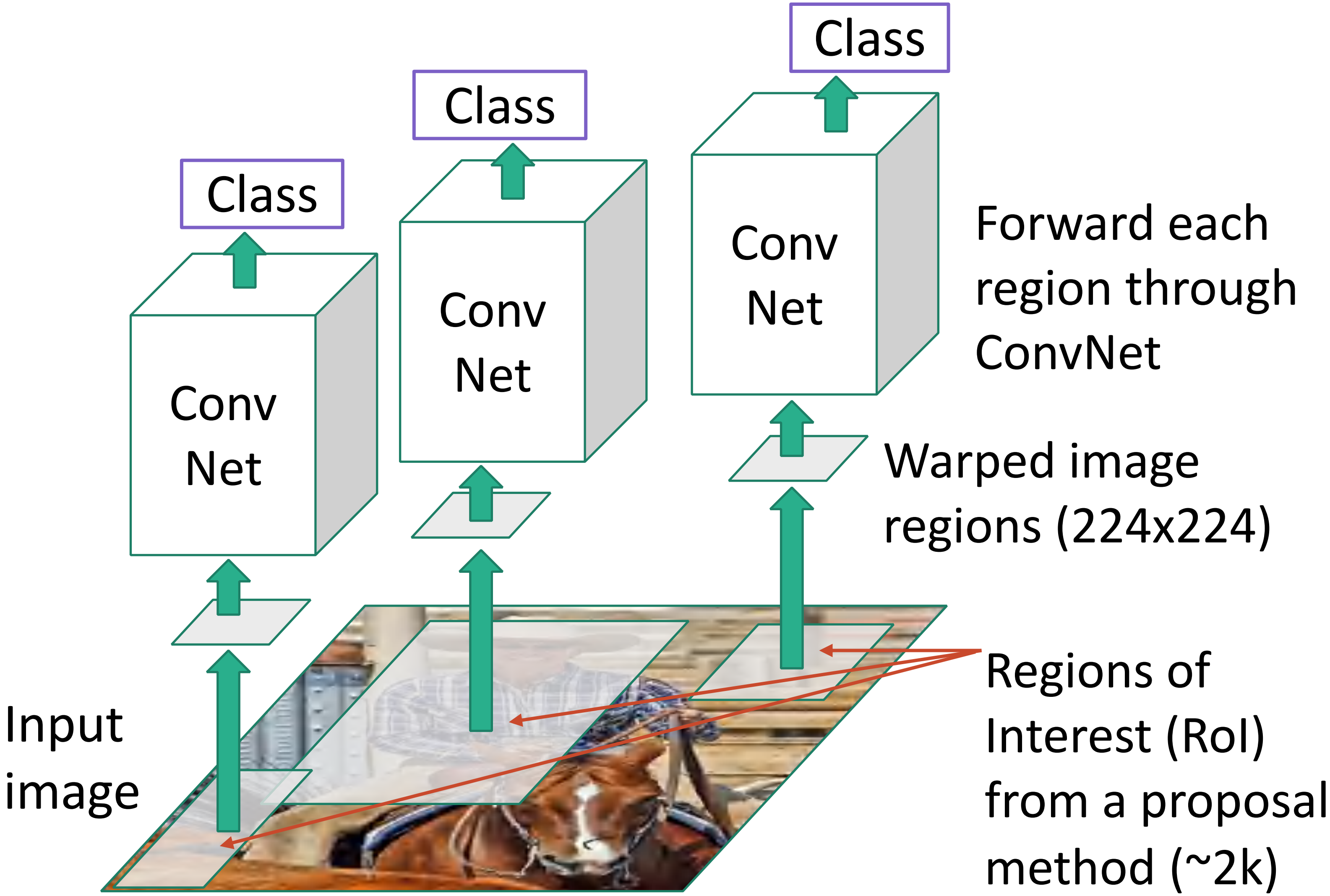
R-CNN: Region-Based CNN



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

R-CNN: Region-Based CNN

Classify each region



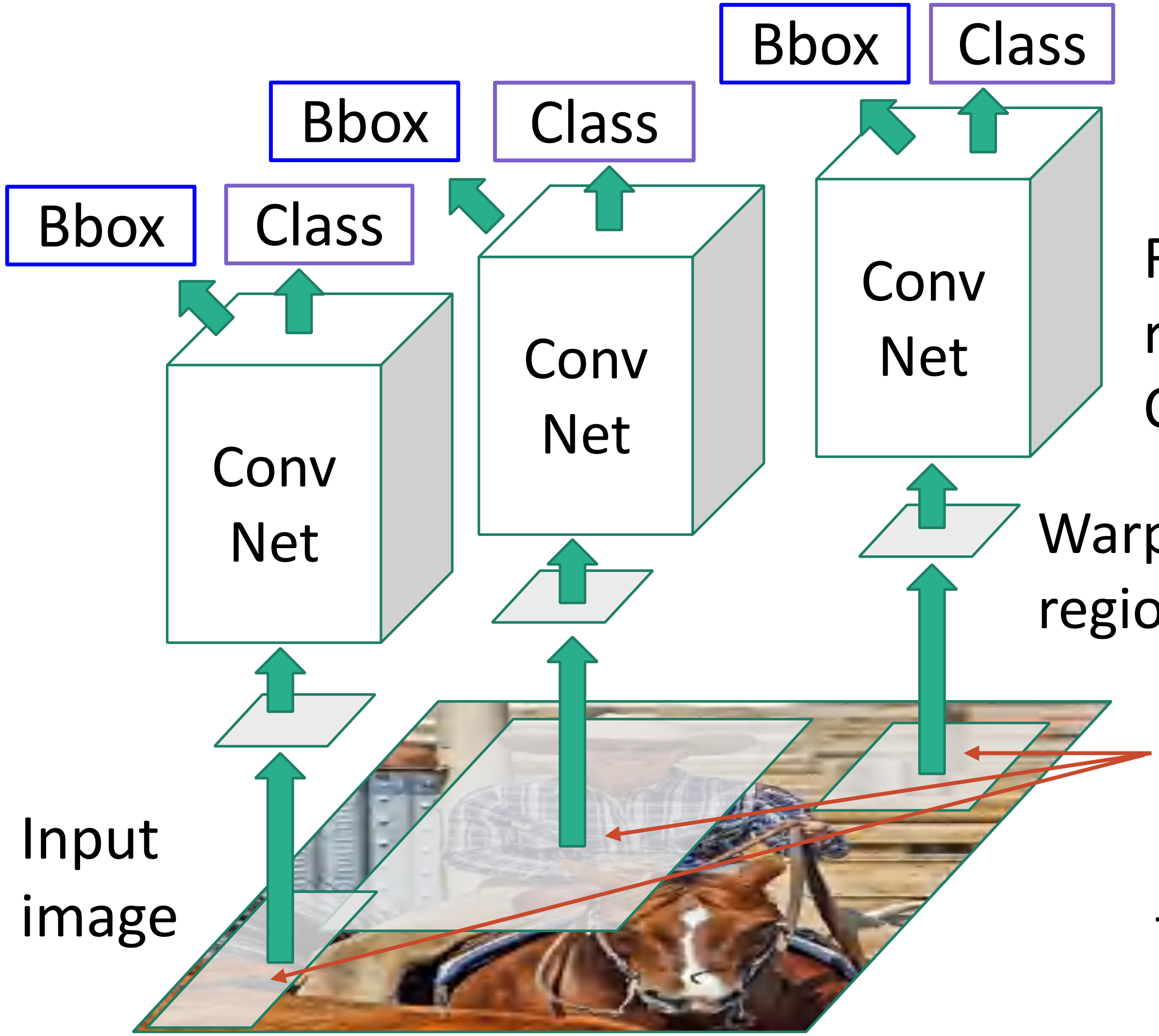
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.

Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN: Region-Based CNN

Classify each region

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



Forward each region through ConvNet

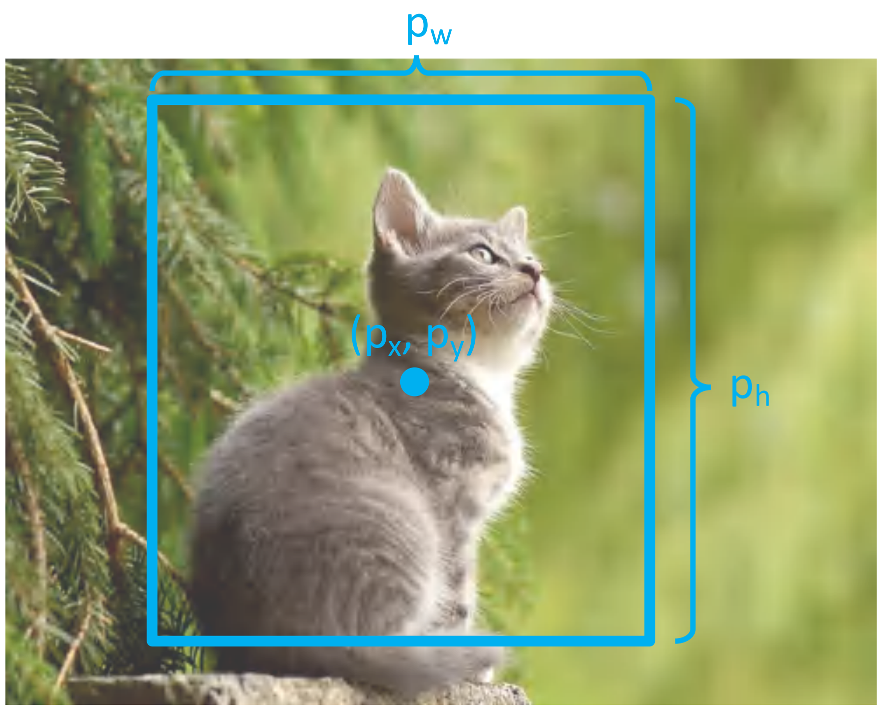
Warped image regions (224x224)

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

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

Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

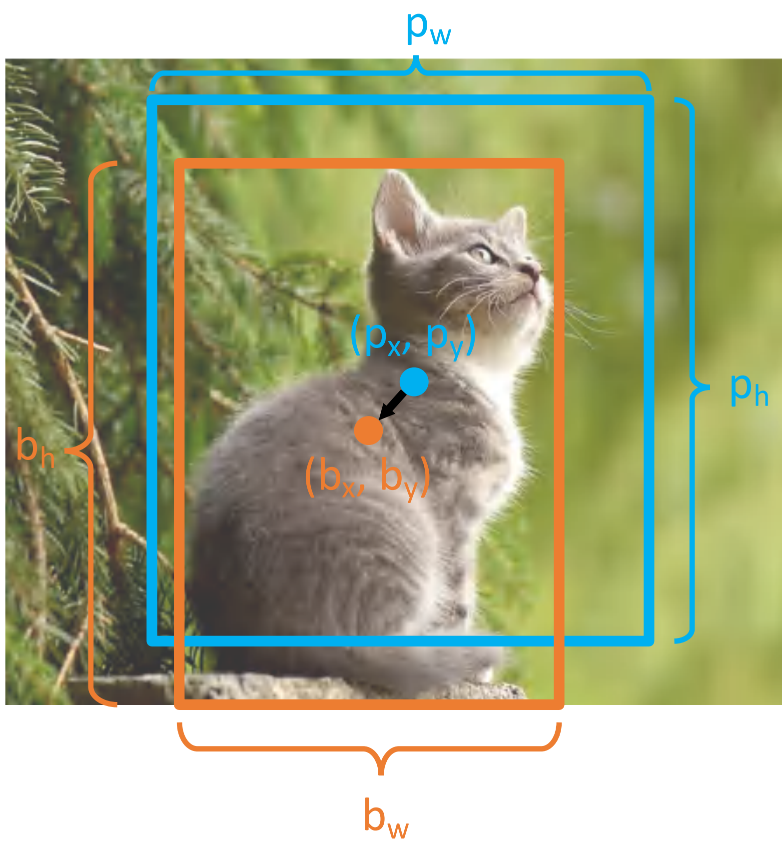
R-CNN: Box Regression



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

R-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:

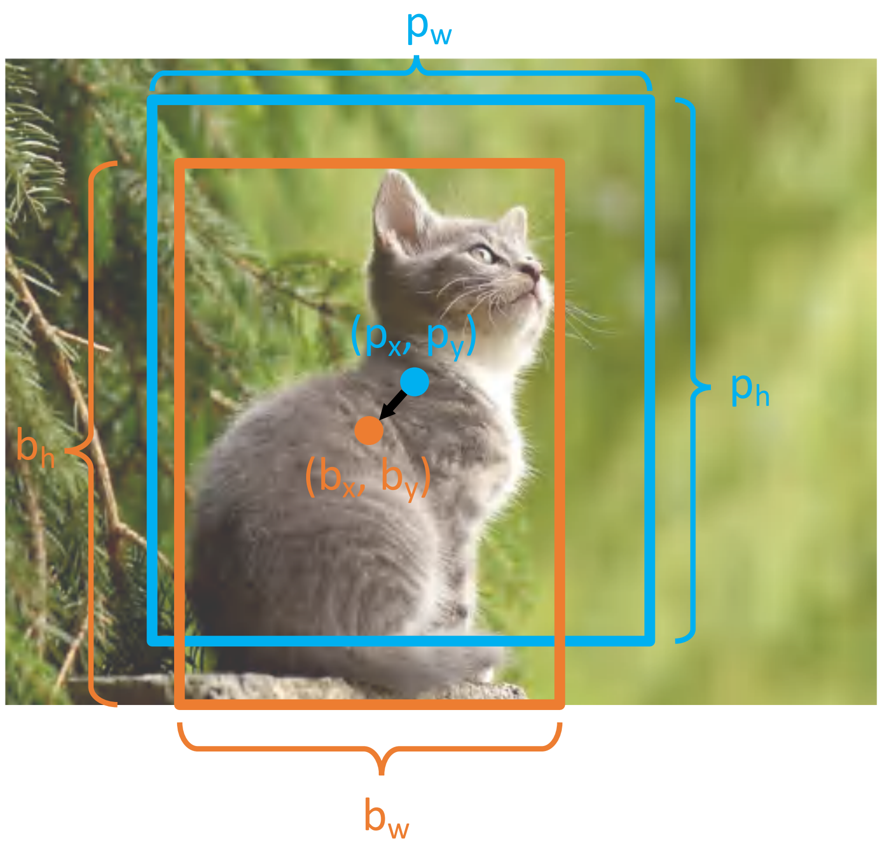
$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$

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

R-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

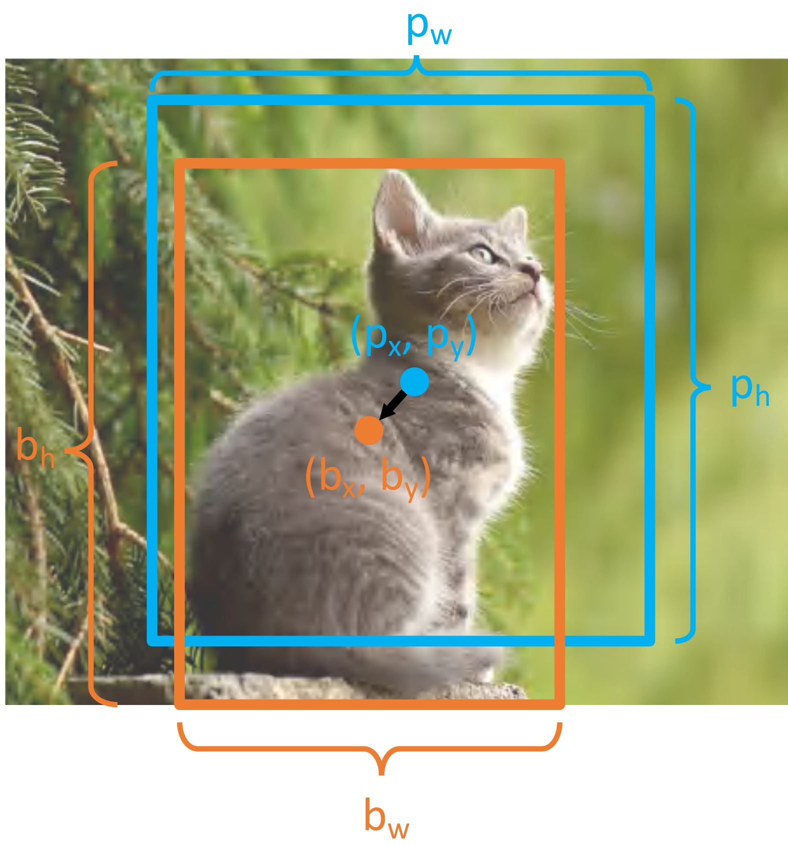
The output box is:

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

When transform is 0, output = proposal

L2 regularization encourages leaving proposal unchanged

R-CNN: Box Regression



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

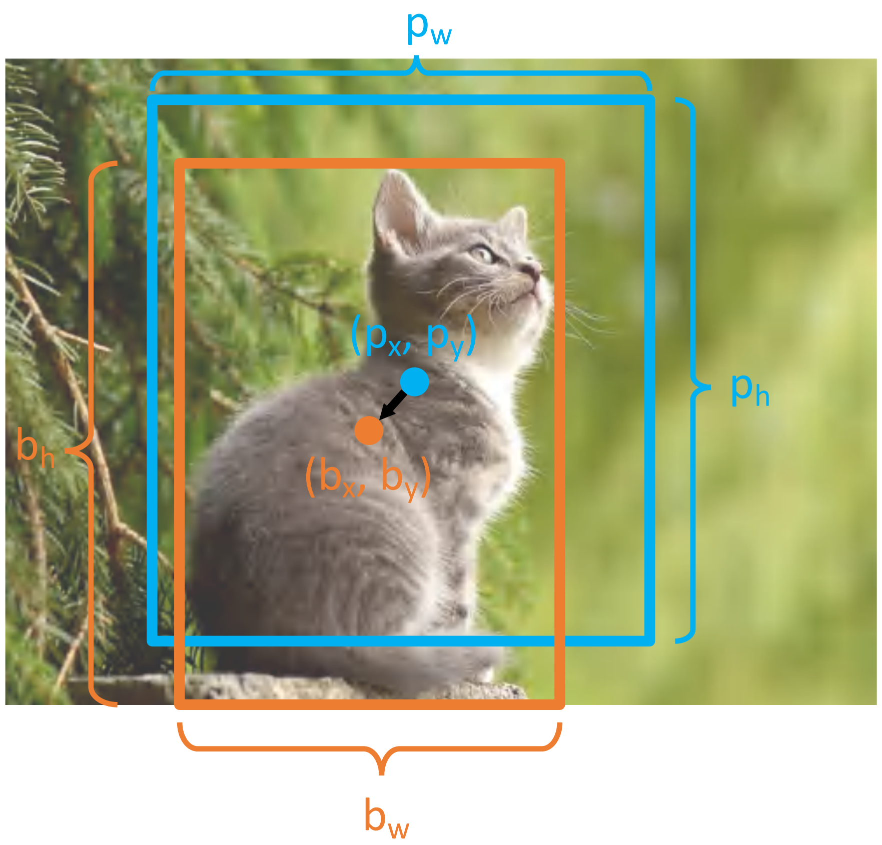
Model predicts a transform (t_x, t_y, t_w, t_h) to correct the region proposal

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

R-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:

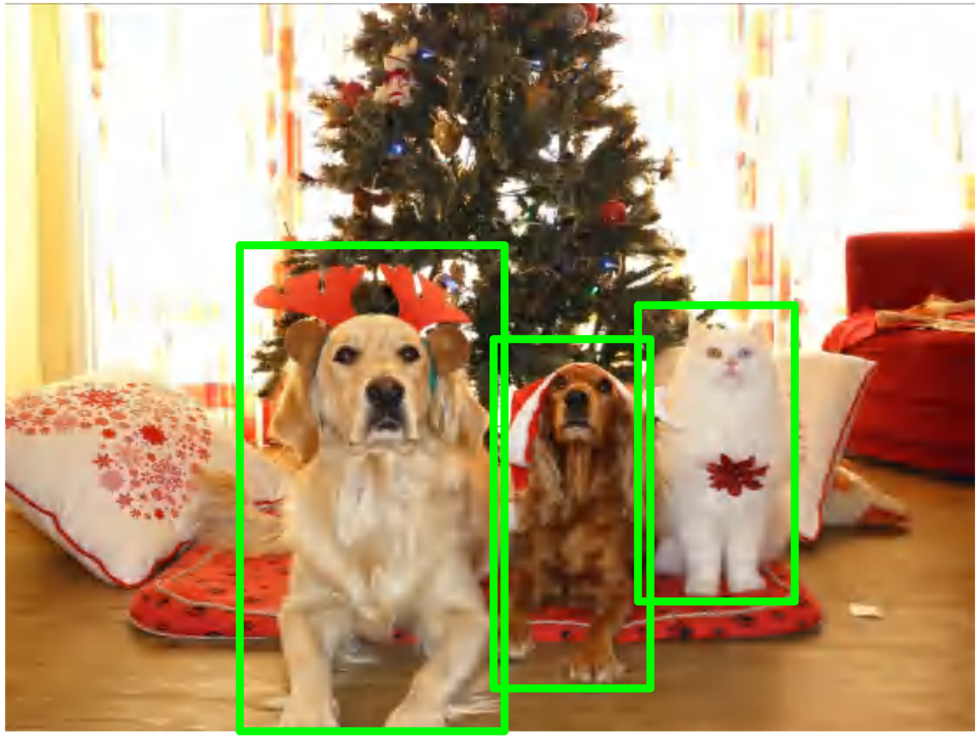
$$b_x = p_x + p_w t_x$$
$$b_y = p_y + p_h t_y$$
$$b_w = p_w \exp(t_w)$$
$$b_h = p_h \exp(t_h)$$

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

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

R-CNN Training

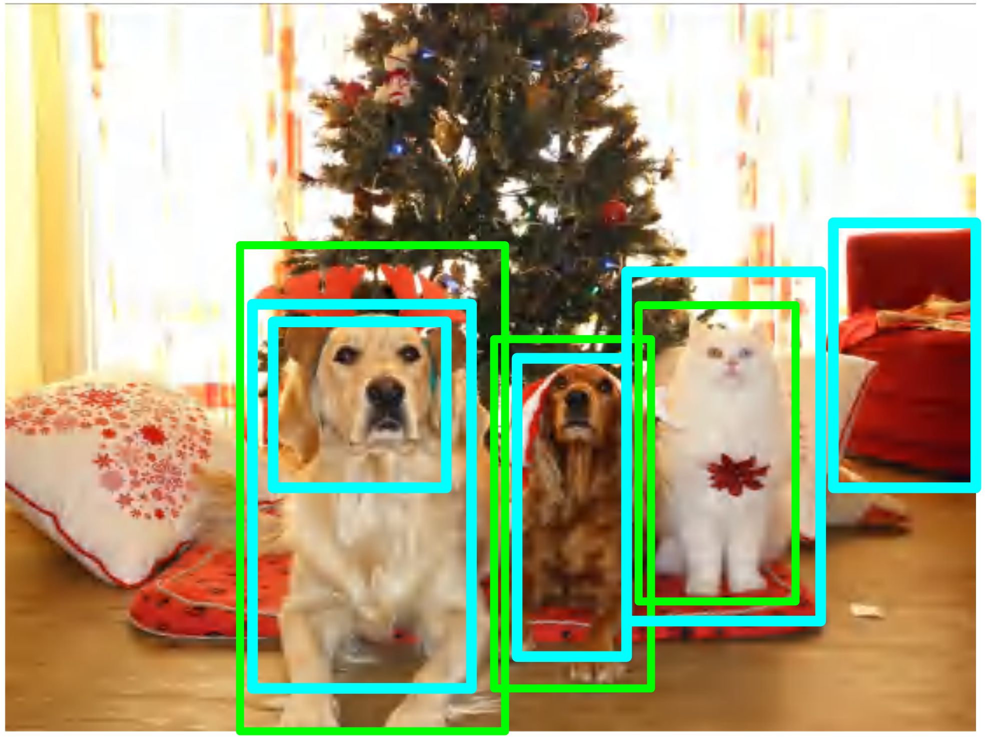
Input Image



Ground-Truth boxes

R-CNN Training

Input Image

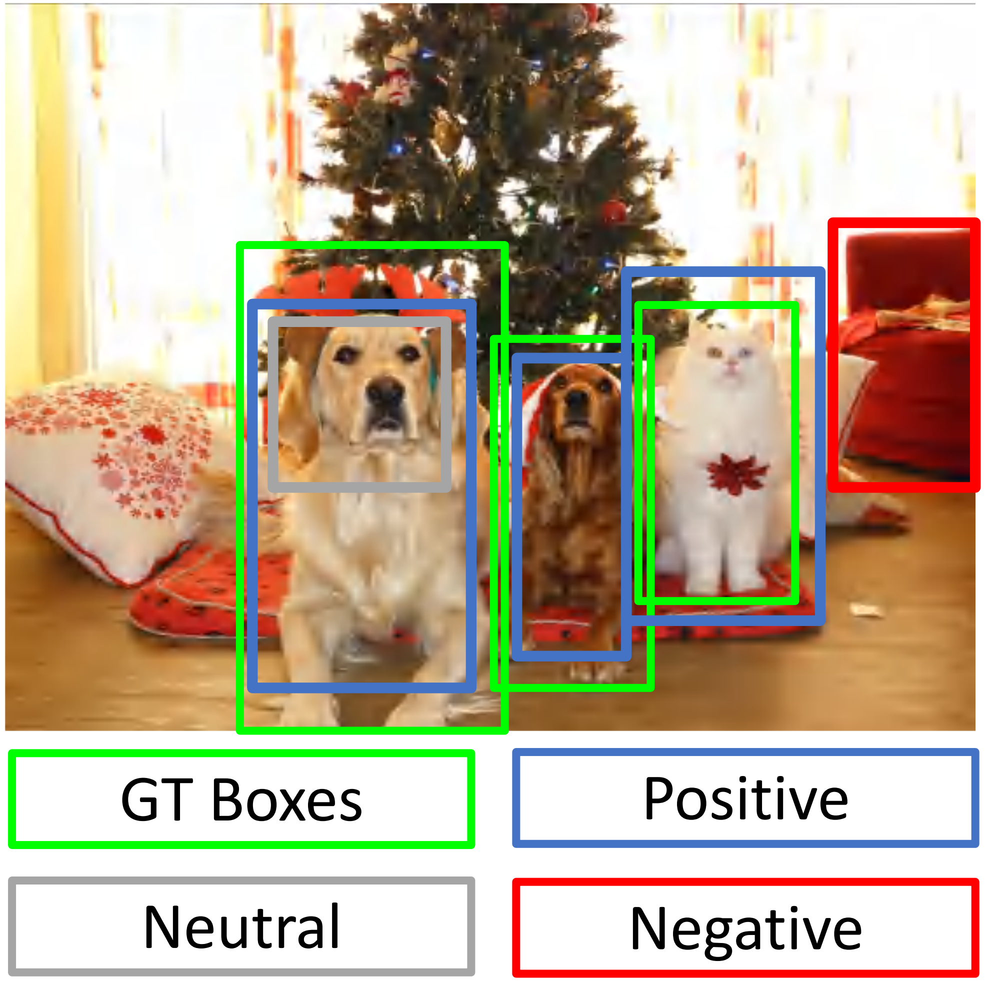


Ground-Truth boxes

Region Proposals

R-CNN Training

Input Image

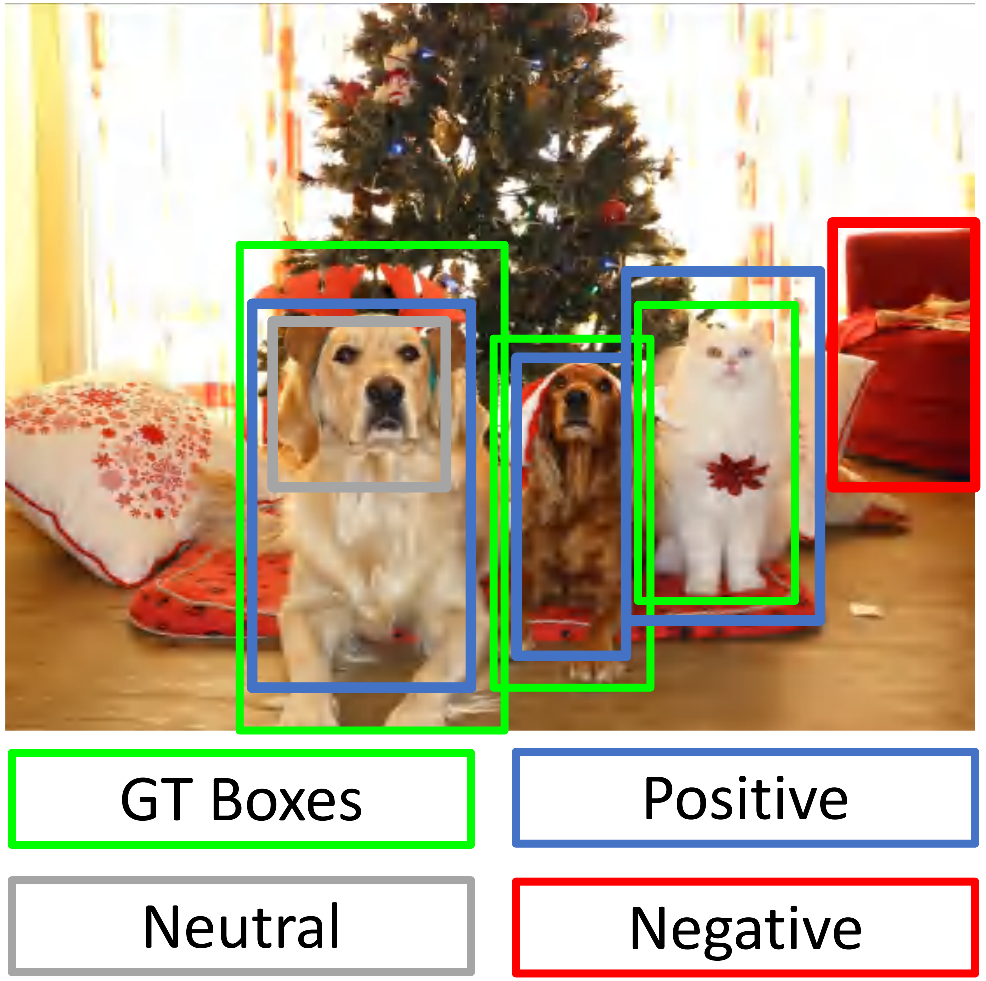


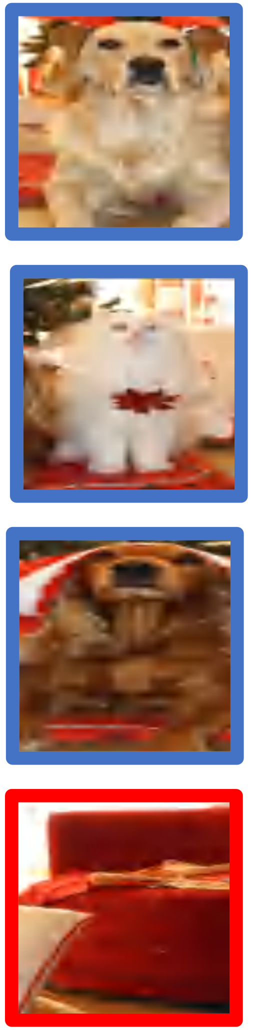
Categorize each region proposal as positive, negative, or neutral based on overlap with ground-truth boxes:

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

R-CNN Training

Input Image

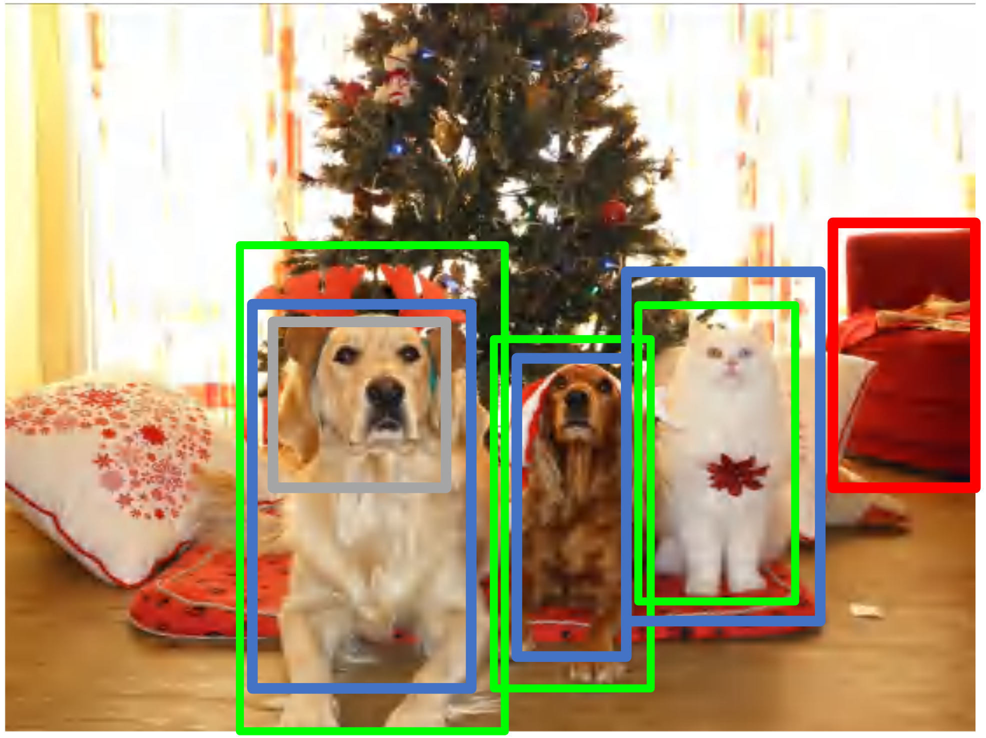




Crop pixels from each positive and negative proposal, resize to 224 x 224

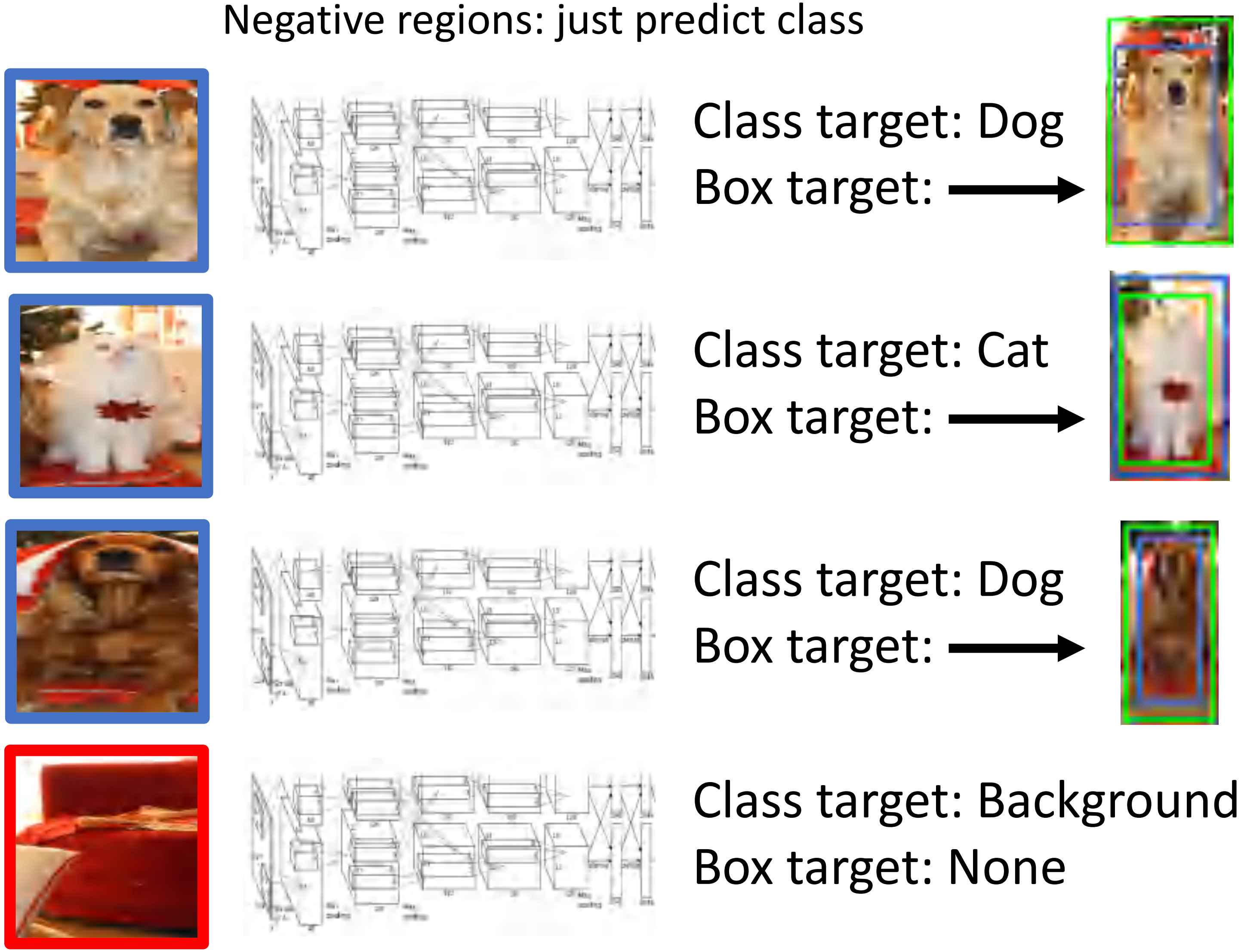
R-CNN Training

Input Image



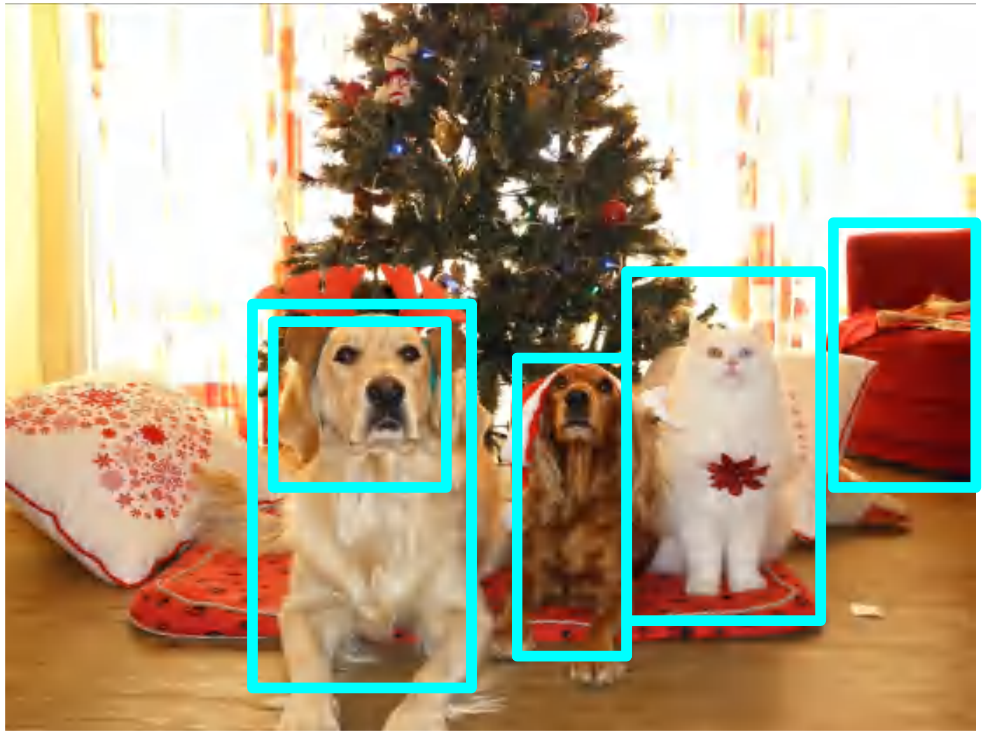
GT Boxes | Positive | Neutral | Negative

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



R-CNN Test-Time

Input Image



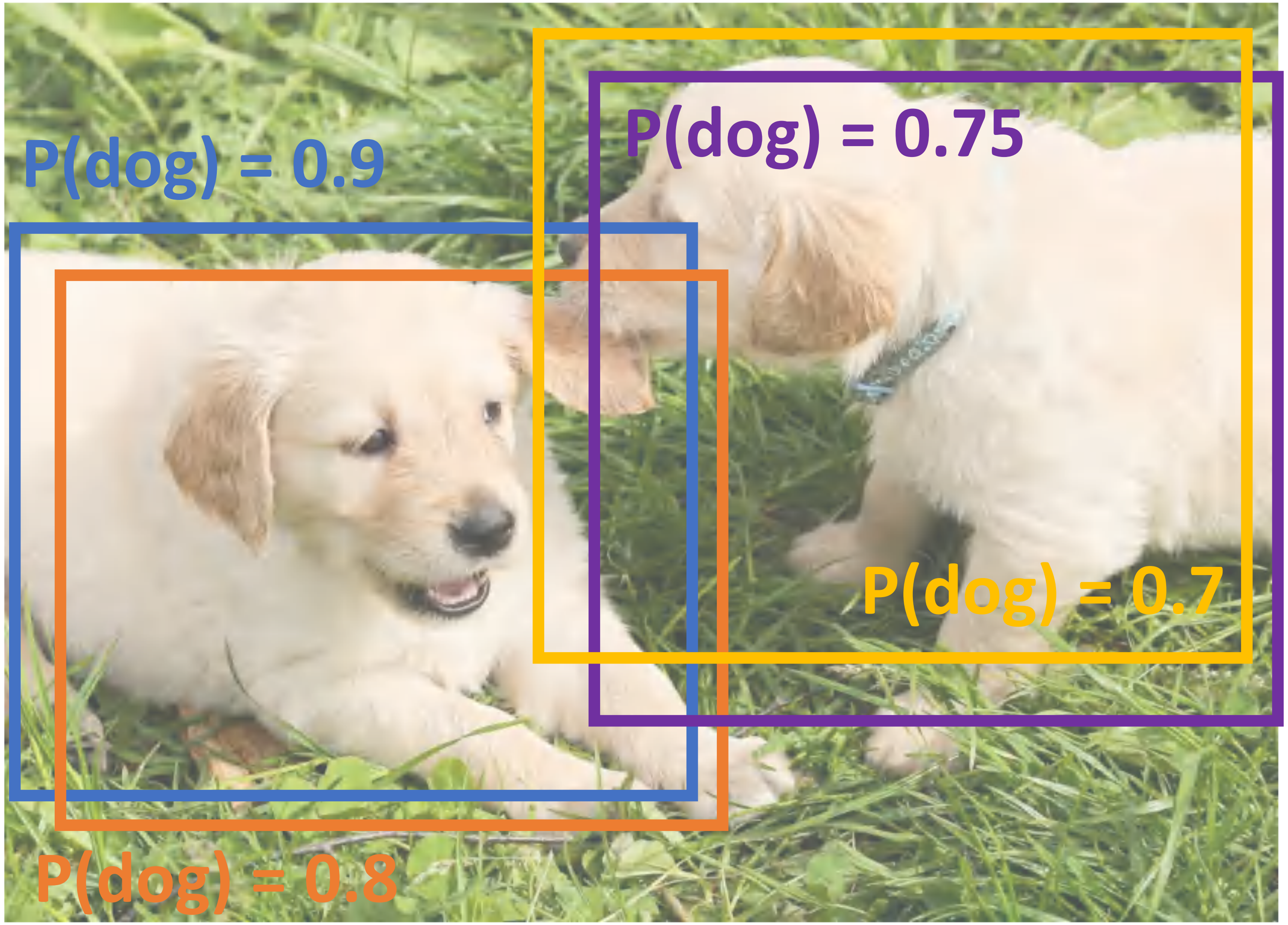
Region Proposals

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

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

Overlapping Boxes

Problem: Object detectors often output many overlapping detections:



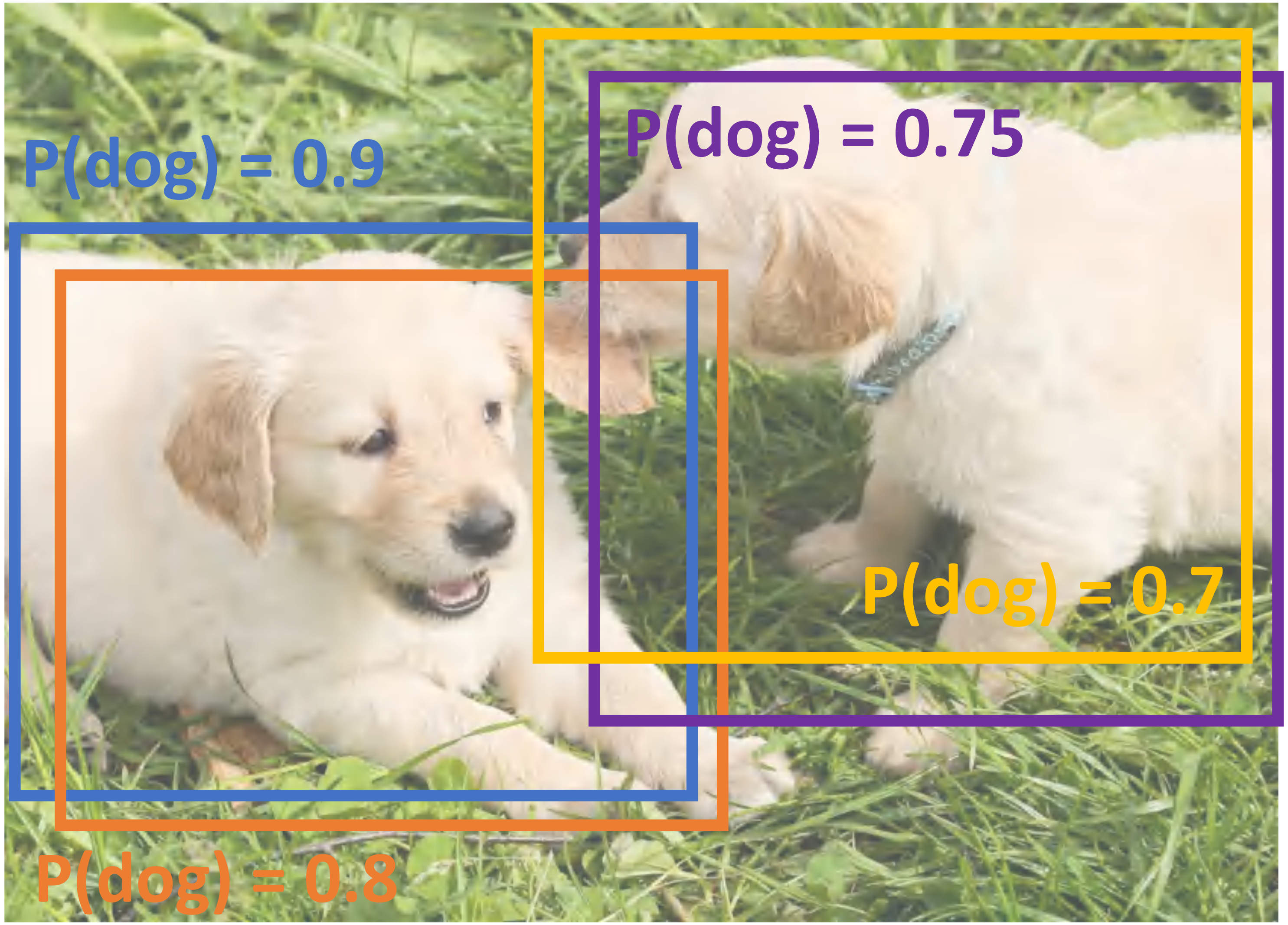
Puppy image is CC0 Public Domain

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



Puppy image is CC0 Public Domain

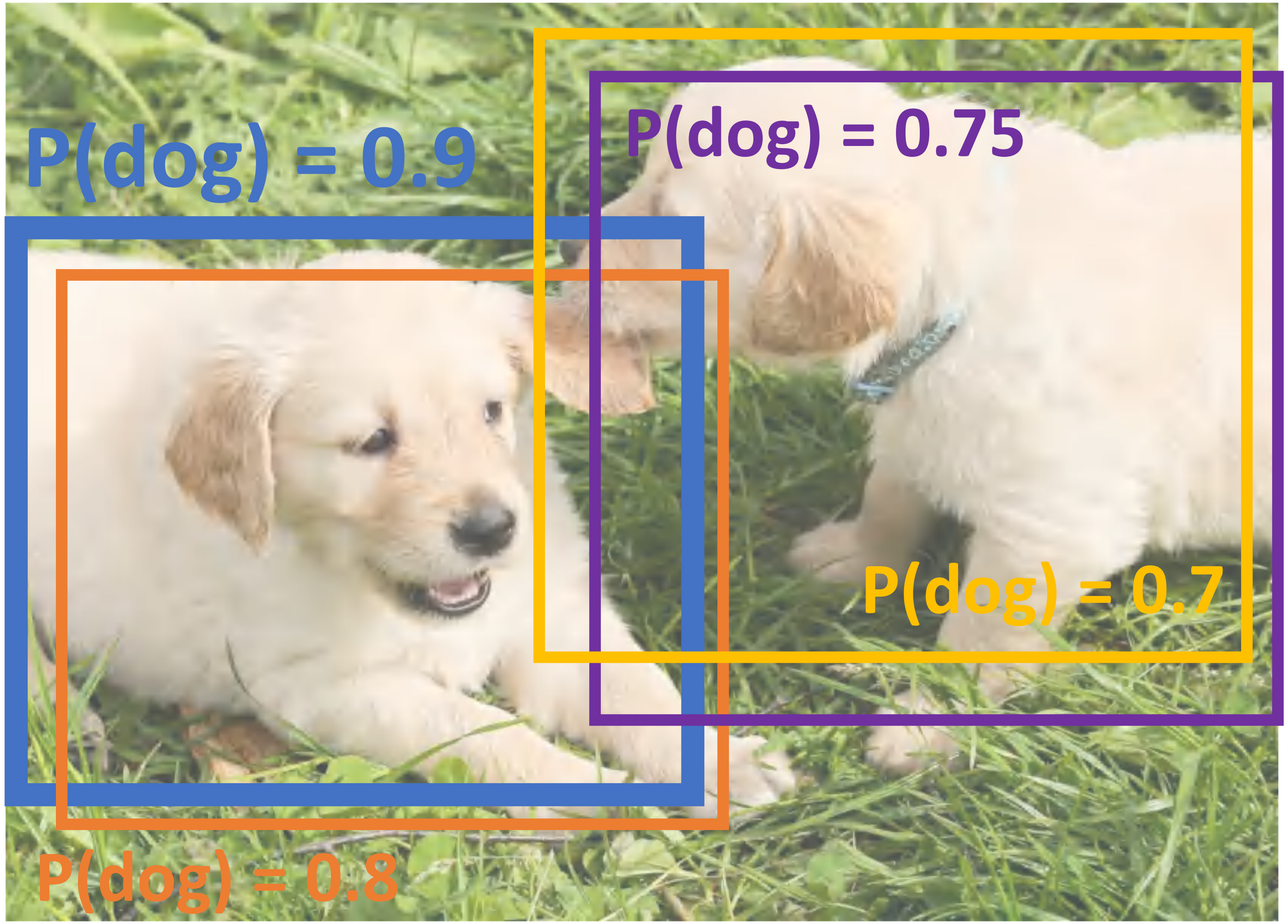
Overlapping Boxes: **Non-Max Suppression (NMS)**

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



Puppy image is CC0 Public Domain

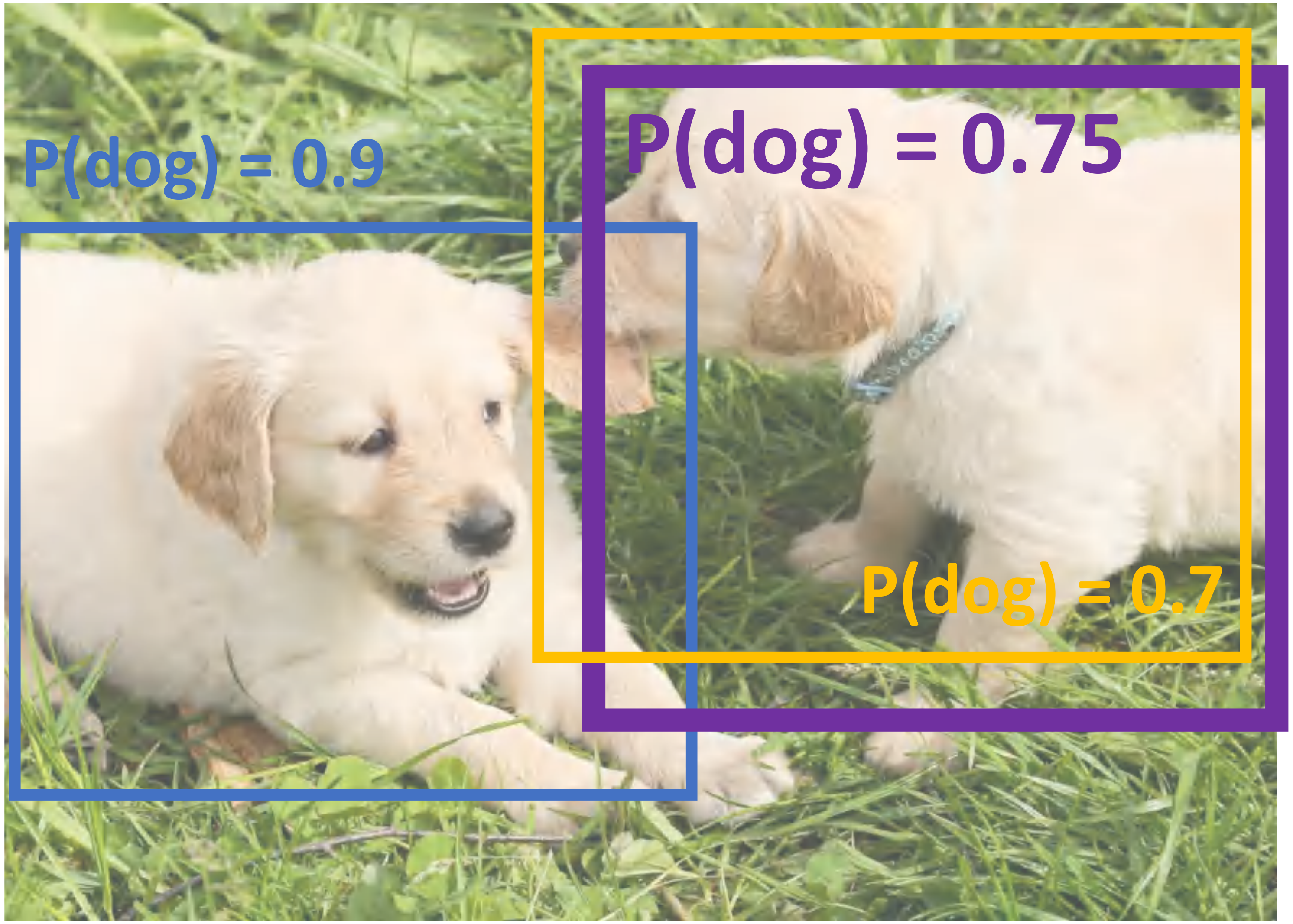
Overlapping Boxes: Non-Max Suppression (NMS)

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**



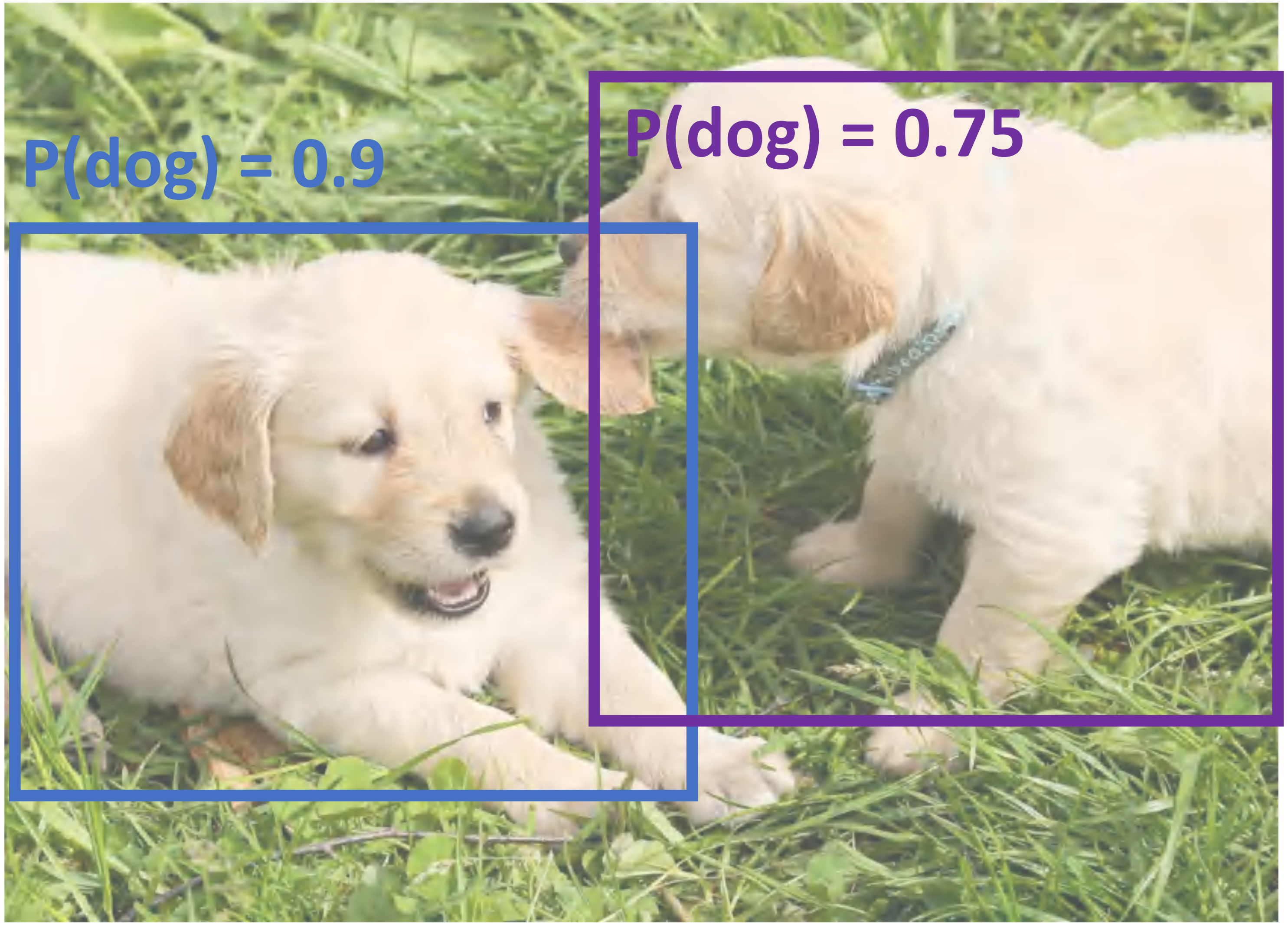
Puppy image is CC0 Public Domain

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



Puppy image is CC0 Public Domain

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

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

Crowd image is free for commercial use under the Pixabay license

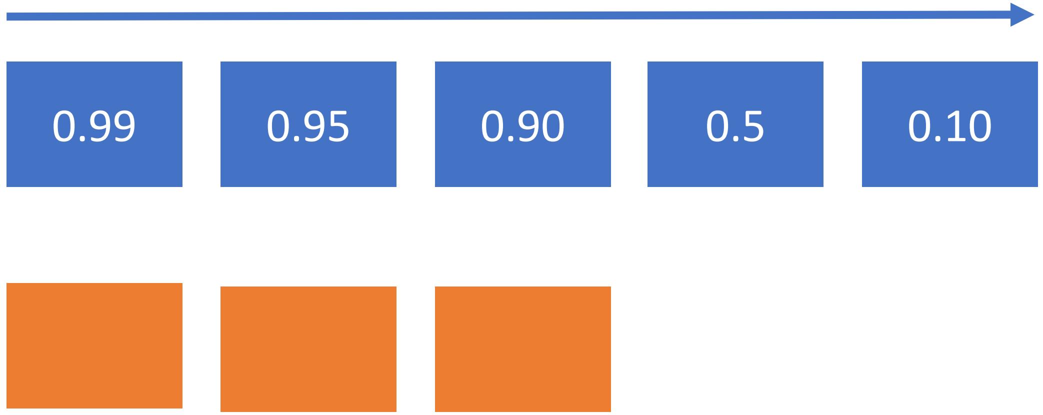
Evaluating Object Detectors: Mean Average Precision (mAP)

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

Evaluating Object Detectors: Mean Average Precision (mAP)

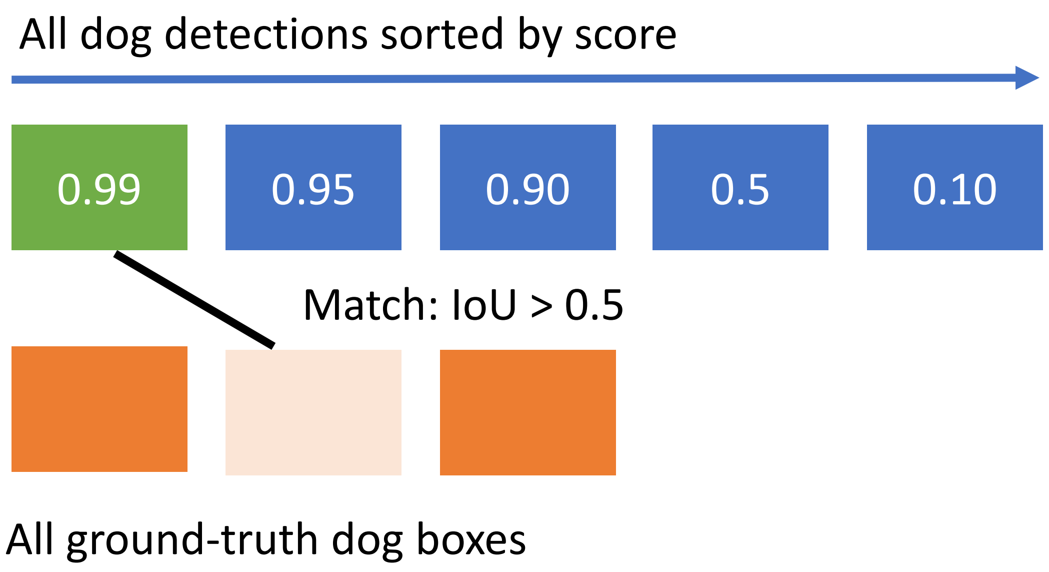
1. Run object detector on all test images (with NMS)
2. For each category, compute Average Precision (AP) = area under Precision vs Recall Curve
   1. For each detection (highest score to lowest score)

All dog detections sorted by score



All ground-truth dog boxes

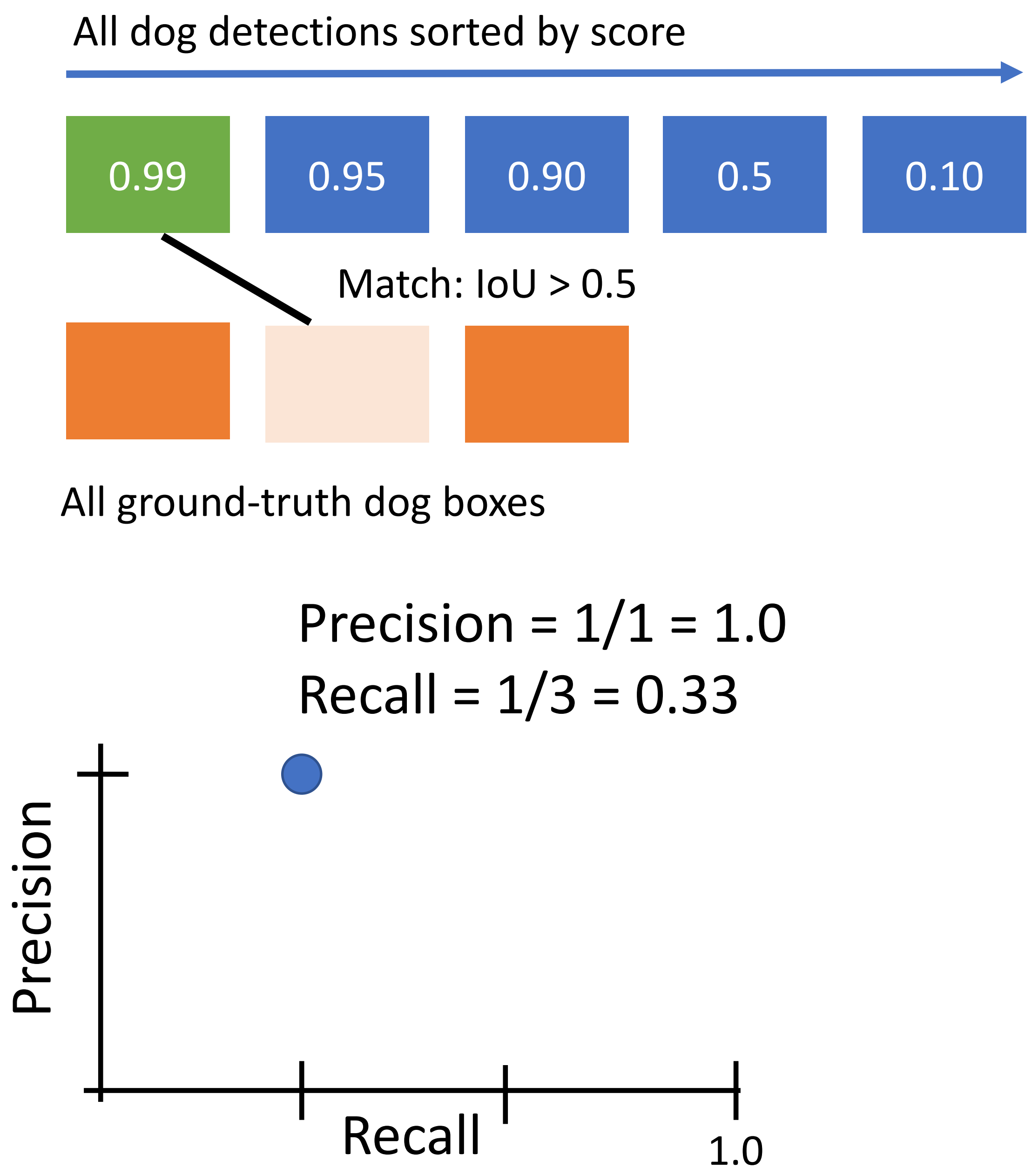
Evaluating Object Detectors: Mean Average Precision (mAP)



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

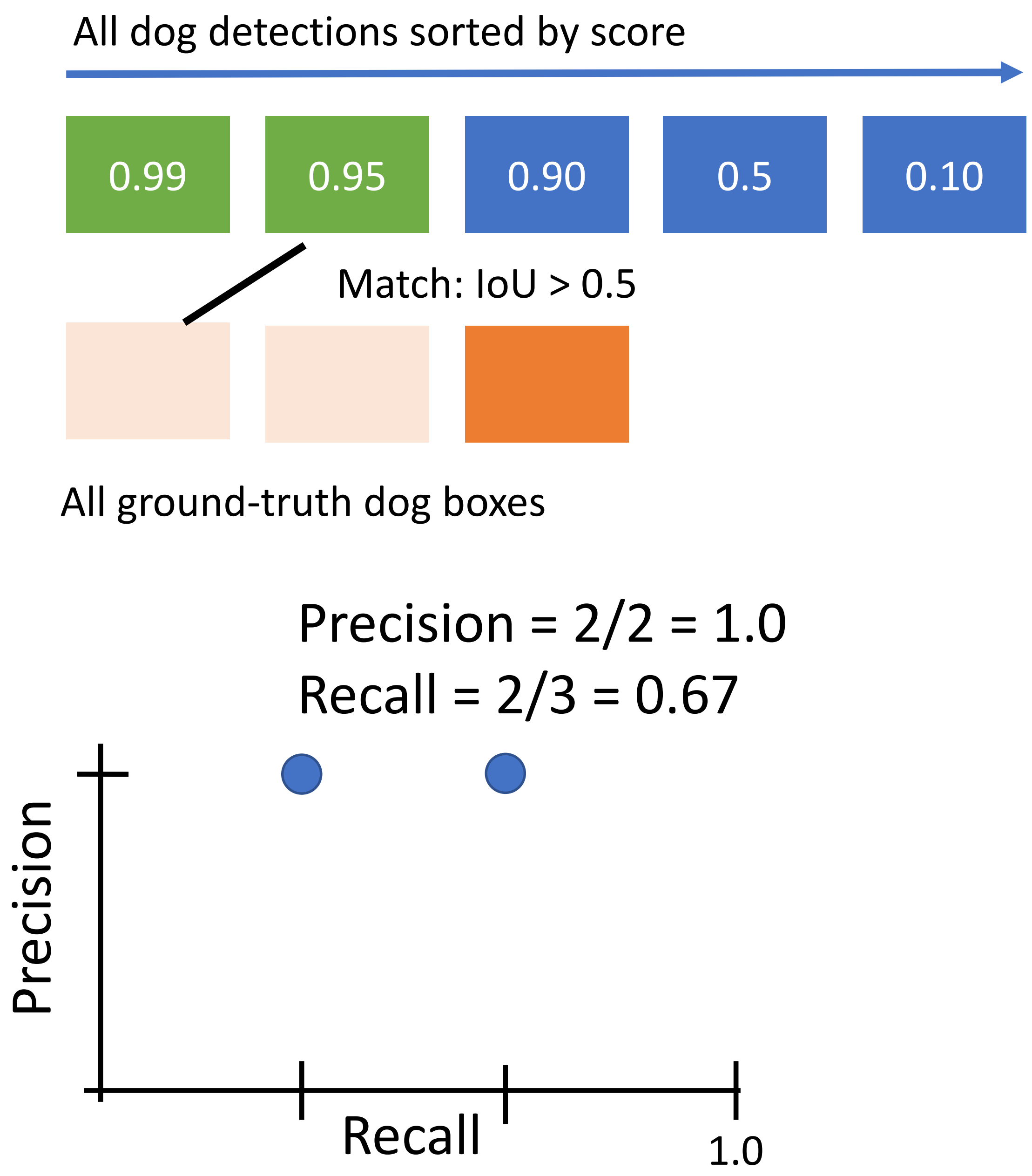
Evaluating Object Detectors: Mean Average Precision (mAP)

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



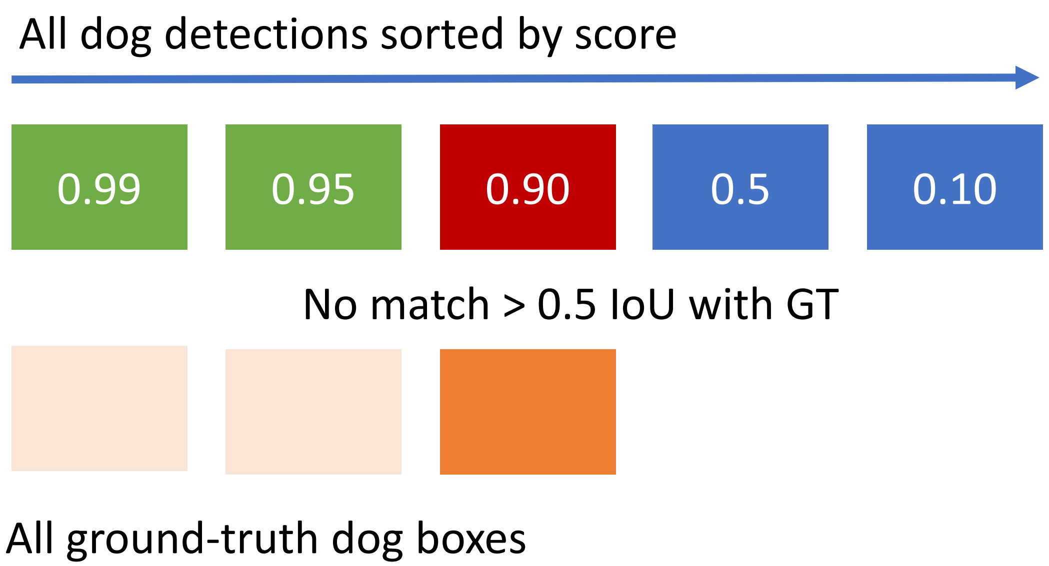
Evaluating Object Detectors: Mean Average Precision (mAP)

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

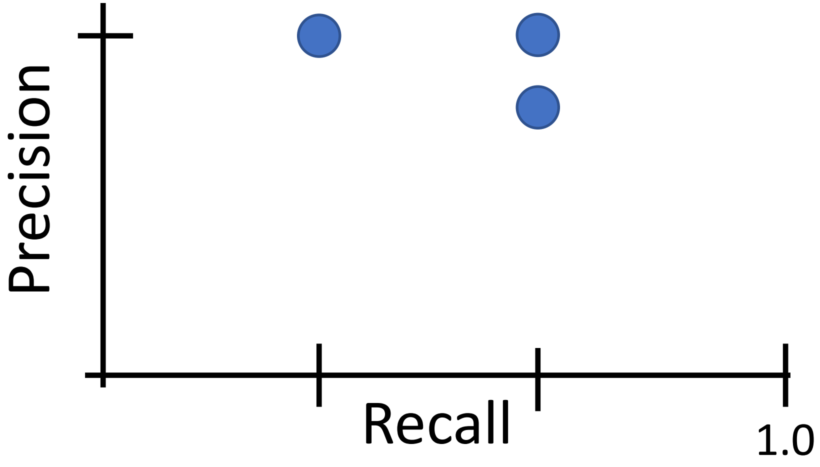


Evaluating Object Detectors: Mean Average Precision (mAP)

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

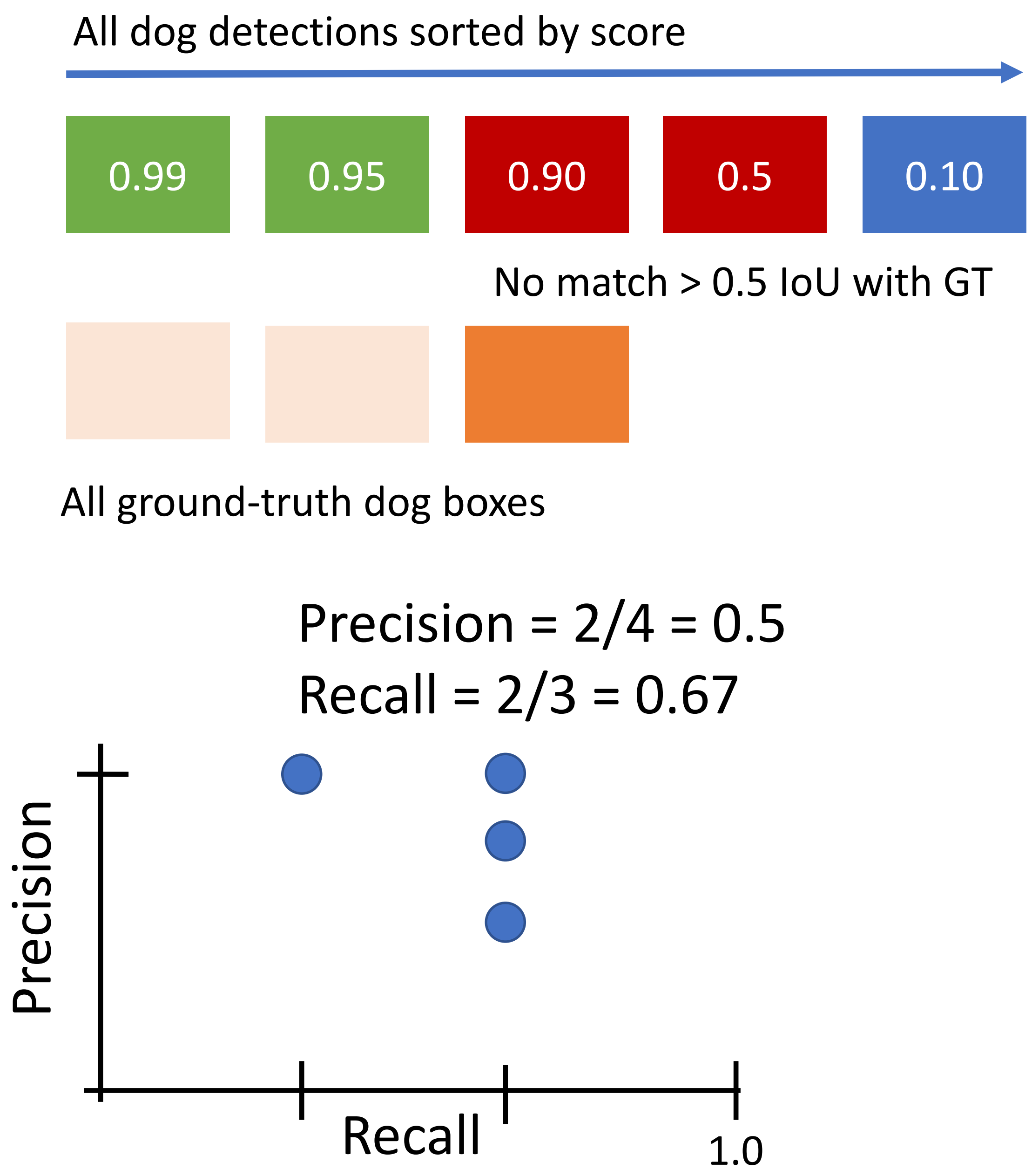


Precision = 2/3 = 0.67
Recall = 2/3 = 0.67



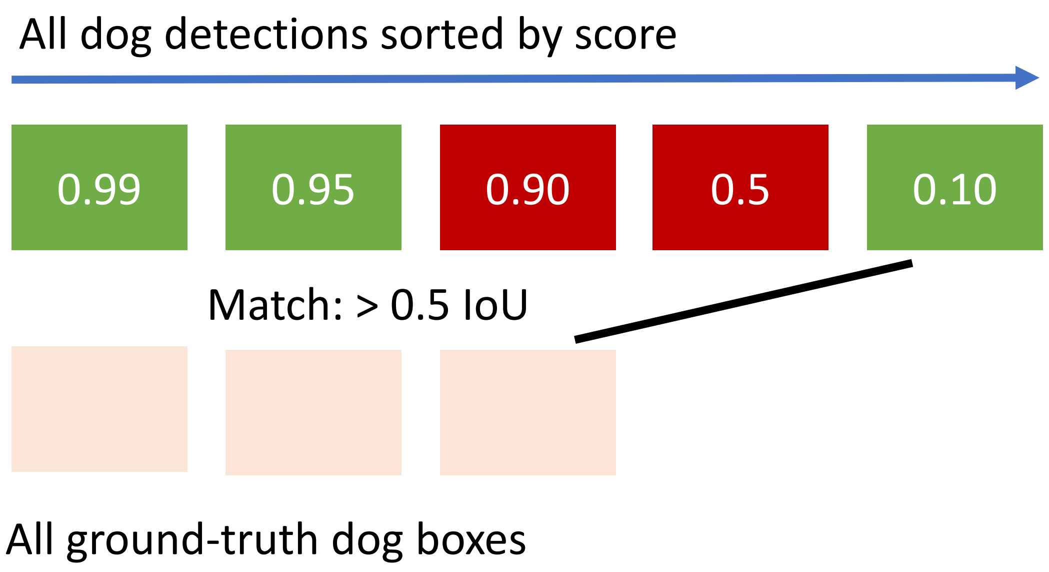
Evaluating Object Detectors: Mean Average Precision (mAP)

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



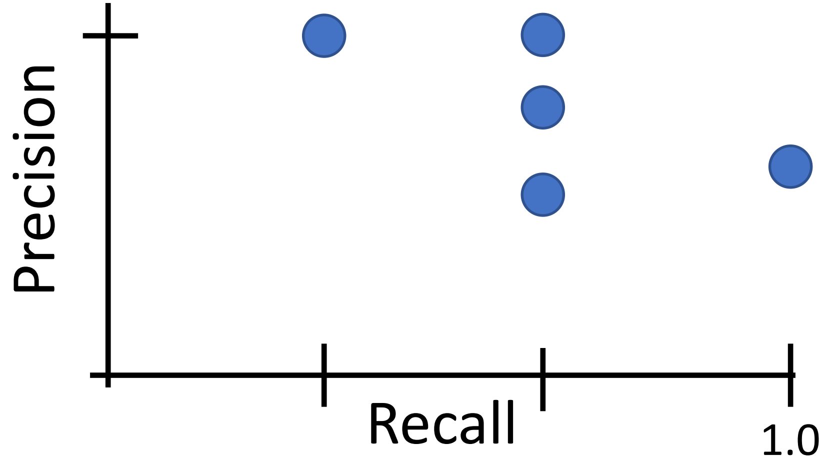
Evaluating Object Detectors: Mean Average Precision (mAP)

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



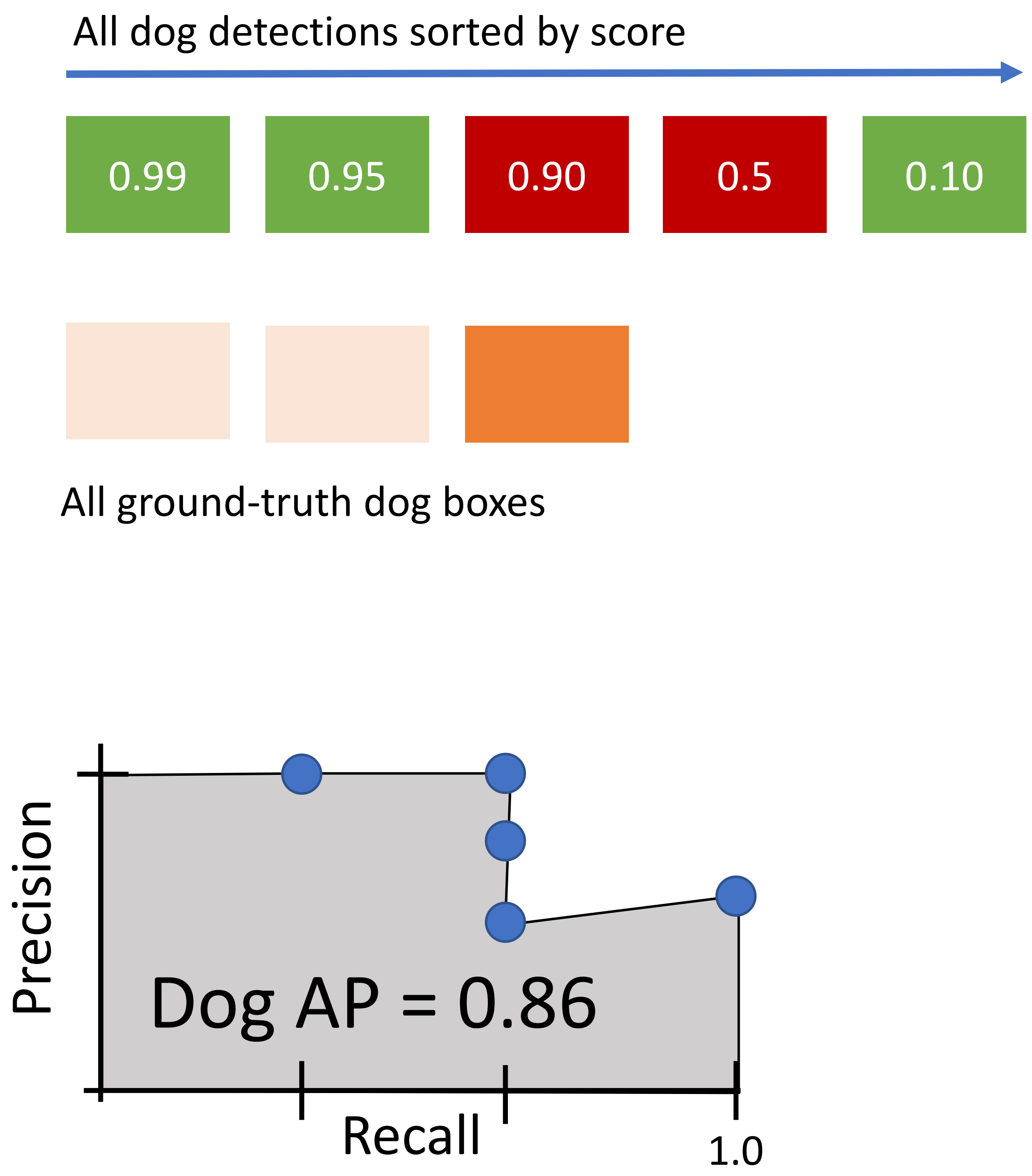
Precision = 3/5 = 0.6

Recall = 3/3 = 1.0



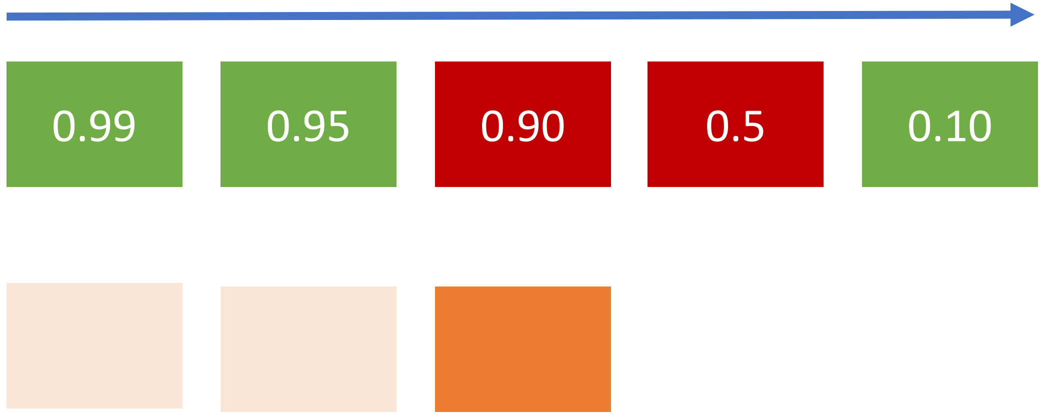
Evaluating Object Detectors: Mean Average Precision (mAP)

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



Evaluating Object Detectors: Mean Average Precision (mAP)

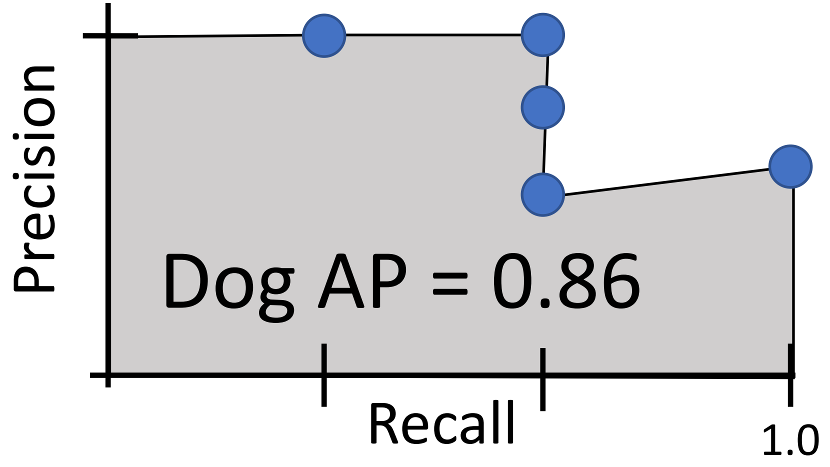
All dog detections sorted by score



All ground-truth dog boxes

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

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"



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

Car AP = 0.65

Cat AP = 0.80

Dog AP = 0.86

mAP@0.5 = 0.77

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

mAP@0.5 = 0.77

mAP@0.55 = 0.71

mAP@0.60 = 0.65

…

mAP@0.95 = 0.2

COCO mAP = 0.4

Summary: Beyond Image Classification

Classification

CAT

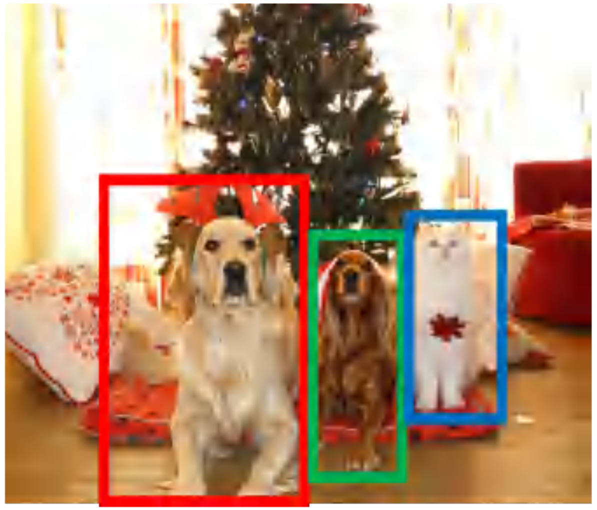
No spatial extent

Semantic Segmentation

GRASS, CAT, TREE, SKY

No objects, just pixels

Object Detection



DOG, DOG, CAT

Instance Segmentation

DOG, DOG, CAT

Multiple Objects

This image is CC0 public domain

Summary

Transfer learning allows us to re-use a trained network for new tasks

Object detection is the task of localizing objects with bounding boxes

Intersection over Union (IoU) quantifies differences between bounding boxes

The **R-CNN** object detector processes **region proposals** with a CNN

At test-time, eliminate overlapping detections using **non-max suppression (NMS)**

Evaluate object detectors using **mean average precision (mAP)**

Next time:
Modern Object Detectors