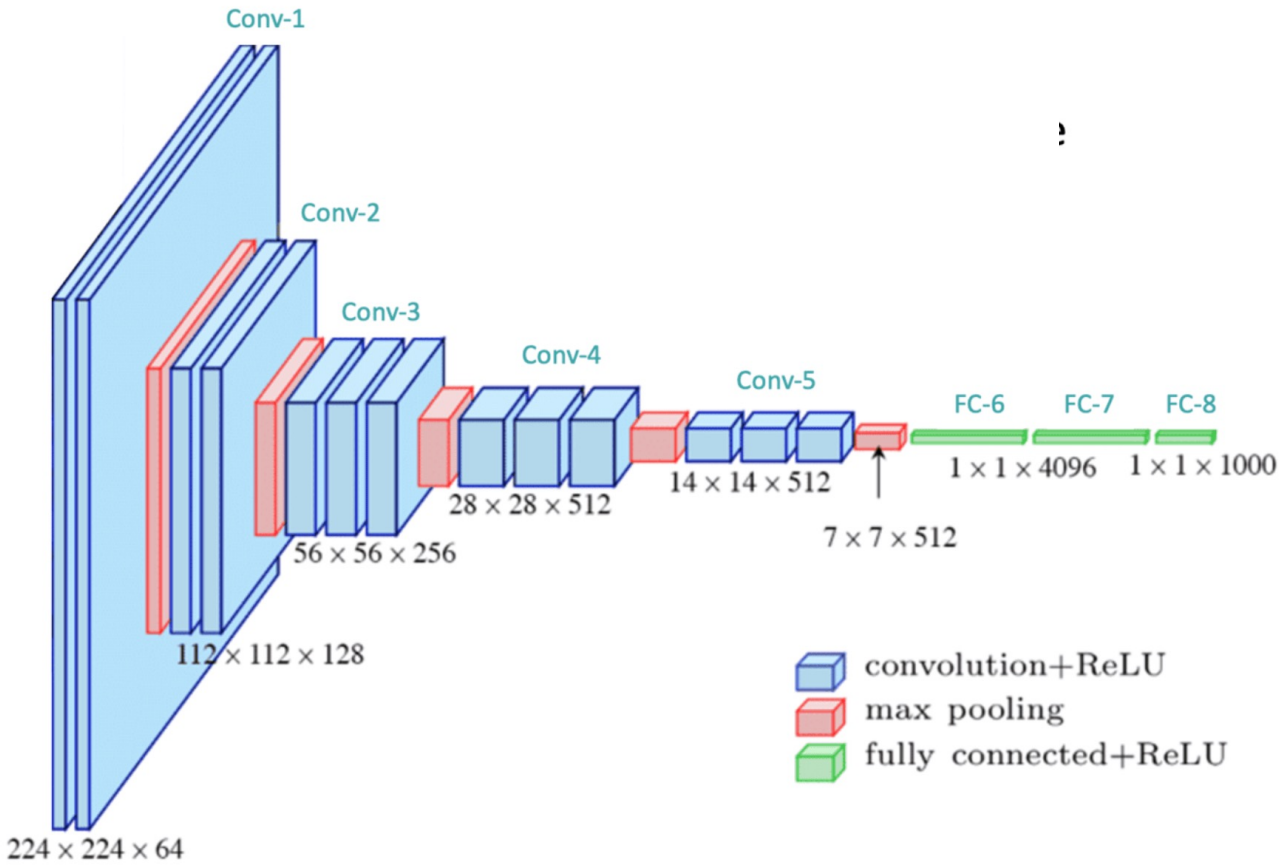
Lecture 4.1
Object Detection

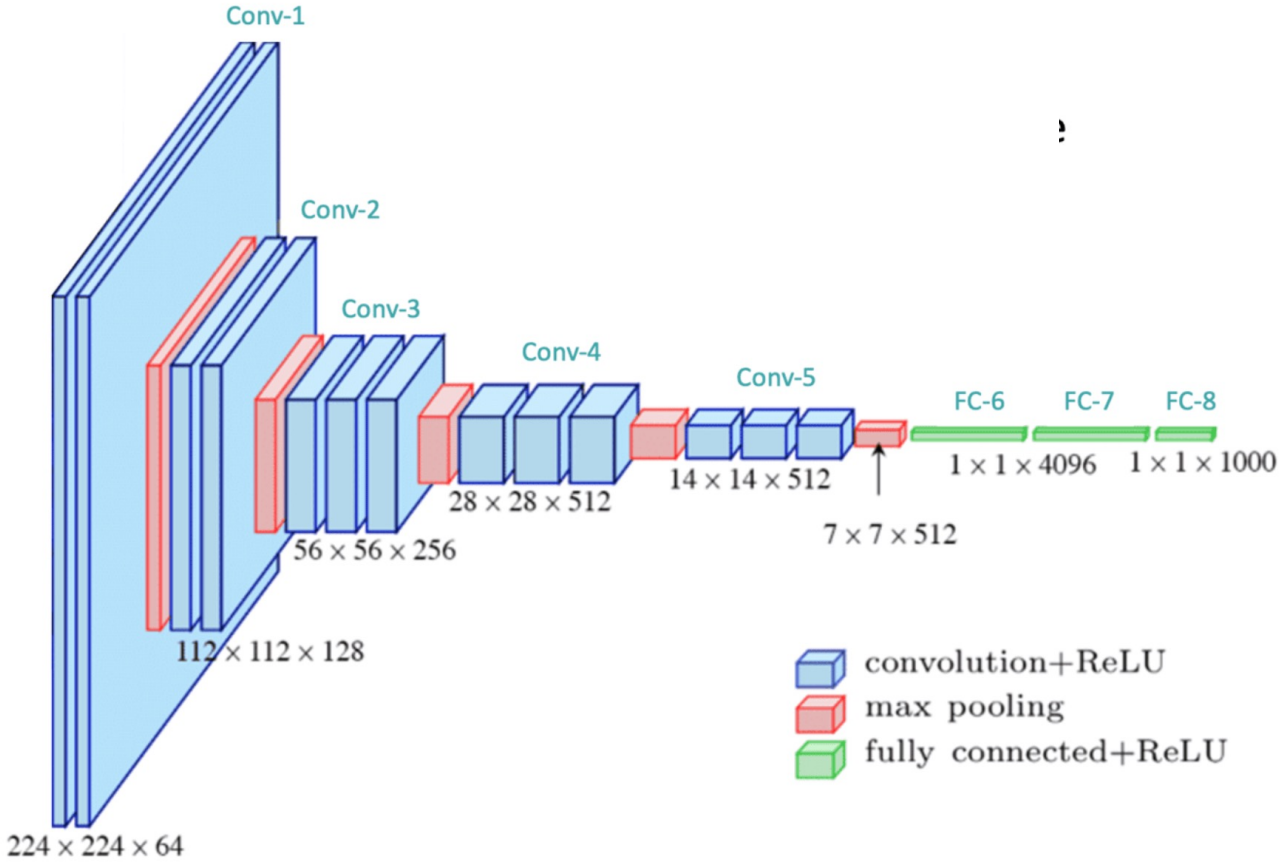
Dimitrios Papadopoulos
Associate Professor, DTU Compute

So far, you have learned…

So far, you have learned…

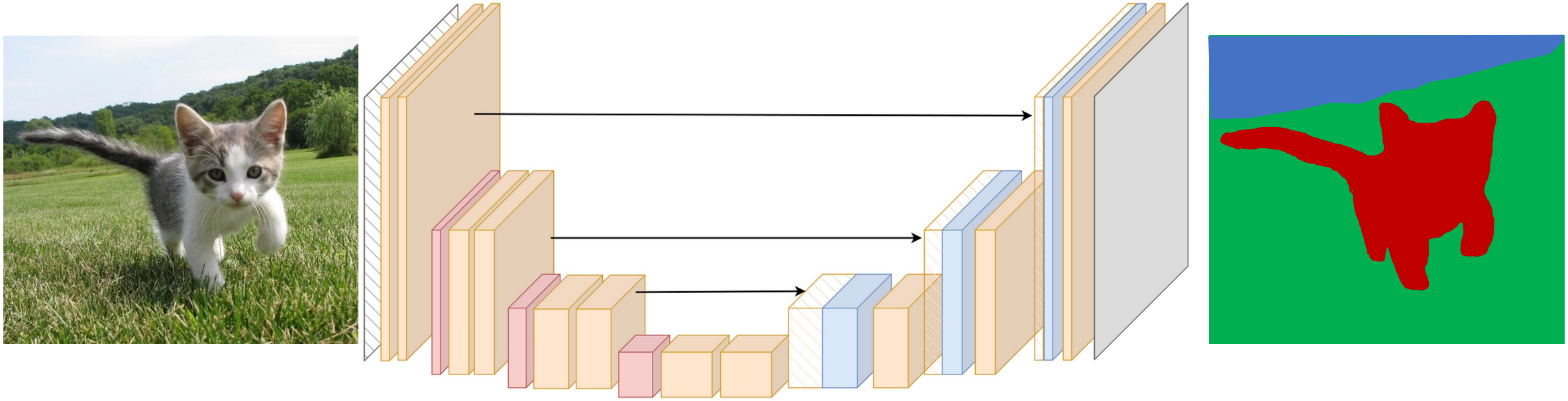


So far, you have learned…

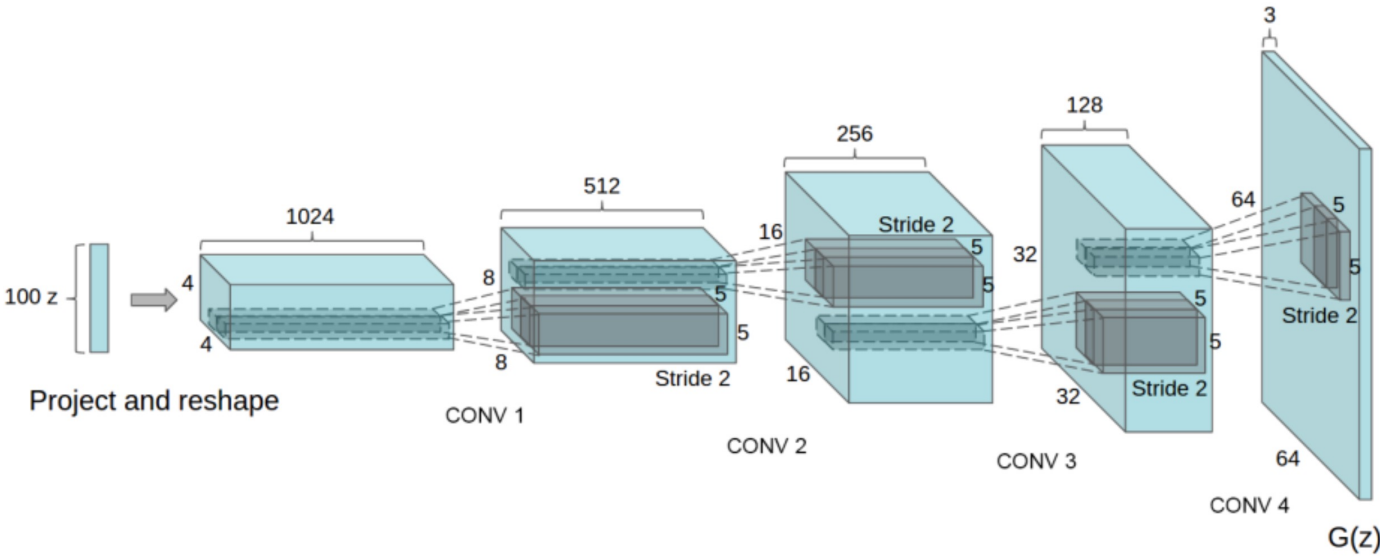


cat

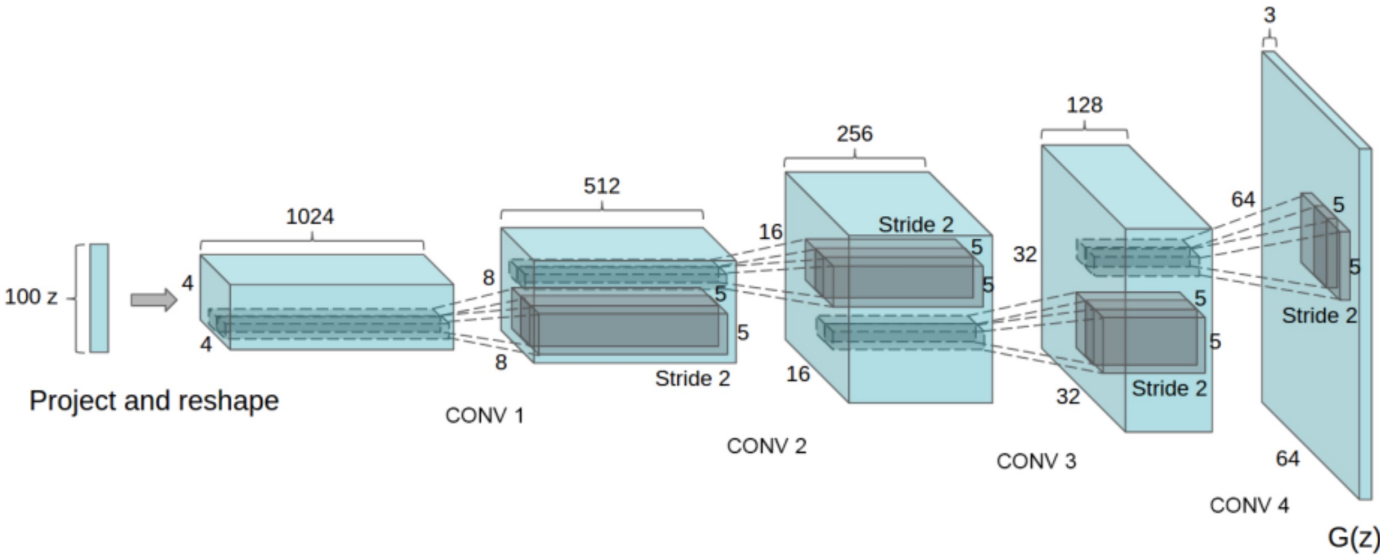
So far, you have learned…



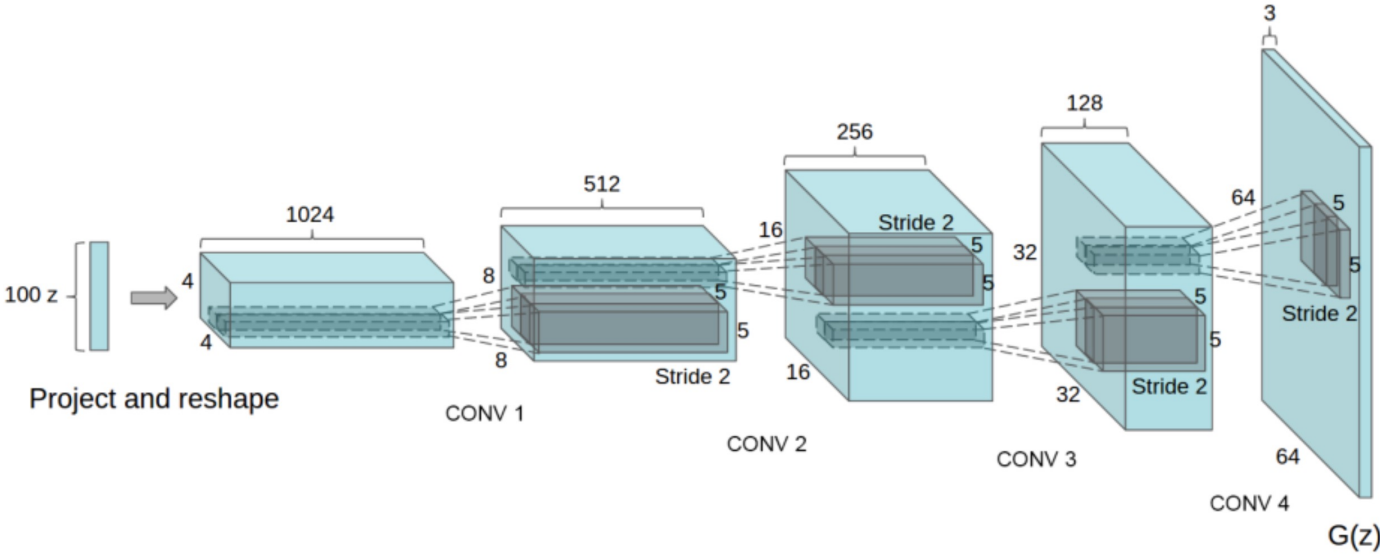
So far, you have learned…



So far, you have learned…



So far, you have learned…



“a cute cat in Copenhagen”

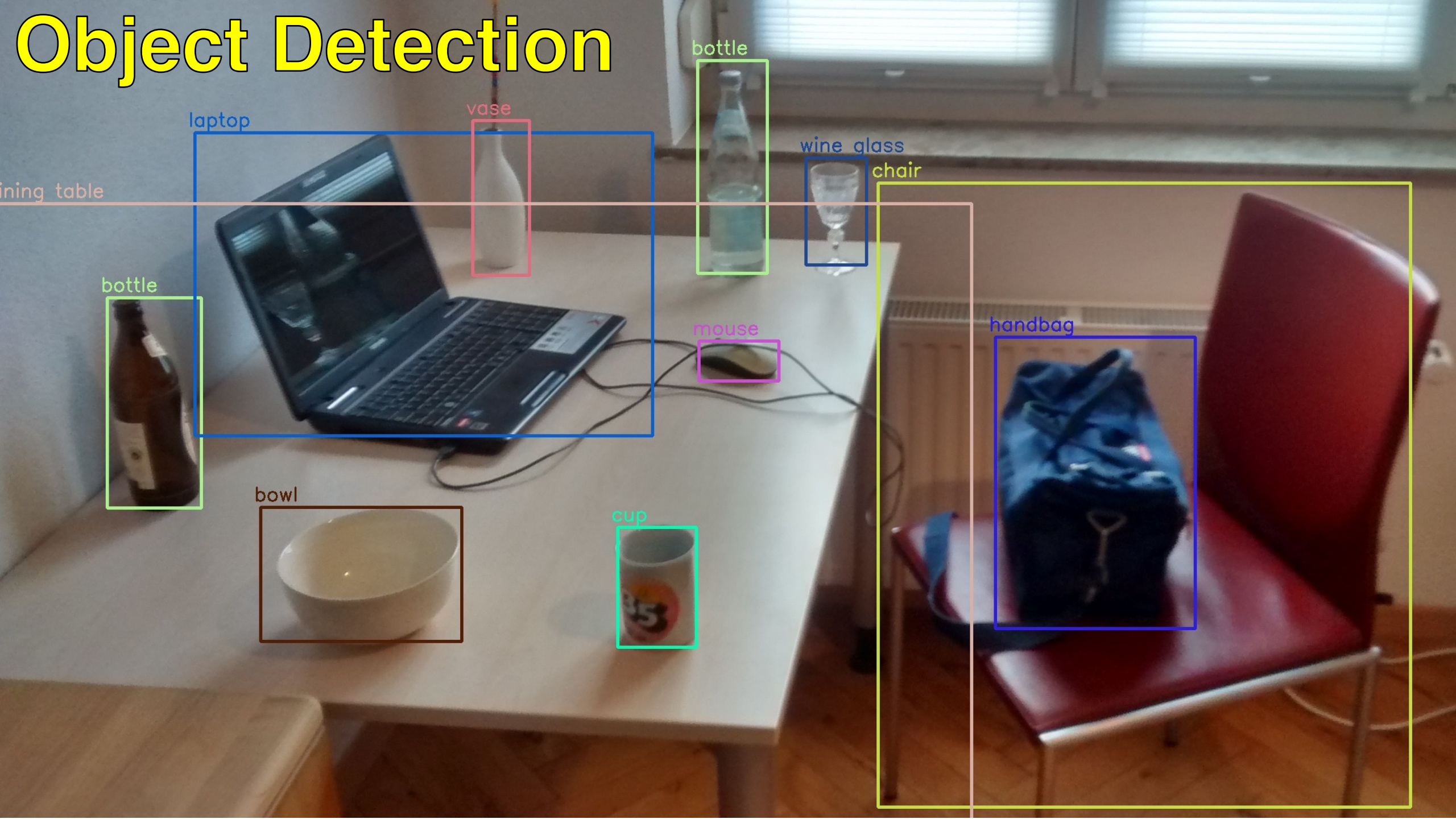
Object Detection

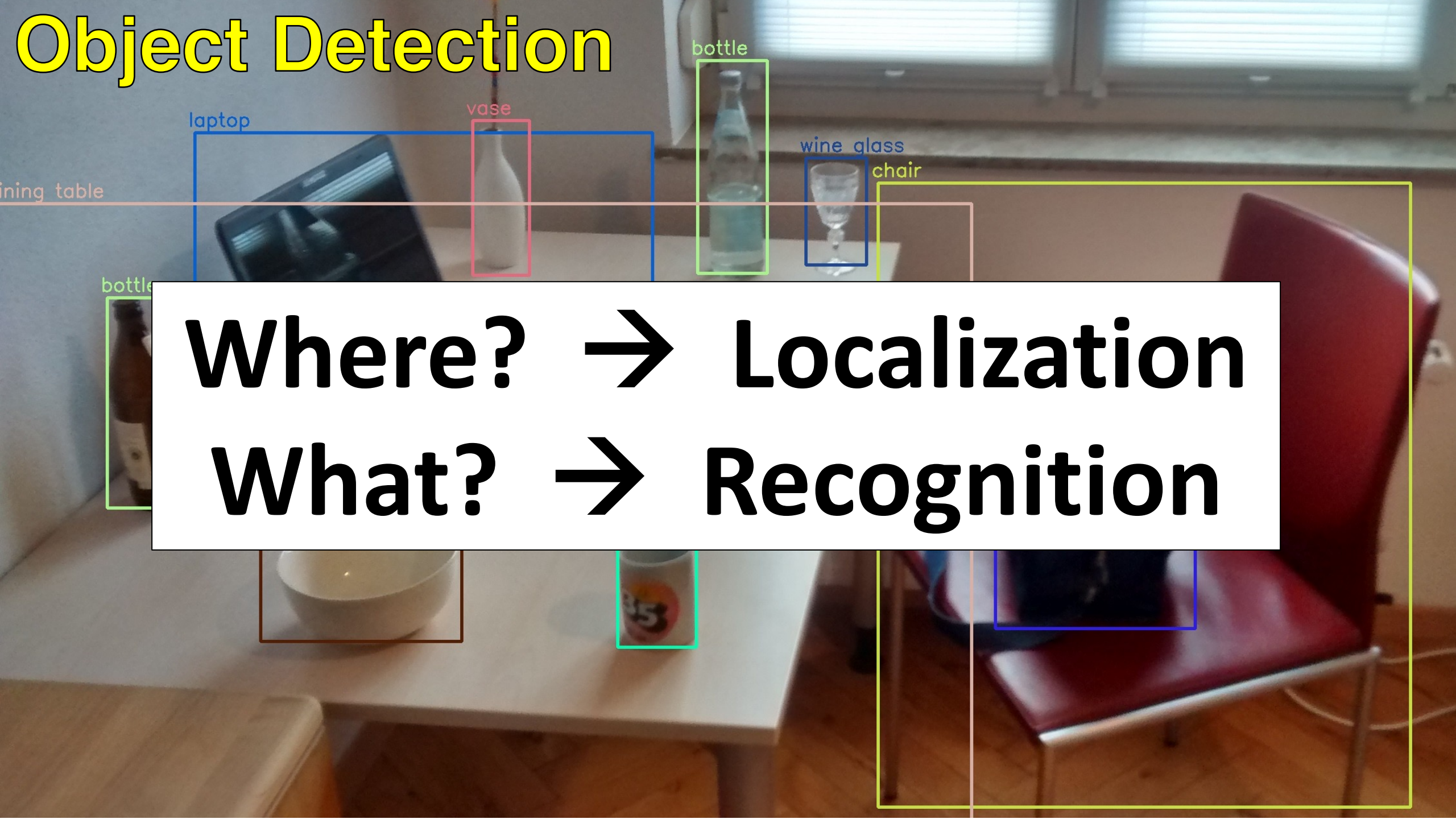
Part 4: Object Detection

Friday 16.6	09:00-11:00 Lecture 1.4 Object Detection Introduction to Project 4 Detection 11:00-17:00 Work on Project 4	Dimitrios 11-12 Dimitrios 13-15 Manxi 14-16 Paraskevas 15-17 Thanos
Monday 19.6	9-10 Lecture - State of the art object recognition 10-11 Poster session on Project 3 11-17 Work on Project 4	Dimitrios 10-12 Dimitrios 13-17 Thanos 14-16 Paraskevas
Tuesday 20.6	9.00-10.00 Lecture - TBA: Scene understanding: Semantic, instance and panoptic segmentation 10-17 Work on Project 4 **Project 4 deadline at midnight**	Dimitrios 10-11 Dimitrios, Thanos 13-17 Thanos 14-16 Paraskevas
Wednesday 21.6	Exam preparation	

Today – Object Detection

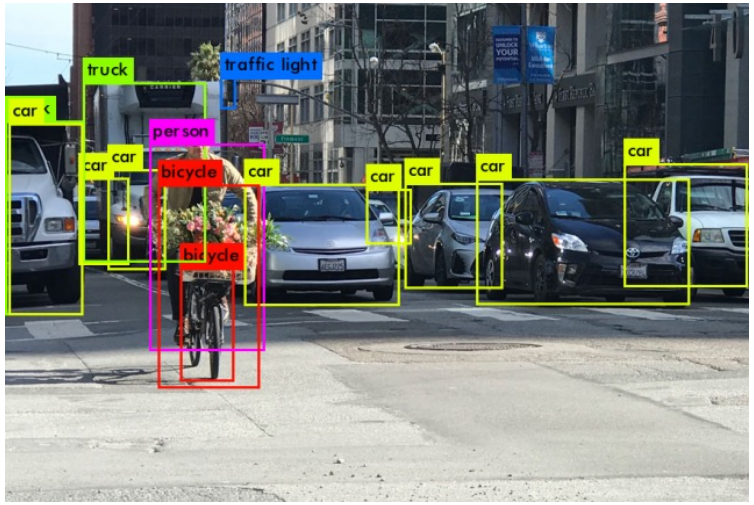
- Why object detection?
- Problem Formulations and General Strategies
- The History of Object Detection (2001 – 2015)
- R-CNN, Fast R-CNN, Faster R-CNN
- Comparing Boxes and Evaluating Object Detectors
- Project 4!!!


Object Detection
bottle
vase
laptop
wine glass
chair
ining table
bottle
mouse
handbag
bowl
cup

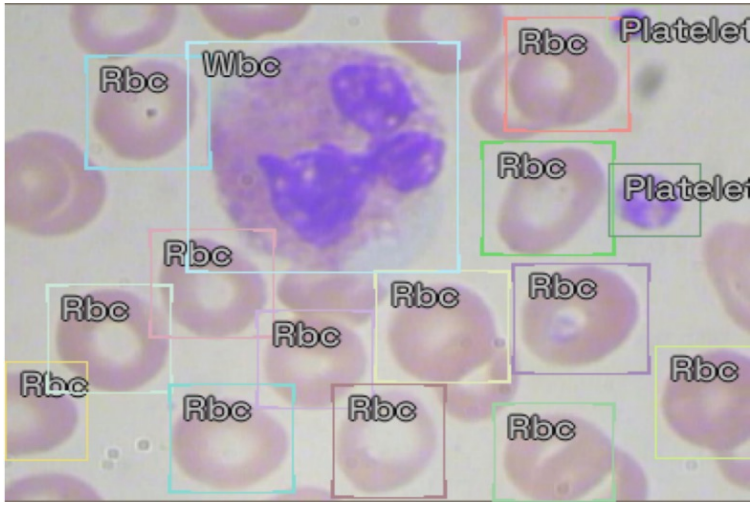

Object Detection
bottle
laptop
vase
wine glass
chair
ining table
bottle
Where? → Localization
What? → Recognition

Object detection applications

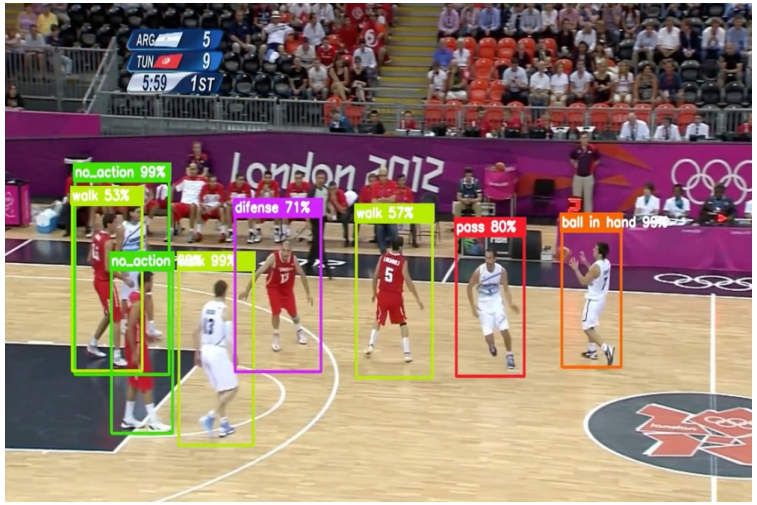
Autonomous driving



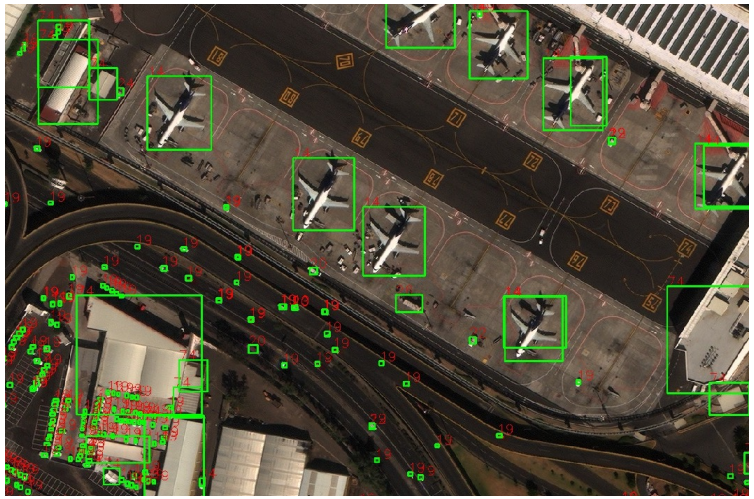
Healthcare



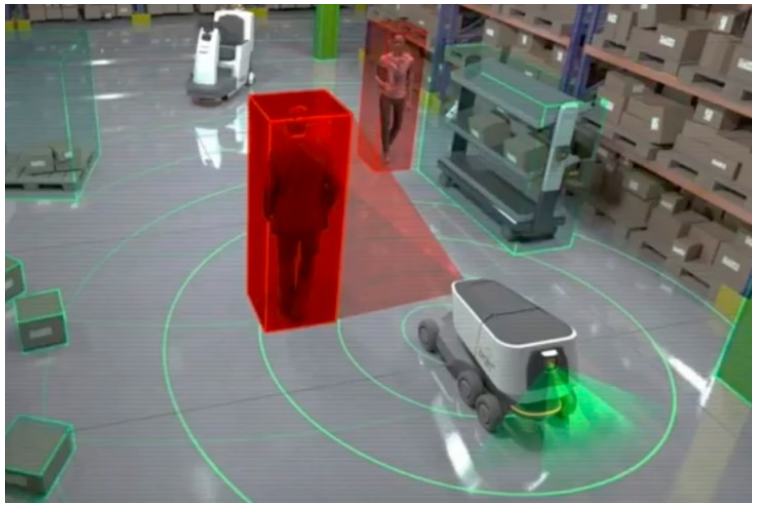
Sport analytics



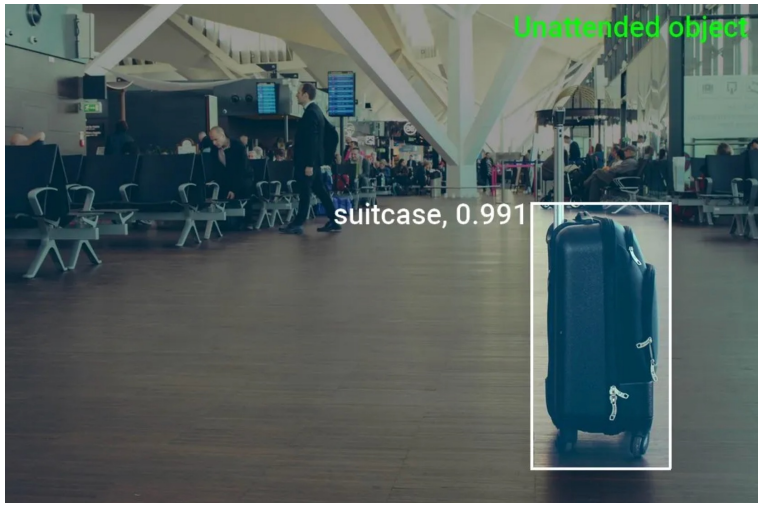
Satellite images



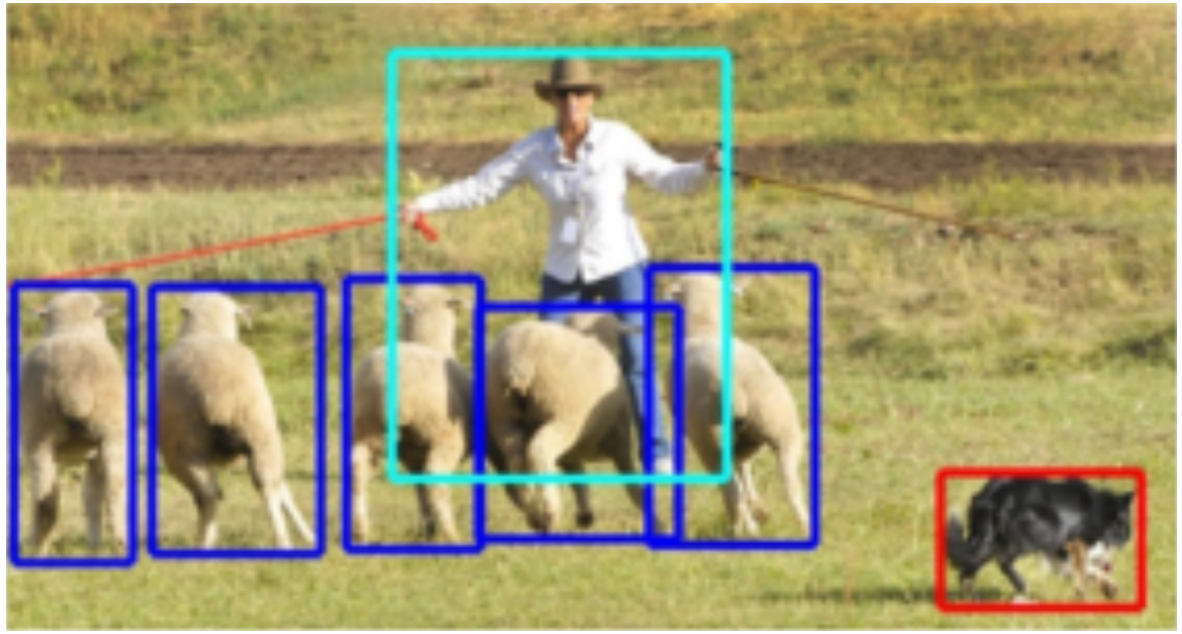
Robot navigation



Security

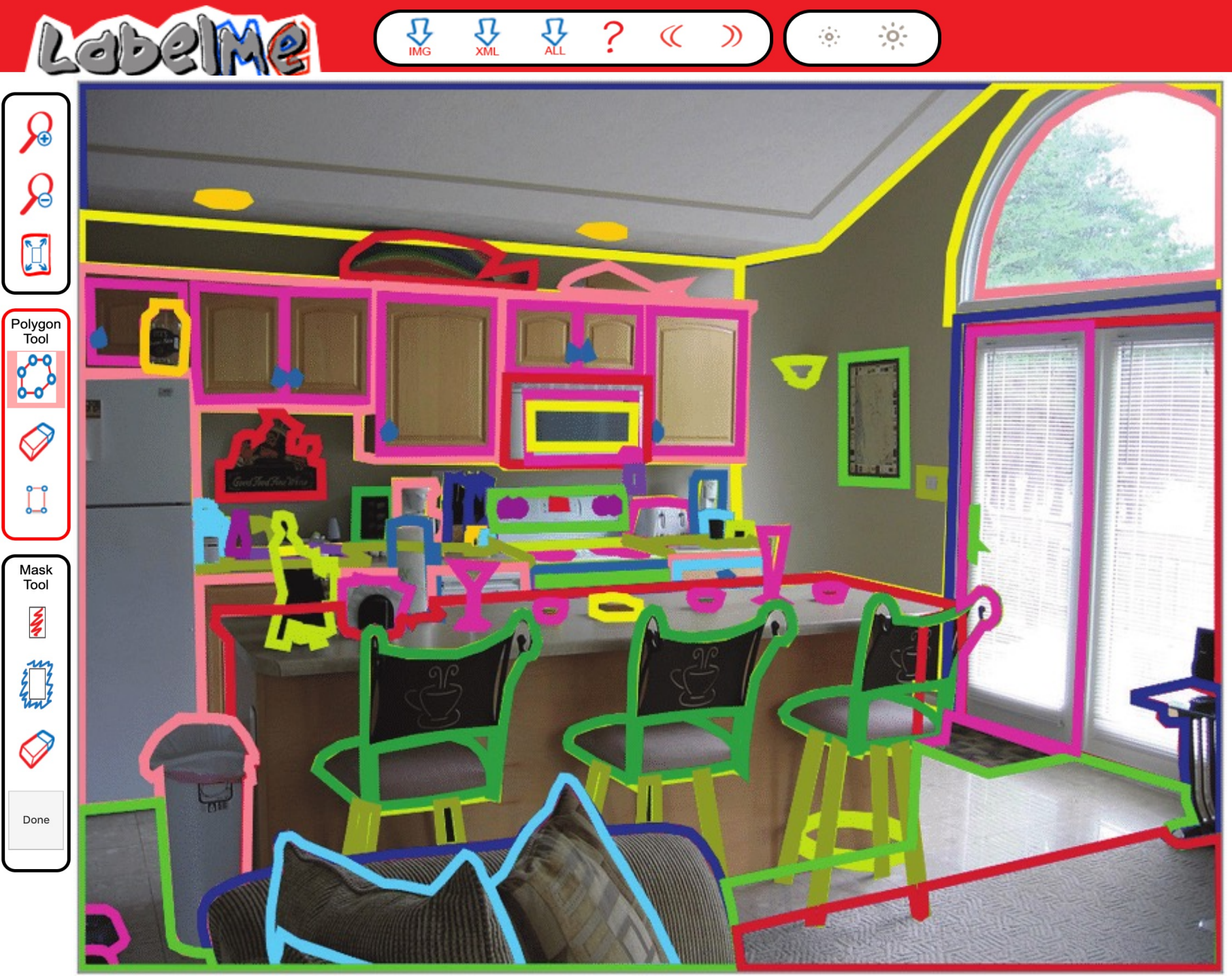


Why bounding boxes???



Object Detection

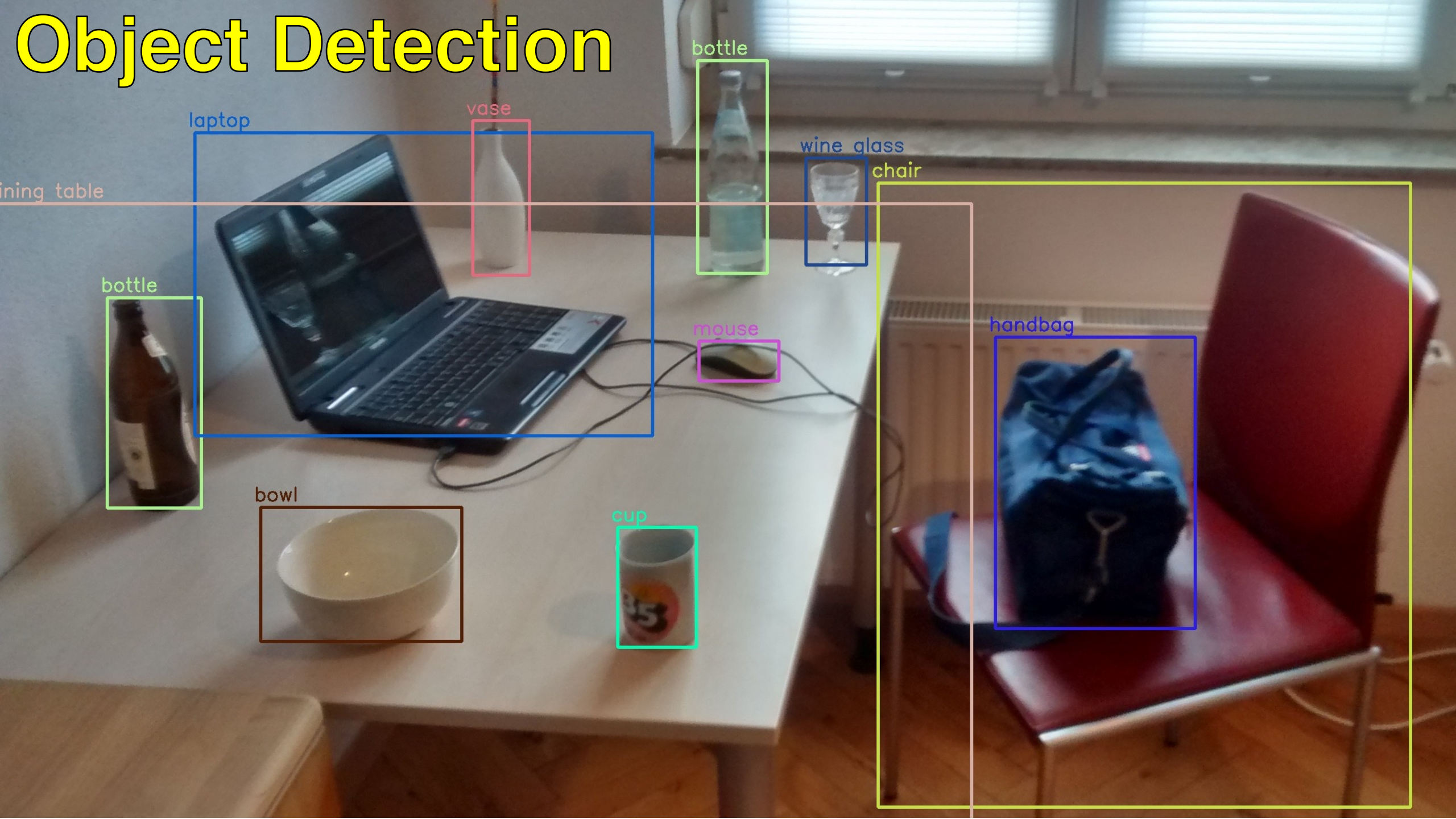
Instance segmentation
(!= semantic segmentation)

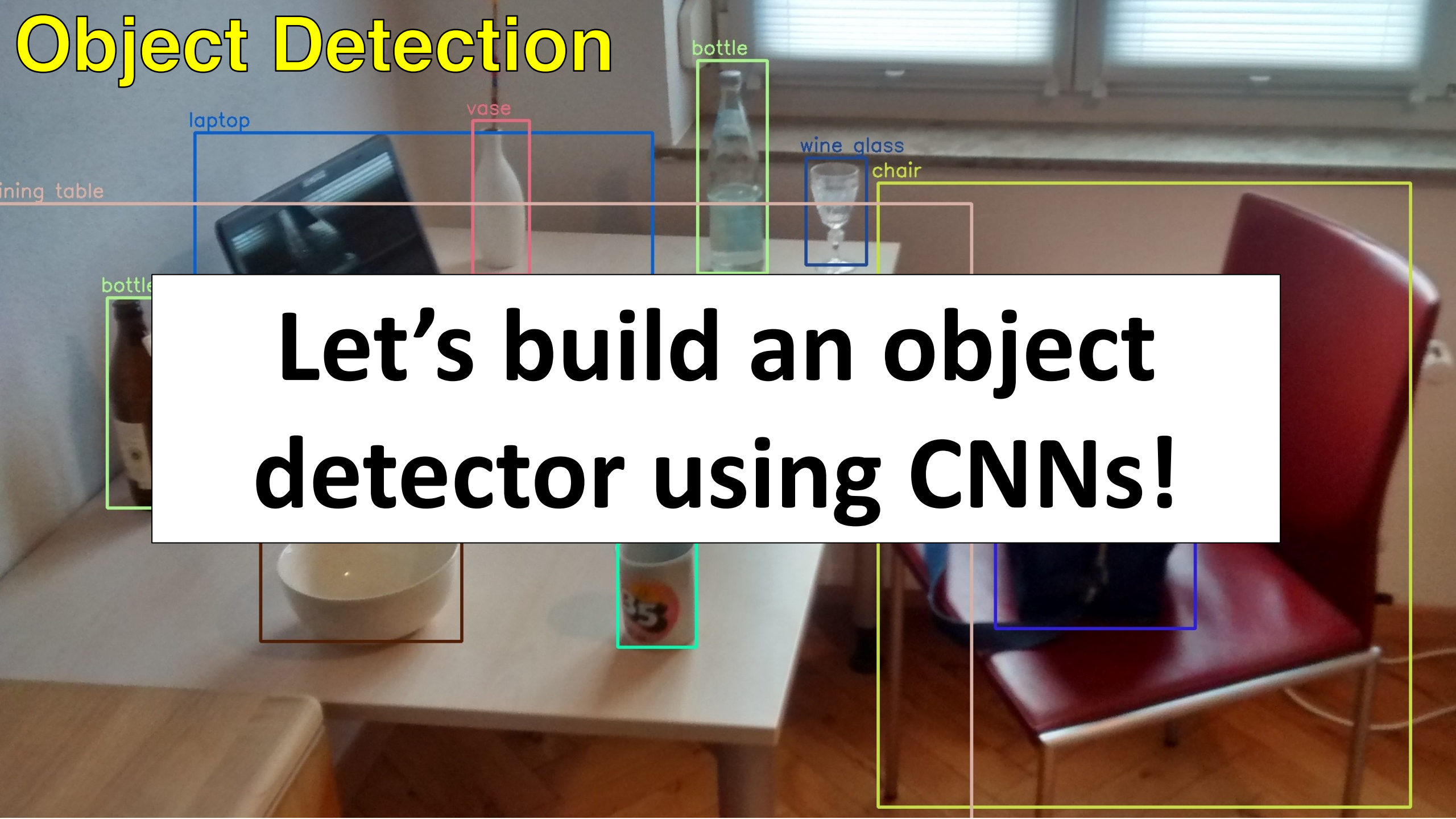


- **Training data?**
- **Expensive!!**

COCO:
80sec per object

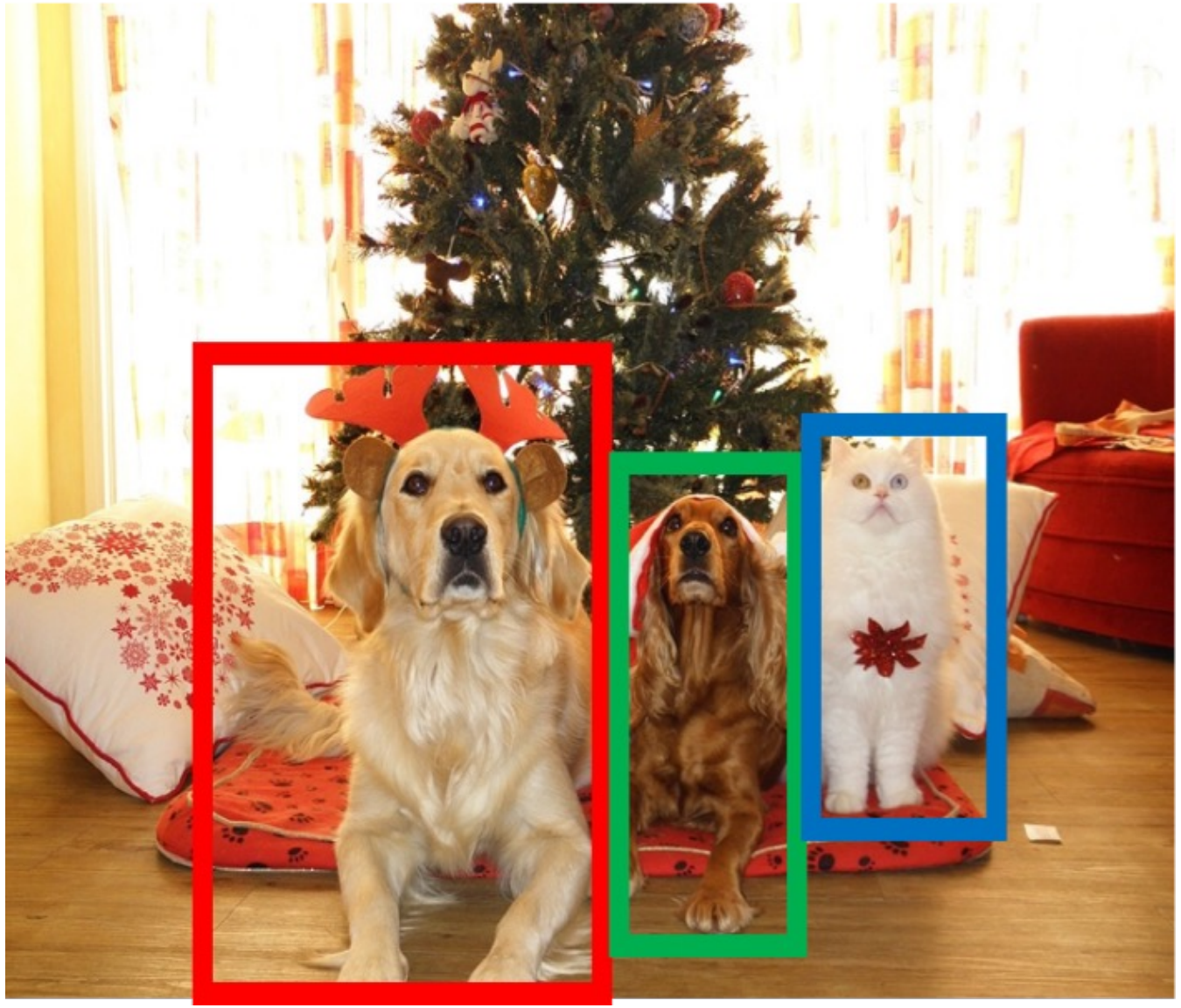
CityScapes:
1.5 hours per image


Object Detection
bottle
vase
laptop
wine glass
chair
ining table
bottle
mouse
handbag
bowl
cup

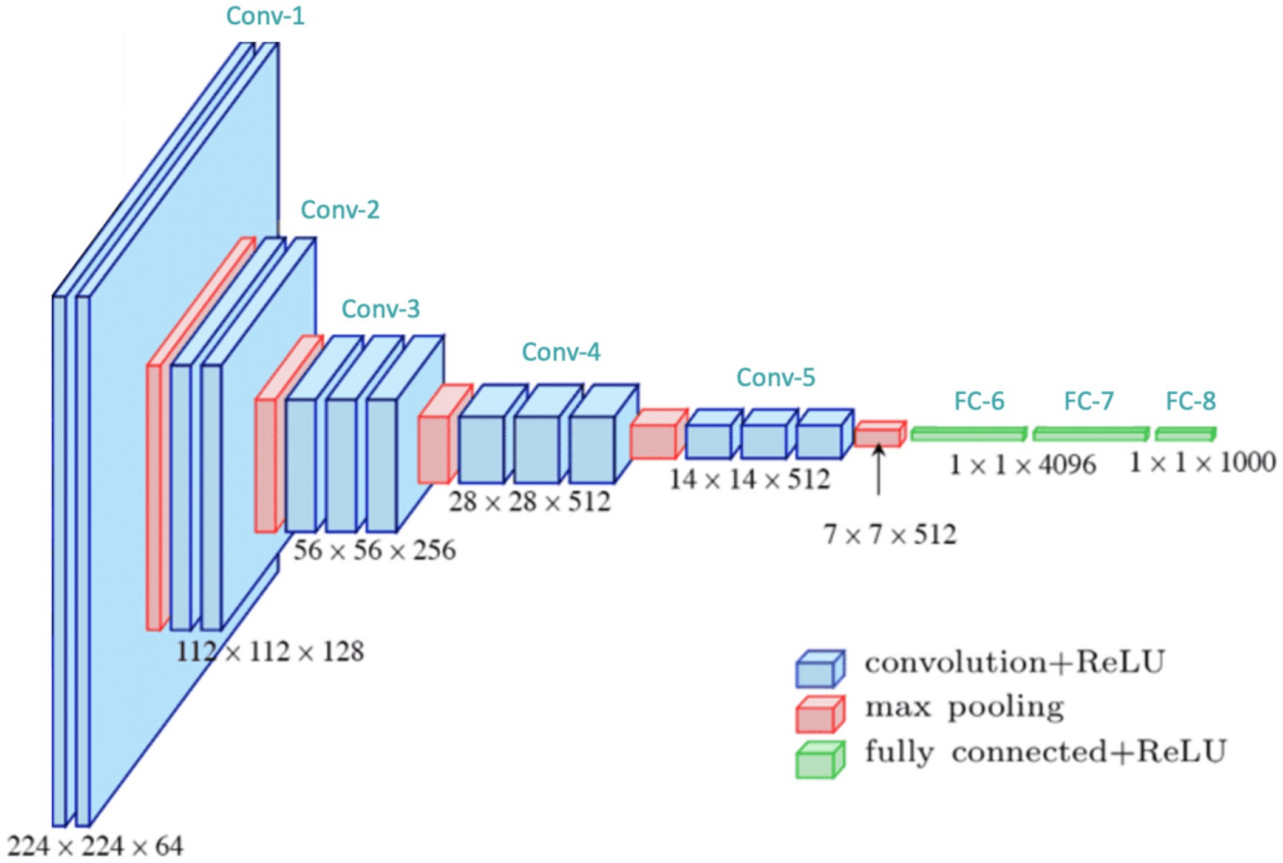

Object Detection
bottle
laptop
vase
wine glass
chair
ining table
bottle
Let’s build an object detector using CNNs!

Object Detection: Task Definition

- Input: Single RGB Image
- Output: A set of detected objects.
- For each object find:
  - Category label:
    - From a fixed known set of categories
  - Bounding box
    - Four numbers: [x, y, width, height]

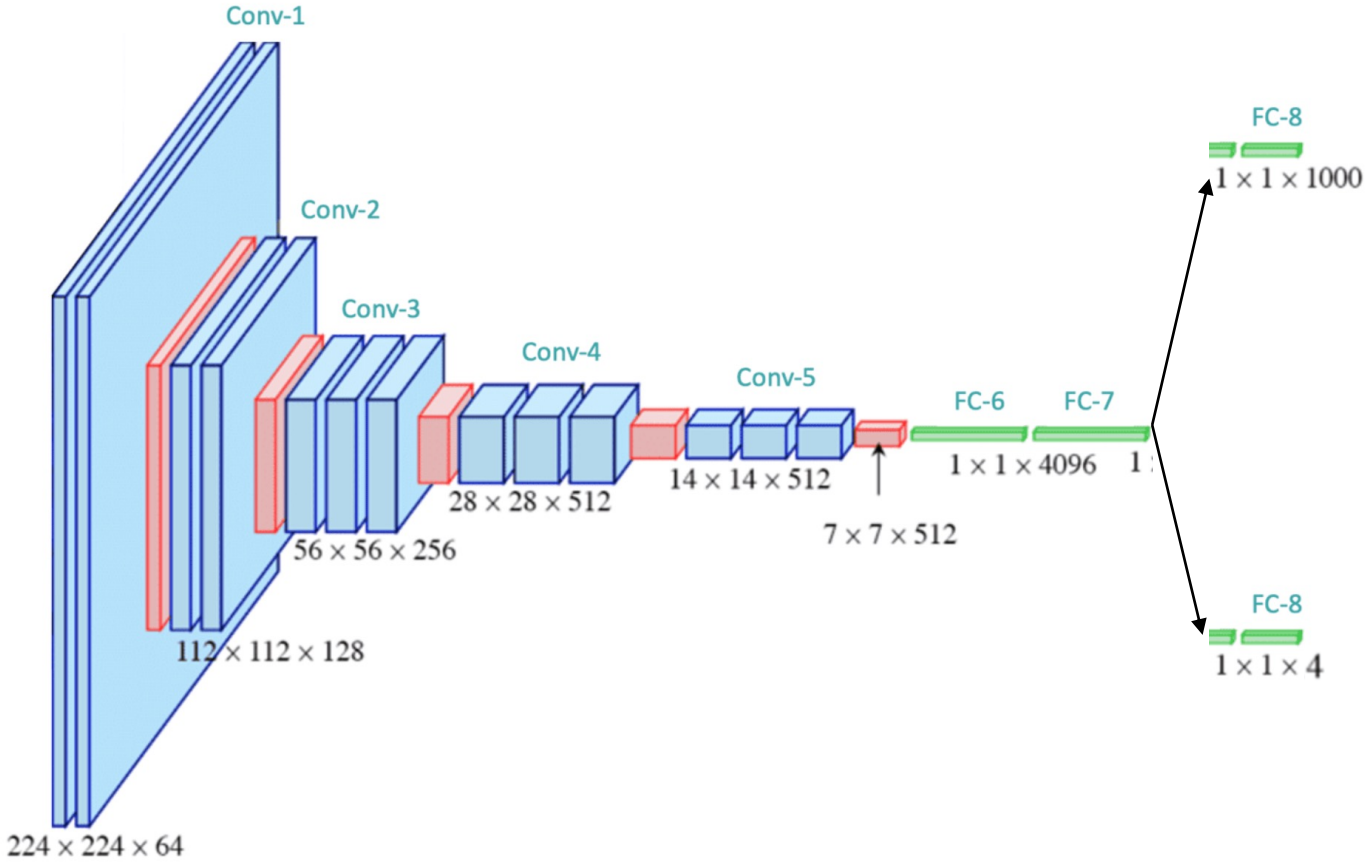


Let's start with a classification network



cat

Add a regression head (FC layer to 4 outputs)



cat: 0.9

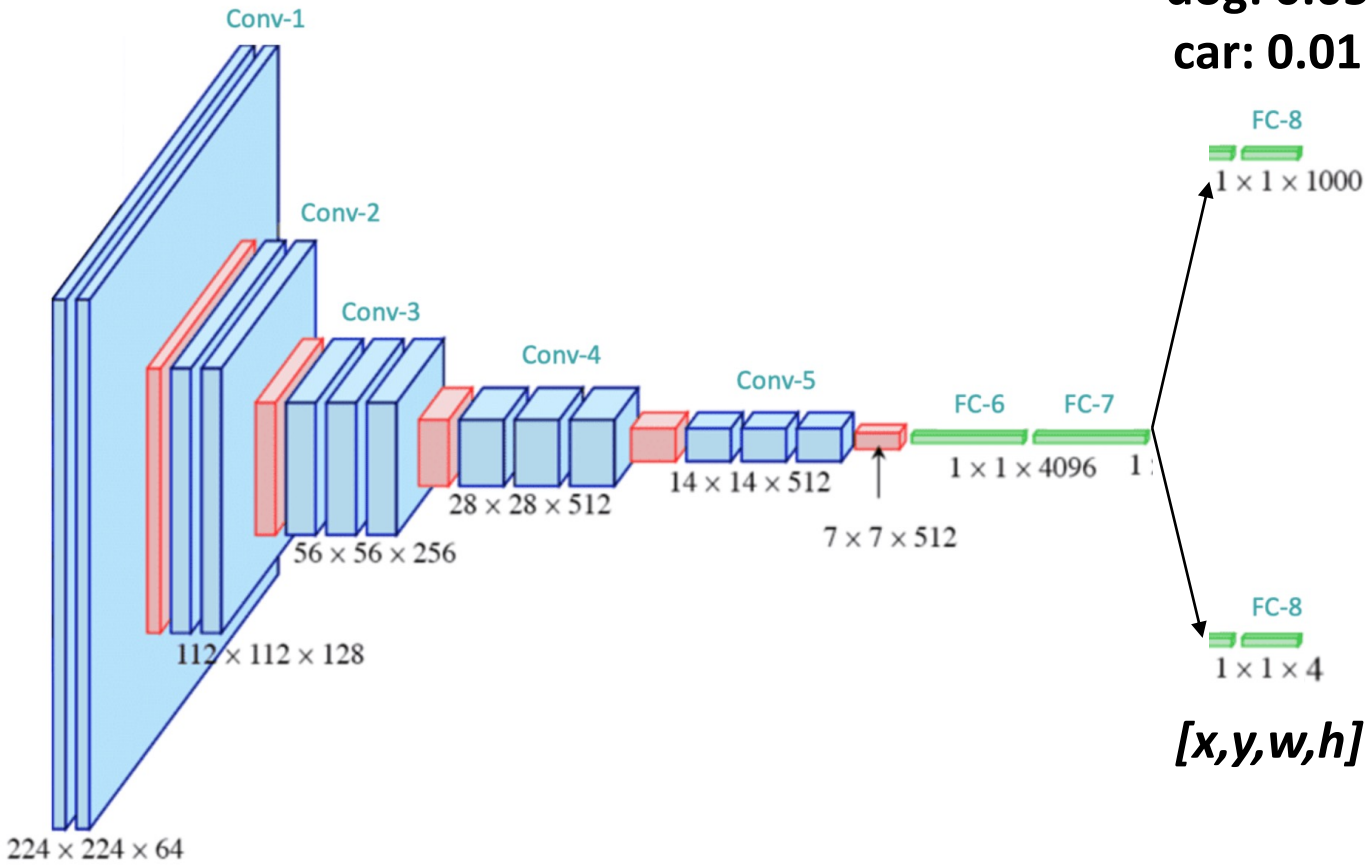
dog: 0.05

car: 0.01

… …

[x,y,w,h]

How to train this model?



cat: 0.9
dog: 0.05
car: 0.01

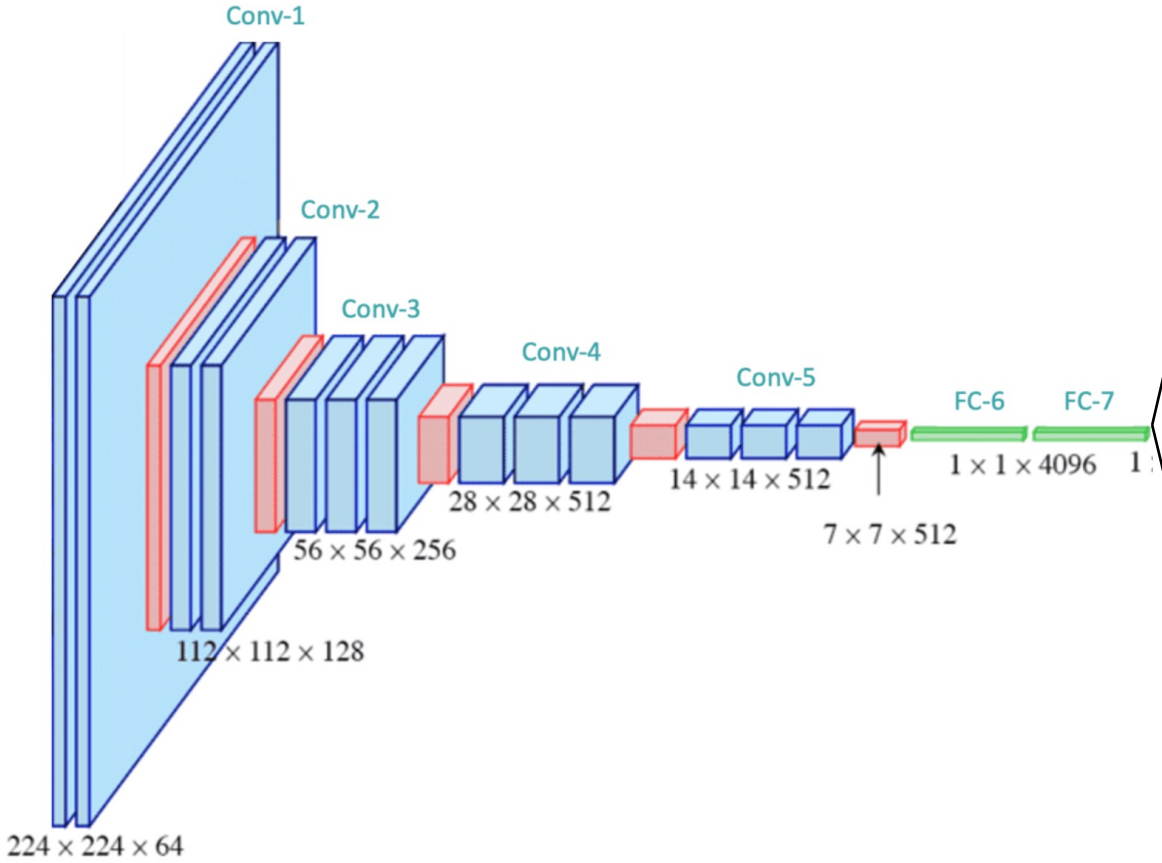
[x,y,w,h]

Correct label:
Cat

"Loss 1"
Difference

"Loss 2"
Difference
Correct box:
(x',y',w',h')

Final loss



cat: 0.9
dog: 0.05
car: 0.01

[x,y,w,h]

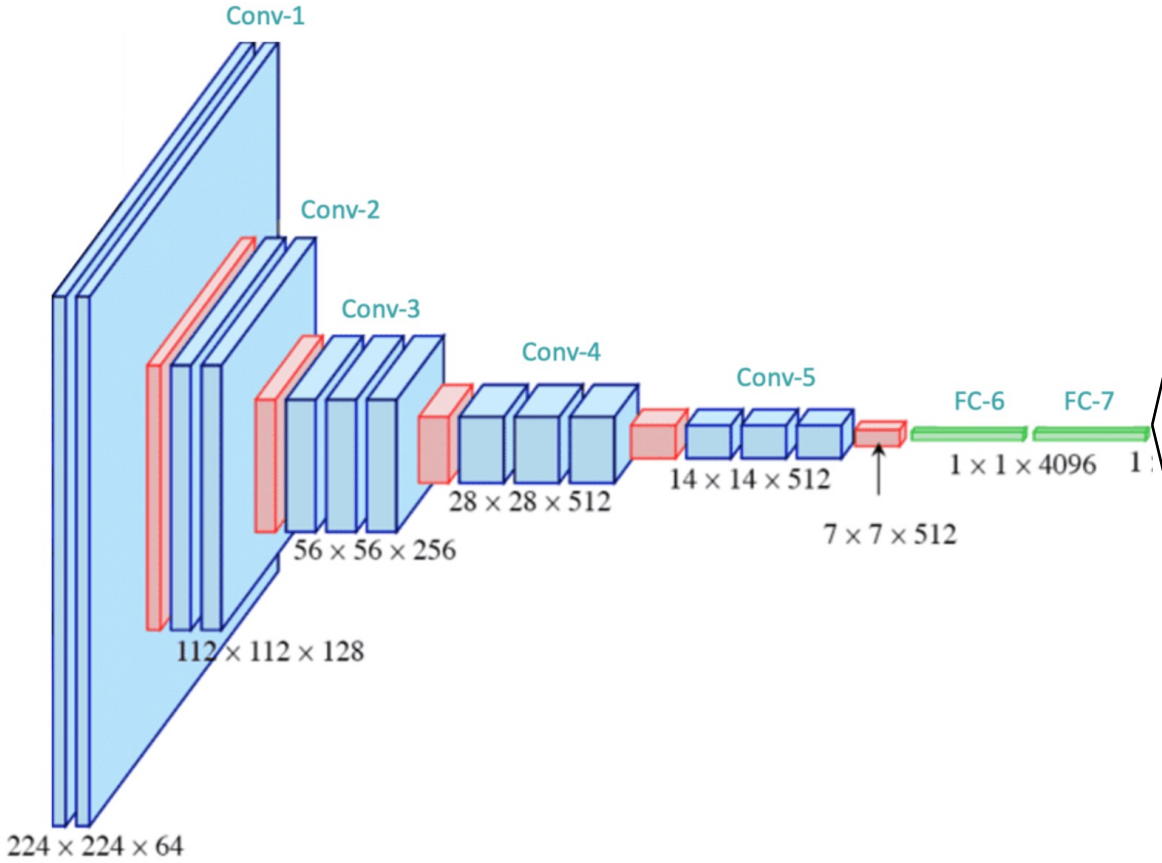
Correct label:
Cat

"Loss 1"
Difference

Loss = (L1 + L2)/2

"Loss 2"
Difference
Correct box:
(x',y',w',h')

BCE and MSE?



cat: 0.9
dog: 0.05
car: 0.01

[x,y,w,h]

Correct label:
Cat

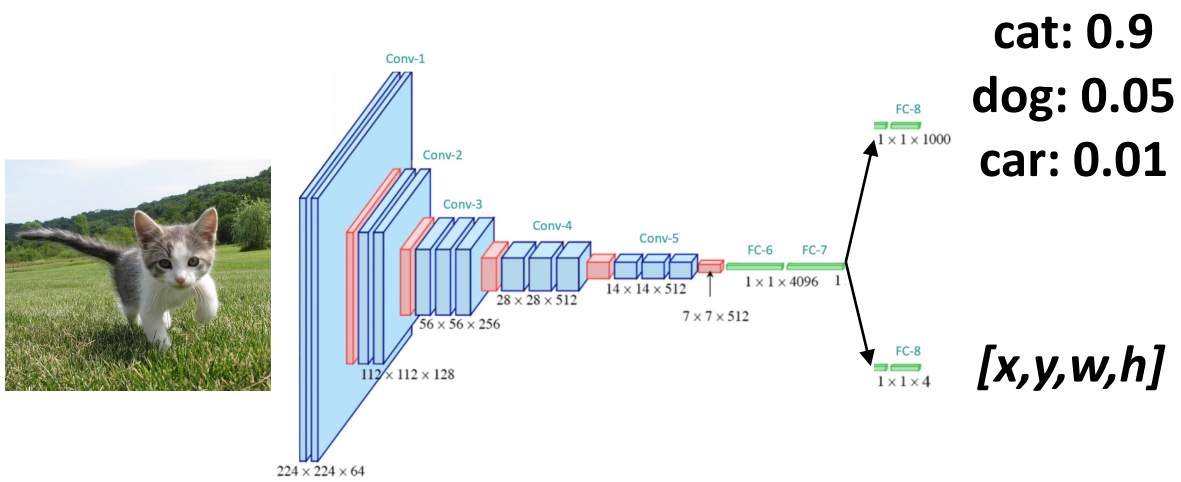
"Loss 1"
Cross-entropy

Loss = (L1 + L2)/2

"Loss 2"
Euclidean distance

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

Quiz Time: Object detection as box regression



Discuss with your neighbor (2min):

(a) Could a system like that work? If so, When?

(b) Are there any limitations?

Quiz Time: Object detection as box regression

Quiz Time: Object detection as box regression

cat: 0.9
dog: 0.9
car: 0.01

[x1,y1,w1,h1]
[x2,y2,w2,h2]

Quiz Time: Object detection as box regression

cat: 0.9
dog: 0.9
car: 0.01

[x1,y1,w1,h1]
[x2,y2,w2,h2]

Multi-label classification
[softmax to sigmoids and BCE to CE]

Need variable sized output!!

Quiz Time: Object detection as box regression

DOG, (x, y, w, h)

CAT, (x, y, w, h)

CAT, (x, y, w, h)

DUCK (x, y, w, h)

= 16 numbers

Quiz Time: Object detection as box regression

DOG, (x, y, w, h)

CAT, (x, y, w, h)

CAT, (x, y, w, h)

DUCK (x, y, w, h)

= 16 numbers

DOG, (x, y, w, h)

CAT, (x, y, w, h)

= 8 numbers

Quiz Time: Object detection as box regression

DOG, (x, y, w, h)

CAT, (x, y, w, h)

CAT, (x, y, w, h)

DUCK (x, y, w, h)

= 16 numbers

DOG, (x, y, w, h)

CAT, (x, y, w, h)

= 8 numbers

CAT, (x, y, w, h)

CAT, (x, y, w, h)

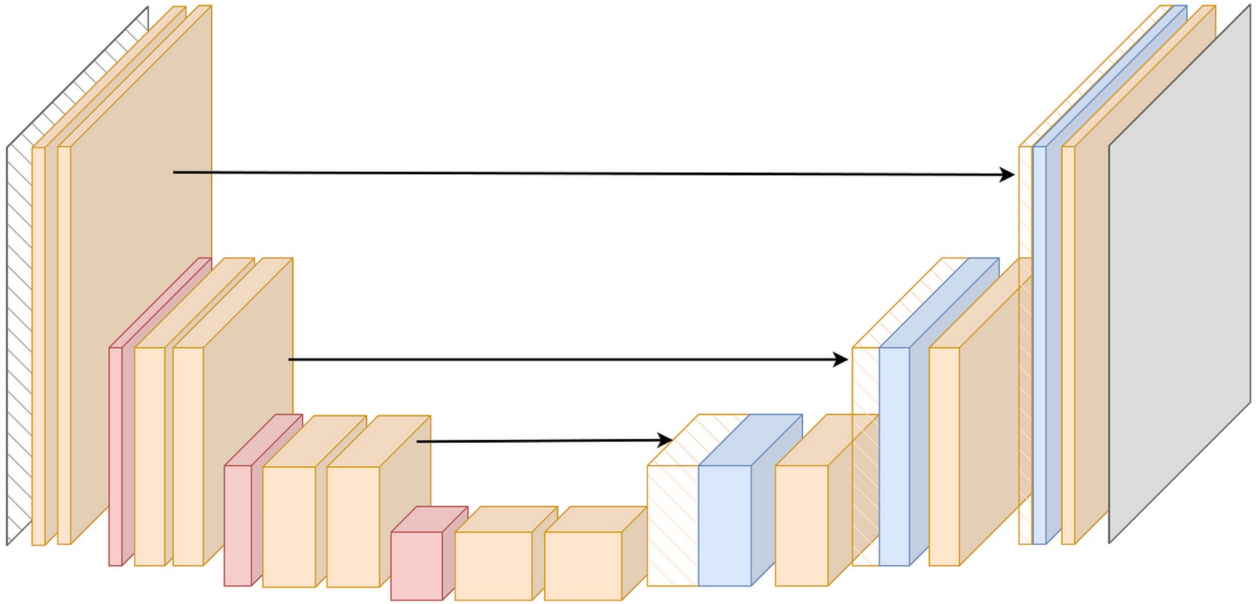
....

CAT (x, y, w, h)

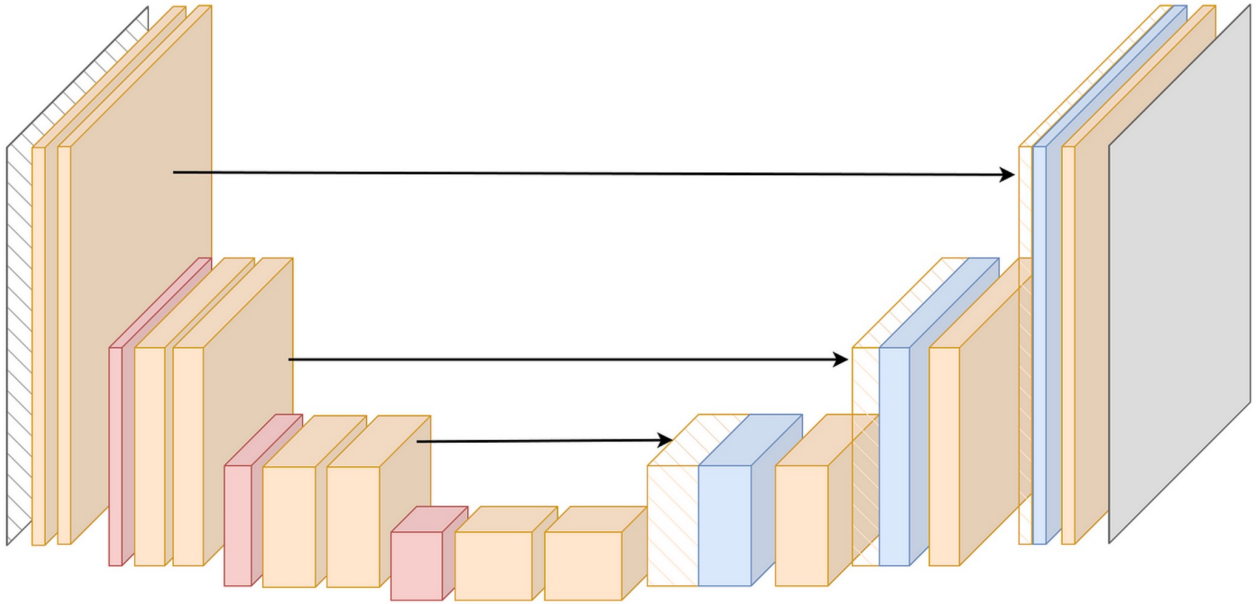
= many numbers

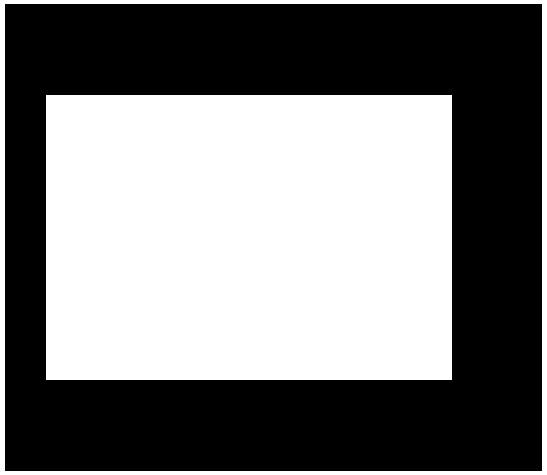
Need variable sized outputs!

I know how to classify pixels

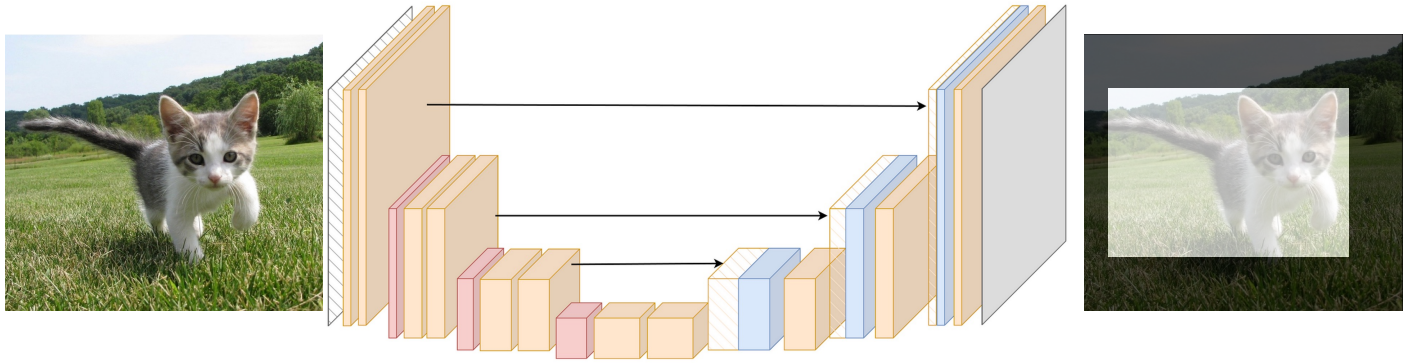


I know how to classify pixels





Quiz Time: Object detection as pixel classification



Discuss with your neighbor (2min):

(a) Could a system like that work? If so, When?

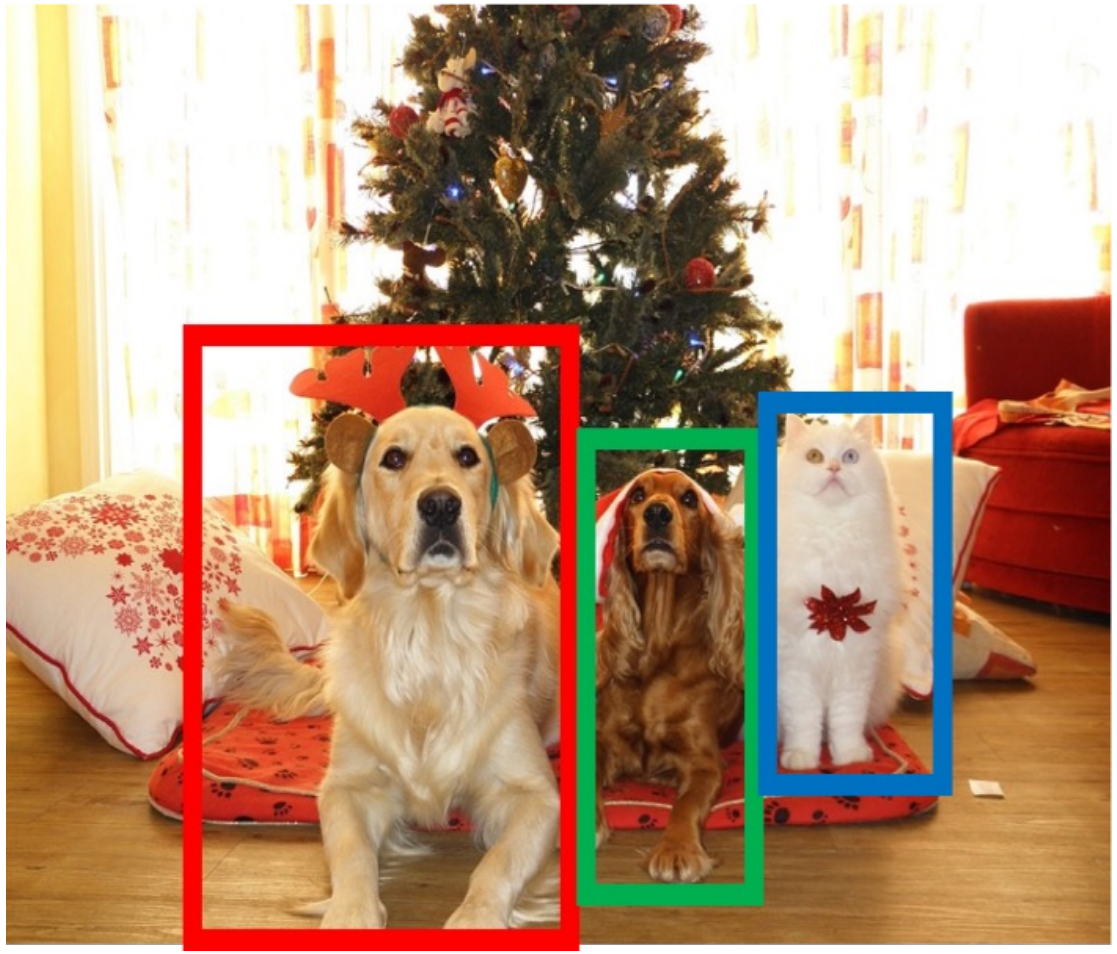
(b) Are there any limitations?

Quiz Time: Object detection as pixel classification

Quiz Time: Object detection as pixel classification

Object Detection: Task Definition

- Multiple outputs:
  - Variable number of objects per image
- Multiple types of output:
  - “ what”: category label
  - “where”: location (bounding box)
- Large images:
  - Classification works at 224x224
  - Need high resolution for detection



Today – Object Detection

- Why object detection?
- Problem Formulations and General Strategies
- **The History of Object Detection (2001 – 2015)**
- R-CNN, Fast R-CNN, Faster R-CNN
- Comparing Boxes and Evaluating Object Detectors


WHERE'S
WALDO?


Keep scanning until you find waldo

The “Waldo” model

The “Waldo” model

The “Waldo” model

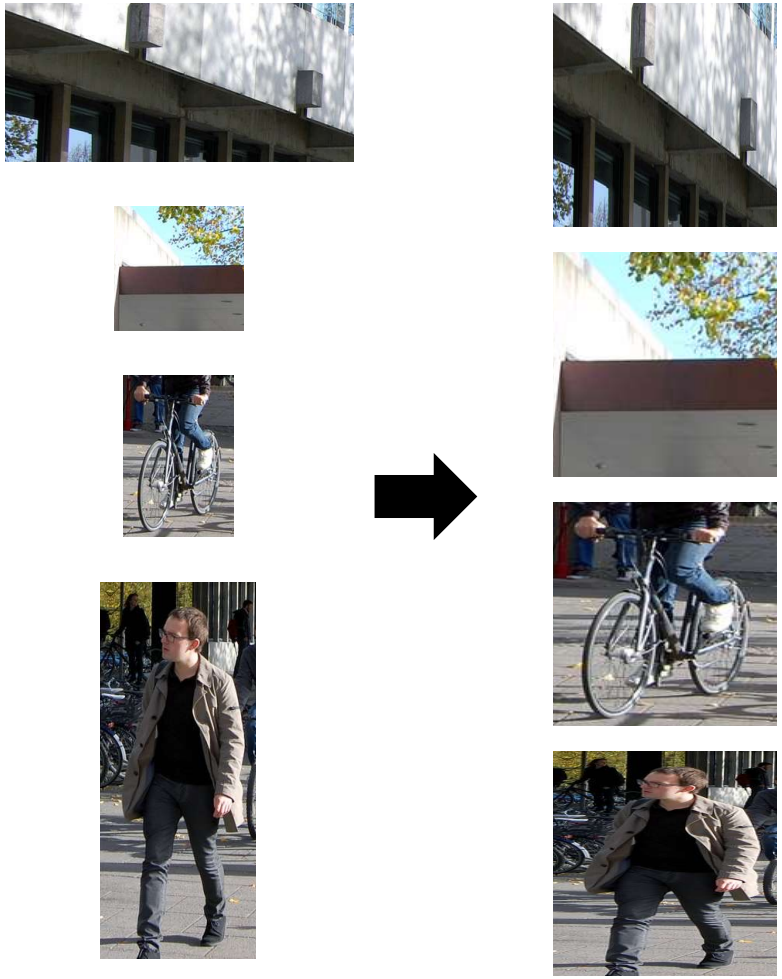
The “Waldo” model

The “Waldo” model

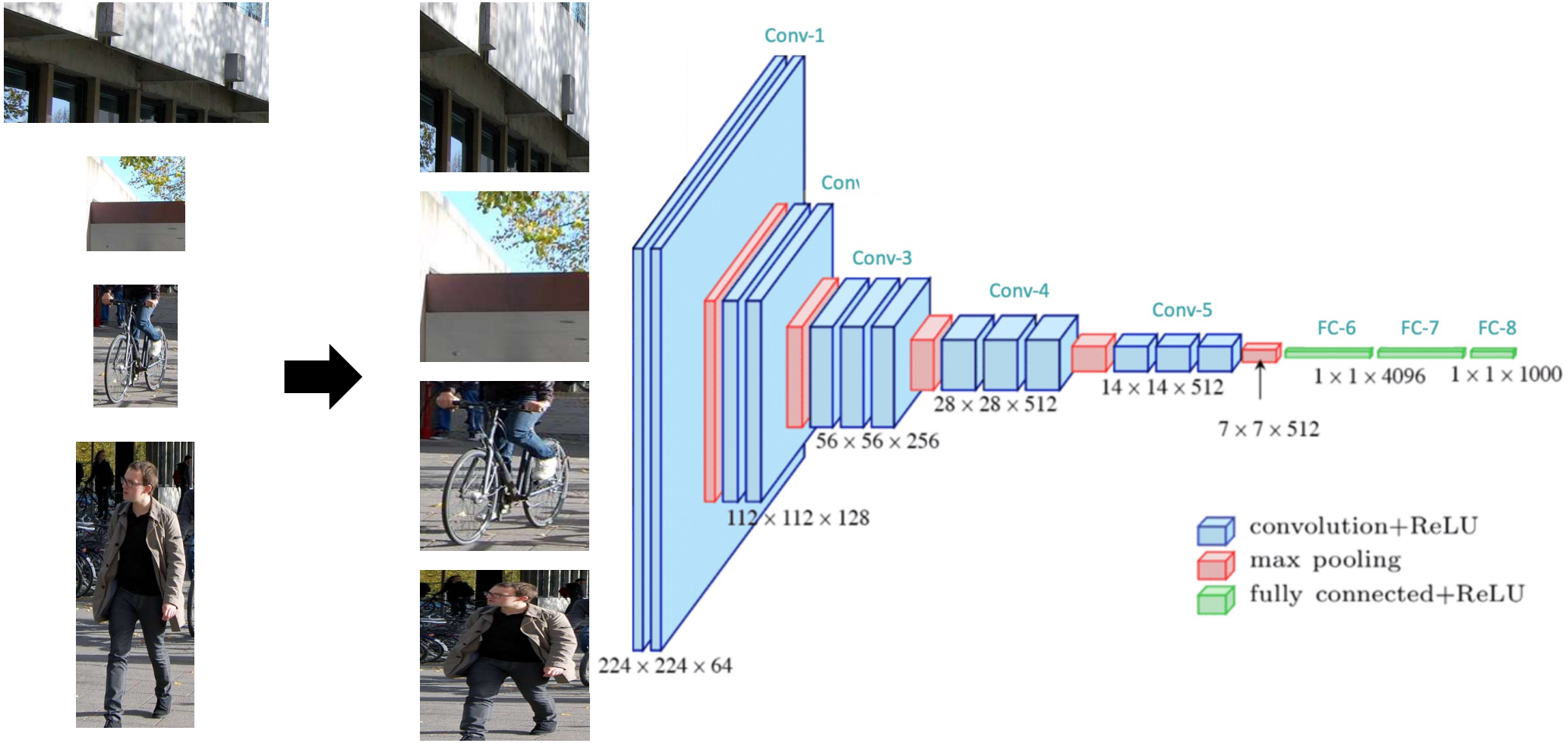
The “Waldo” model

The “Waldo” model

The “Waldo” model



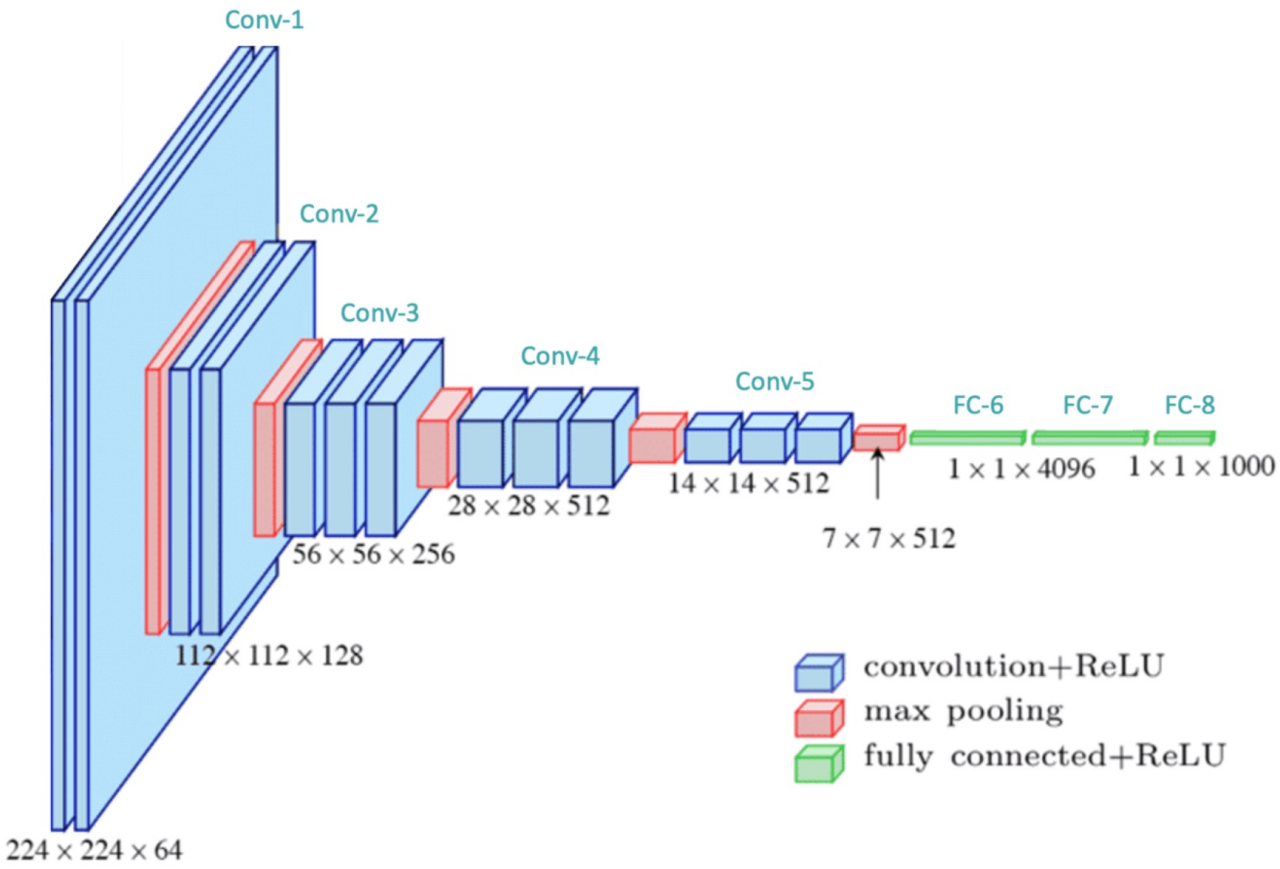
The “Waldo” model



N outputs

person: 0.9
car: 0.02
tree: 0.01
dog: 0.01
…
…

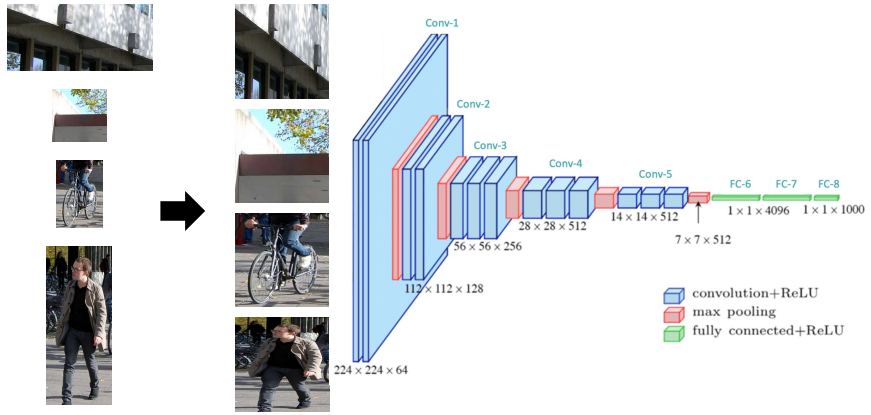
The “Waldo” model

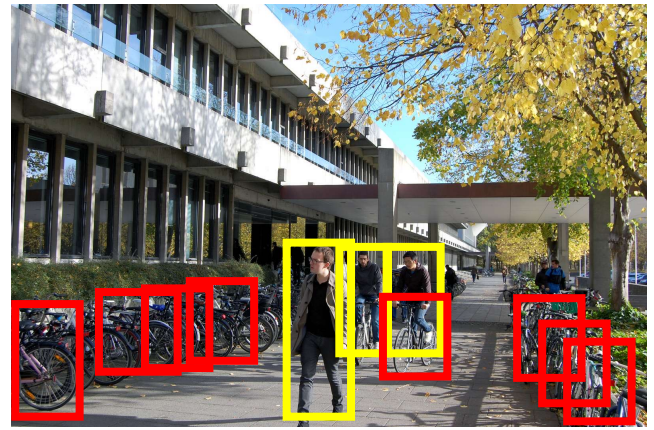


N+1 outputs

person: 0.9
car: 0.02
tree: 0.01
dog: 0.01
…
…
background: 0.01

Quiz Time: Sliding window model





Discuss with your neighbor (2min):

(a) Could a system like that work? If so, When?

(b) Are there any limitations?

Quiz Time: Sliding window model

DOG, (x, y, w, h)

CAT, (x, y, w, h)

CAT, (x, y, w, h)

DUCK (x, y, w, h)

= 16 numbers

DOG, (x, y, w, h)

CAT, (x, y, w, h)

= 8 numbers

CAT, (x, y, w, h)

CAT, (x, y, w, h)

....

CAT (x, y, w, h)

= many numbers

Sliding window → computationally prohibited

2001

Accepted Conference on Computer Vision and Pattern Recognition 2001

Rapid Object Detection using a Boosted Cascade of Simple Features

Paul Viola
viola@merl.com
Mitsubishi Electric Research Labs
201 Broadway, 8th FL
Cambridge, MA 02139

Michael Jones
mjones@crl.dec.com
Compaq CRL
One Cambridge Center
Cambridge, MA 02142

Abstract

This paper describes a machine learning approach for visual object detection which is capable of processing images extremely rapidly and achieving high detection rates. This work is distinguished by three key contributions. The first is the introduction of a new image representation called the "Integral Image" which allows the features used by our detector to be computed very quickly. The second is a learning algorithm, based on AdaBoost, which selects a small number of critical visual features from a larger set and yields extremely efficient classifiers[6]. The third contribution is a method for combining increasingly more complex classifiers in a "cascade" which allows background regions of the image to be quickly discarded while spending more computation on promising object-like regions. The cascade can be viewed as an object specific focus-of-attention mechanism which unlike previous approaches provides statistical guarantees that discarded regions are unlikely to contain the object of interest. In the domain of face detection the system yields detection rates comparable to the best previous systems. Used in real-time applications, the detector runs at 15 frames per second without resorting to image differencing or skin color detection.

1. Introduction

This paper brings together new algorithms and insights to construct a framework for robust and extremely rapid object detection. This framework is demonstrated on, and in part motivated by, the task of face detection. Toward this end we have constructed a frontal face detection system which achieves detection and false positive rates which are equivalent to the best published results [16, 12, 15, 11, 1]. This face detection system is most clearly distinguished from previous approaches in its ability to detect faces extremely rapidly. Operating on 384 by 288 pixel images, faces are detected at 15 frames per second on a conventional 700 MHz Intel Pentium III. In other face detection systems, auxiliary information, such as image differences in video sequences, or pixel color in color images, have been used to achieve high frame rates. Our system achieves high frame rates working only with the information present in a single grey scale image. These alternative sources of information can also be integrated with our system to achieve even higher frame rates.

There are three main contributions of our object detection framework. We will introduce each of these ideas briefly below and then describe them in detail in subsequent sections.

The first contribution of this paper is a new image representation called an *integral image* that allows for very fast feature evaluation. Motivated in part by the work of Papageorgiou et al. our detection system does not work directly with image intensities [10]. Like these authors we use a set of features which are reminiscent of Haar Basis functions (though we will also use related filters which are more complex than Haar filters). In order to compute these features very rapidly at many scales we introduce the integral image representation for images. The integral image can be computed from an image using a few operations per pixel. Once computed, any one of these Harr-like features can be computed at any scale or location in *constant* time.

The second contribution of this paper is a method for constructing a classifier by selecting a small number of important features using AdaBoost [6]. Within any image subwindow the total number of Harr-like features is very large, far larger than the number of pixels. In order to ensure fast classification, the learning process must exclude a large majority of the available features, and focus on a small set of critical features. Motivated by the work of Tieu and Viola, feature selection is achieved through a simple modification of the AdaBoost procedure: the weak learner is constrained so that each weak classifier returned can depend on only a

2011-2014

Object Detection with Discriminatively Trained Part Based Models

Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester and Deva Ramanan

Abstract—We describe an object detection system based on mixtures of multiscale deformable part models. Our system is able to represent highly variable object classes and achieves state-of-the-art results in the PASCAL object detection challenges. While deformable part models have become quite popular, their value had not been demonstrated on difficult benchmarks such as the PASCAL datasets. Our system relies on new methods for discriminative training with partially labeled data. We combine a margin-sensitive approach for data-mining hard negative examples with a formalism we call latent SVM. A latent SVM is a reformulation of MI-SVM in terms of latent variables. A latent SVM is semi-convex and the training problem becomes convex once latent information is specified for the positive examples. This leads to an iterative training algorithm that alternates between fixing latent values for positive examples and optimizing the latent SVM objective function.

Index Terms—Object Recognition, Deformable Models, Pictorial Structures, Discriminative Training, Latent SVM

1 Introduction

Object recognition is one of the fundamental challenges in computer vision. In this paper we consider the problem of detecting and localizing generic objects from categories such as people or cars in static images. This is a difficult problem because objects in such categories can vary greatly in appearance. Variations arise not only from changes in illumination and viewpoint, but also due to non-rigid deformations, and intraclass variability in shape and other visual properties. For example, people wear different clothes and take a variety of poses while cars come in a various shapes and colors.

We describe an object detection system that represents highly variable objects using mixtures of multiscale deformable part models. These models are trained using a discriminative procedure that only requires bounding boxes for the objects in a set of images. The resulting system is both efficient and accurate, achieving state-of-the-art results on the PASCAL VOC benchmarks [11]–[13] and the INRIA Person dataset [10].

Our approach builds on the pictorial structures framework [15], [20]. Pictorial structures represent objects by a collection of parts arranged in a deformable configuration. Each part captures local appearance properties of an object while the deformable configuration is characterized by spring-like connections between certain pairs of parts.

Deformable part models such as pictorial structures provide an elegant framework for object detection. Yet it has been difficult to establish their value in practice. On difficult datasets deformable part models are often outperformed by simpler models such as rigid templates [10] or bag-of-features [44]. One of the goals of our work is to address this performance gap.

While deformable models can capture significant variations in appearance, a single deformable model is often not expressive enough to represent a rich object category. Consider the problem of modeling the appearance of bicycles in photographs. People build bicycles of different types (e.g., mountain bikes, tandems, and 19th-century cycles with one big wheel and a small one) and view them in various poses (e.g., frontal versus side views). The system described here uses mixture models to deal with these more significant variations.

We are ultimately interested in modeling objects using "visual grammars". Grammar based models (e.g. [16], [24], [45]) generalize deformable part models by representing objects using variable hierarchical structures. Each part in a grammar based model can be defined directly or in terms of other parts. Moreover, grammar based models allow for, and explicitly model, structural variations. These models also provide a natural framework for sharing information and computation between different object classes. For example, different models might share reusable parts.

Although grammar based models are powerful, the goal, which we have adopted a research strategy of maintaining a high level performance by enriching Simple models have ticated model machine ampl

• P.F. Felzenszwalb is with the Department of Computer Science, University of Chicago. E-mail: pff@cs.uchicago.edu
• R.B. Girshick is with the Department of Computer Science, University of Chicago. E-mail: rbg@cs.uchicago.edu
• D. McAllester is with the Toyota Technological Institute at Chicago. E-mail: mcallester@tti-c.org
• D. Ramanan is with the Department of Computer Science, UC Irvine. E-mail: dramanan@ics.uci.edu

Today
Object detection

- Various Problem Formulations
- General Strategies for Object Detection
  - Single Object Localization
  - Detection as Regression
  - **Detection as Classification → R-CNN**
- Comparing Boxes
- Evaluating Object Detectors

Detection as classification

CAT? NO

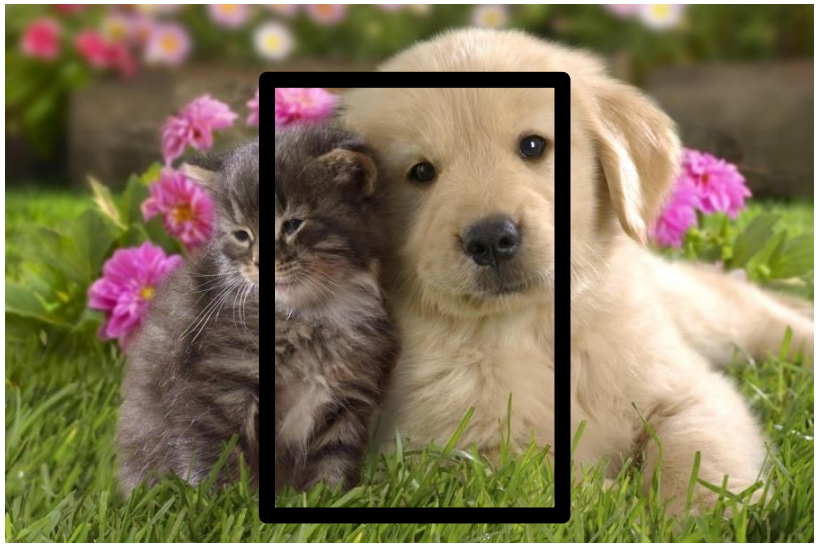
DOG? NO

Detection as classification

CAT? YES

DOG? NO

Detection as classification



CAT? NO

DOG? NO

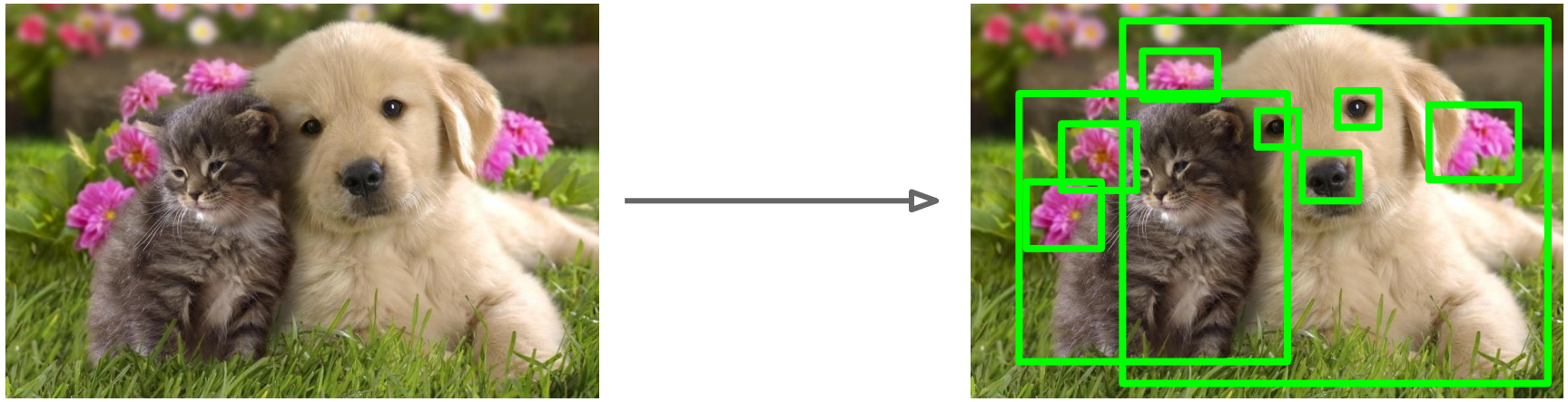
Detection as Classification

Instead of looking at all possible spatial windows at multiple positions and scales (sliding window)

Look only at a **tiny** subset of carefully selected possible positions

Region Proposals

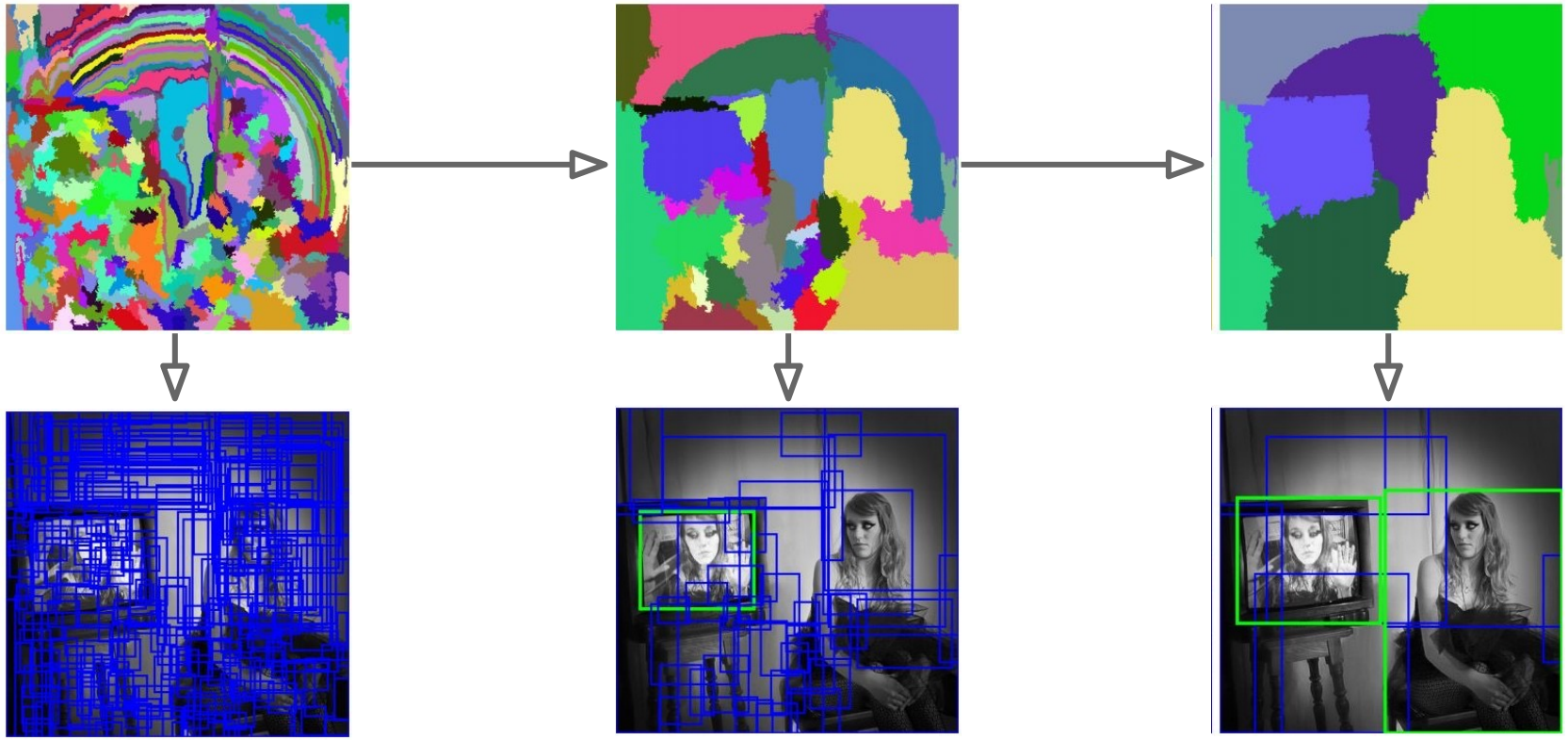
- Find “blobby” image regions that are likely to contain objects
- “Class-agnostic” object detector
- Look for “blob-like” regions



Region Proposals: Selective Search (SS)

Bottom-up segmentation, merging regions at multiple scales

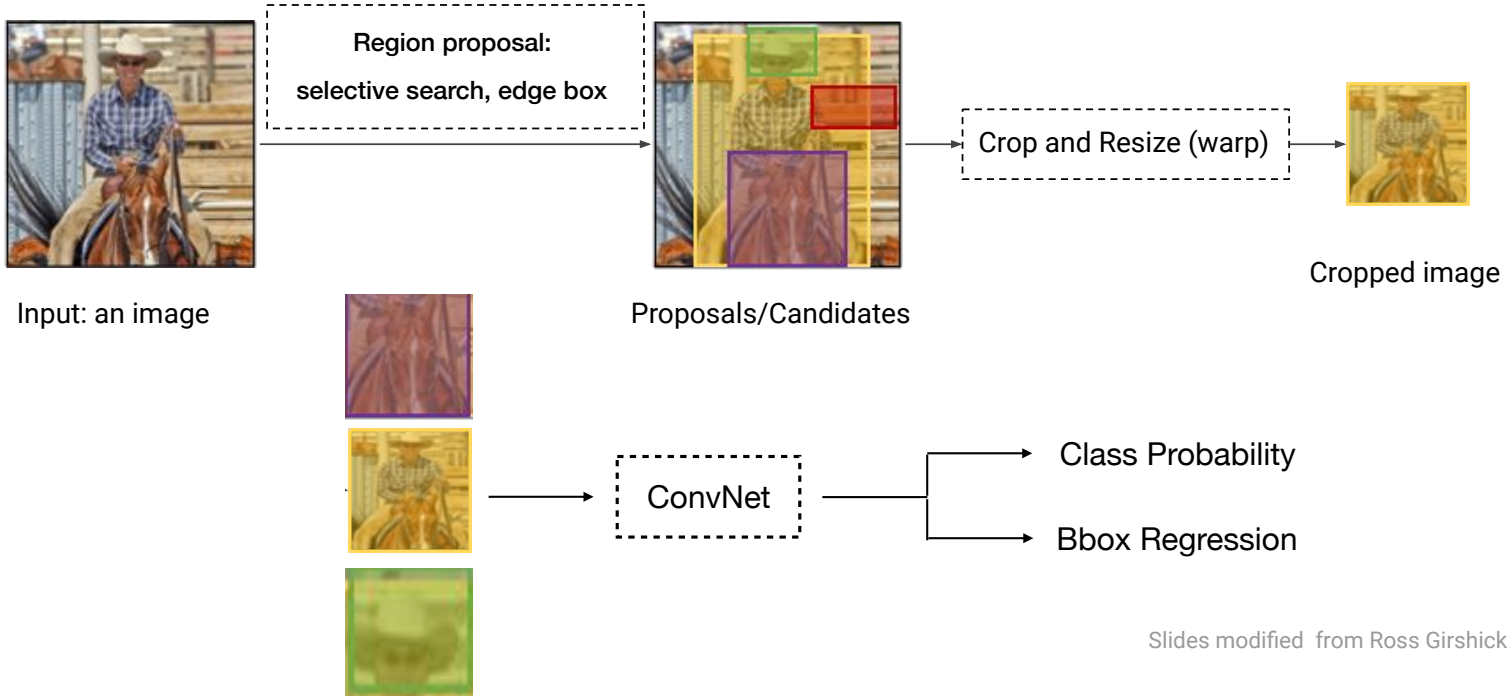
Convert regions to boxes



Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013

All together:
R-CNN: Region-based CNN

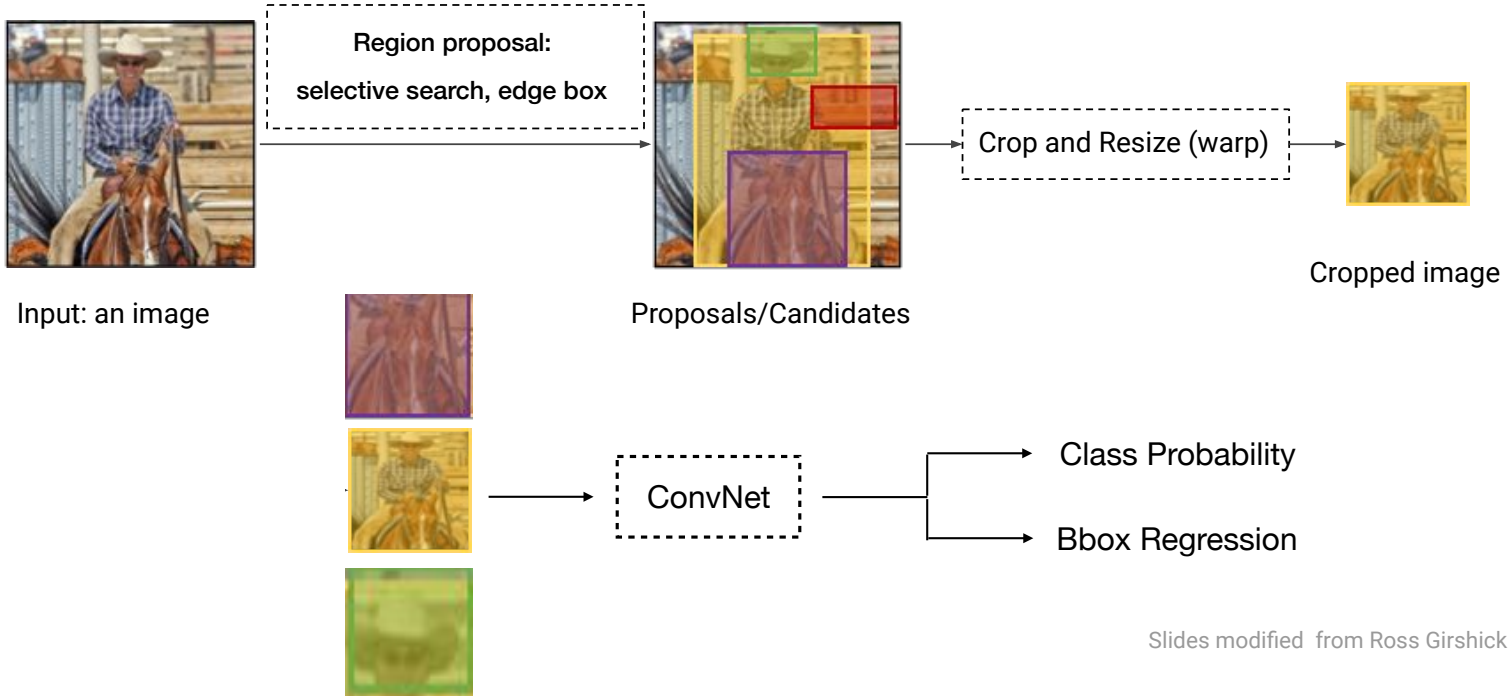
- Propose large number of regions potentially with objects
- Classify each proposed region



Slides modified from Ross Girshick tutorial at CVPR 2019

R-CNN Training

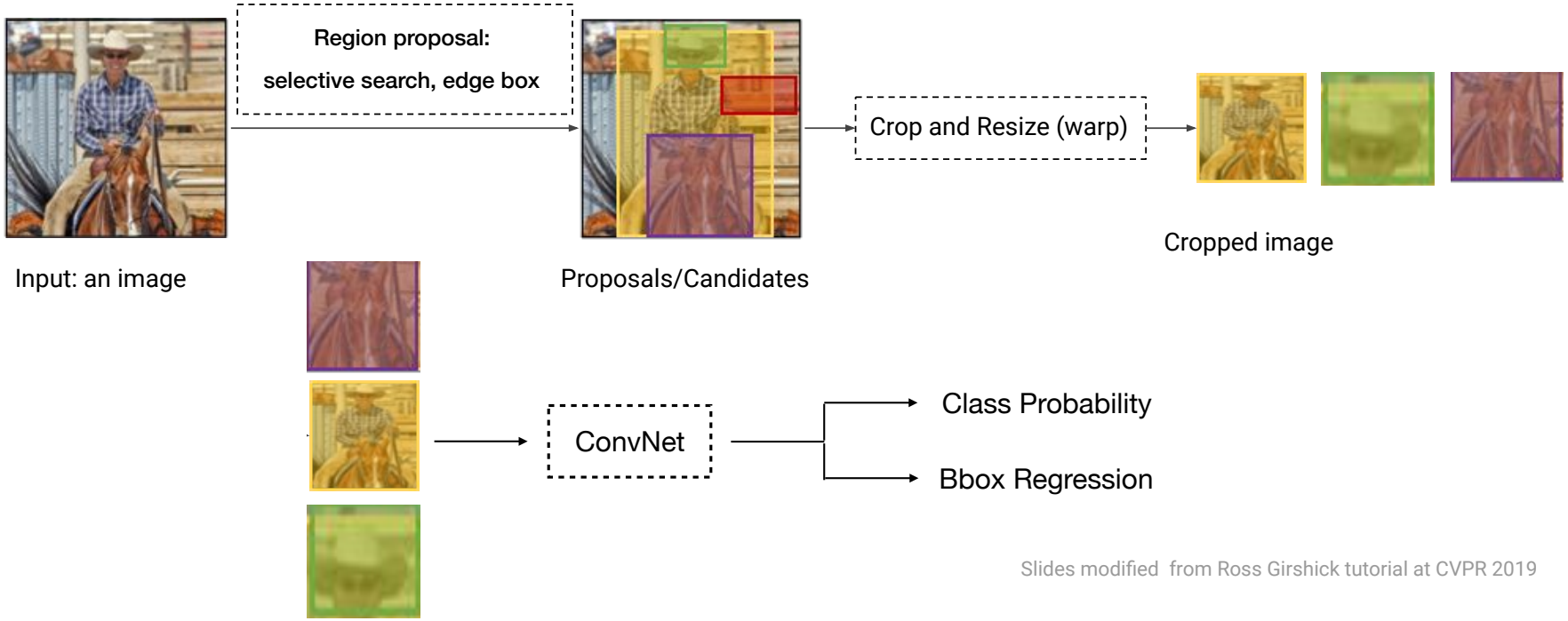
- Step 1: Train (or download) a classification model for ImageNet (AlexNet)



Slides modified from Ross Girshick tutorial at CVPR 2019

R-CNN Training

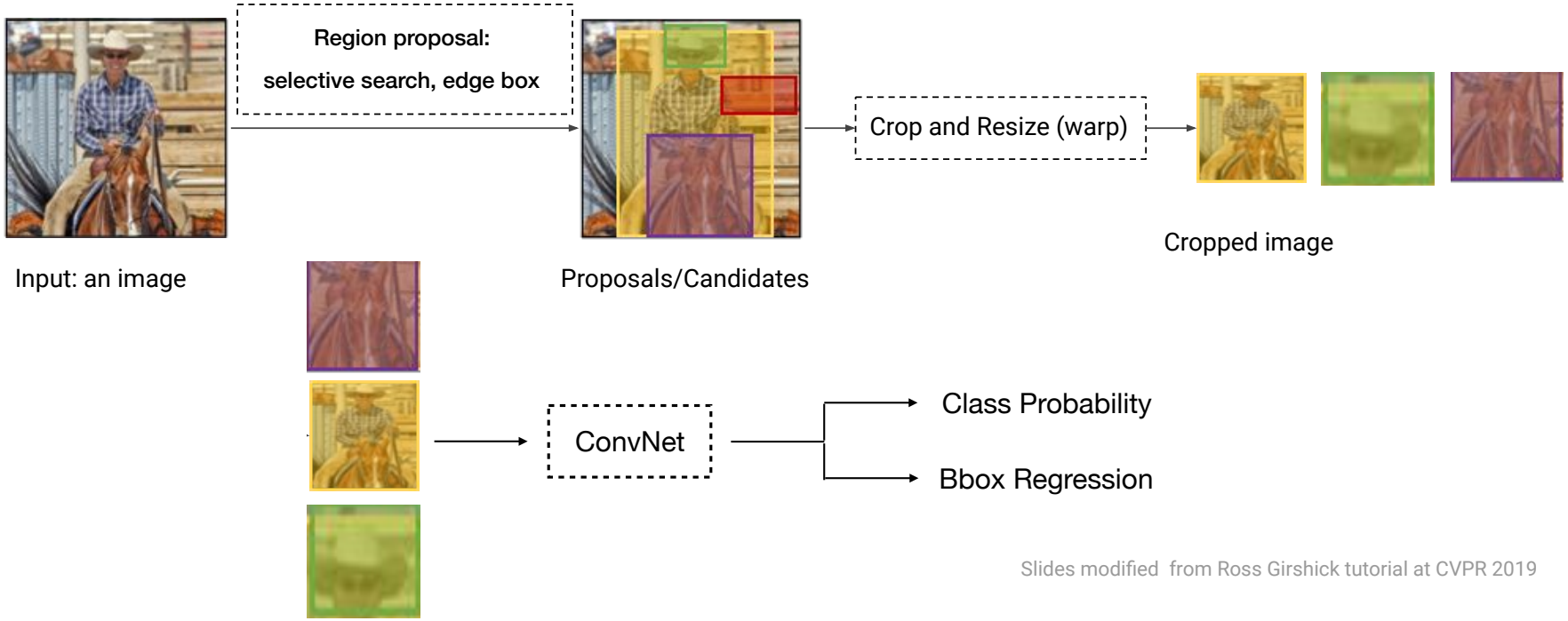
- Step 2: Fine-tune model for detection:
  - Instead of 1000 ImageNet classes → 20 object classes + 1 background
  - Throw away fc layer, re-initialize it
  - Input: Instead of images → Region Proposals (cropped and resized)



Slides modified from Ross Girshick tutorial at CVPR 2019

R-CNN Training

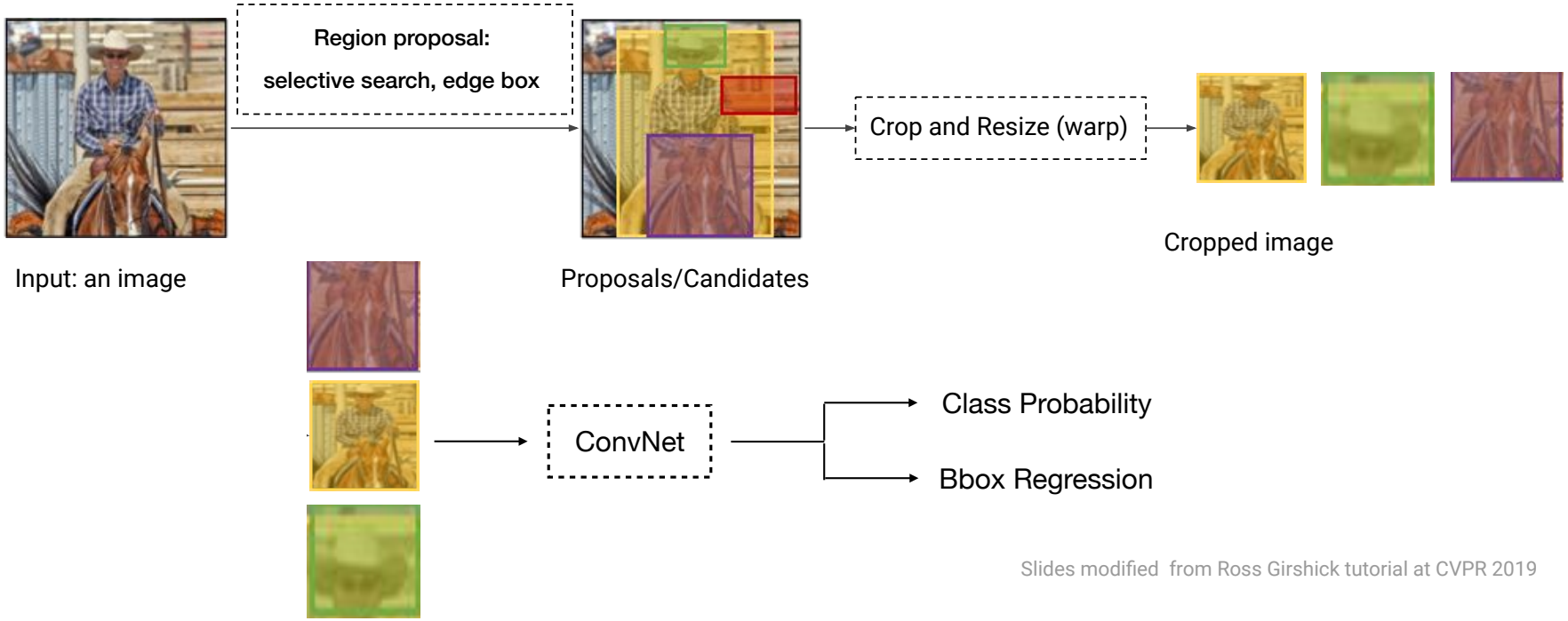
- Step 3: Extract features:
  - Input: Instead of images → Region Proposals (cropped and resized)
  - Save pool5 features to disk → ~100GB for a dataset of 10k images with 20 object classes (PASCAL VOC 2007)



Slides modified from Ross Girshick tutorial at CVPR 2019

R-CNN Training

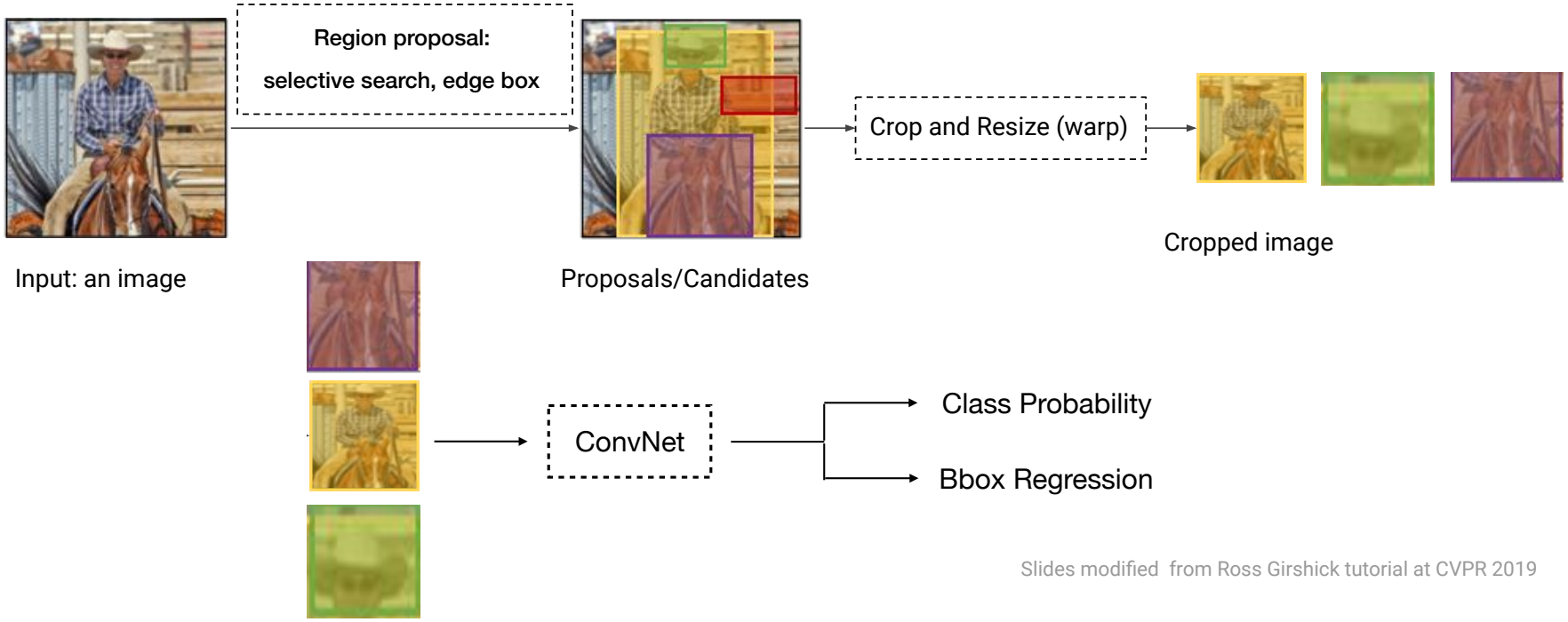
- Step 4: Train a binary SVM per class to classify region features



Slides modified from Ross Girshick tutorial at CVPR 2019

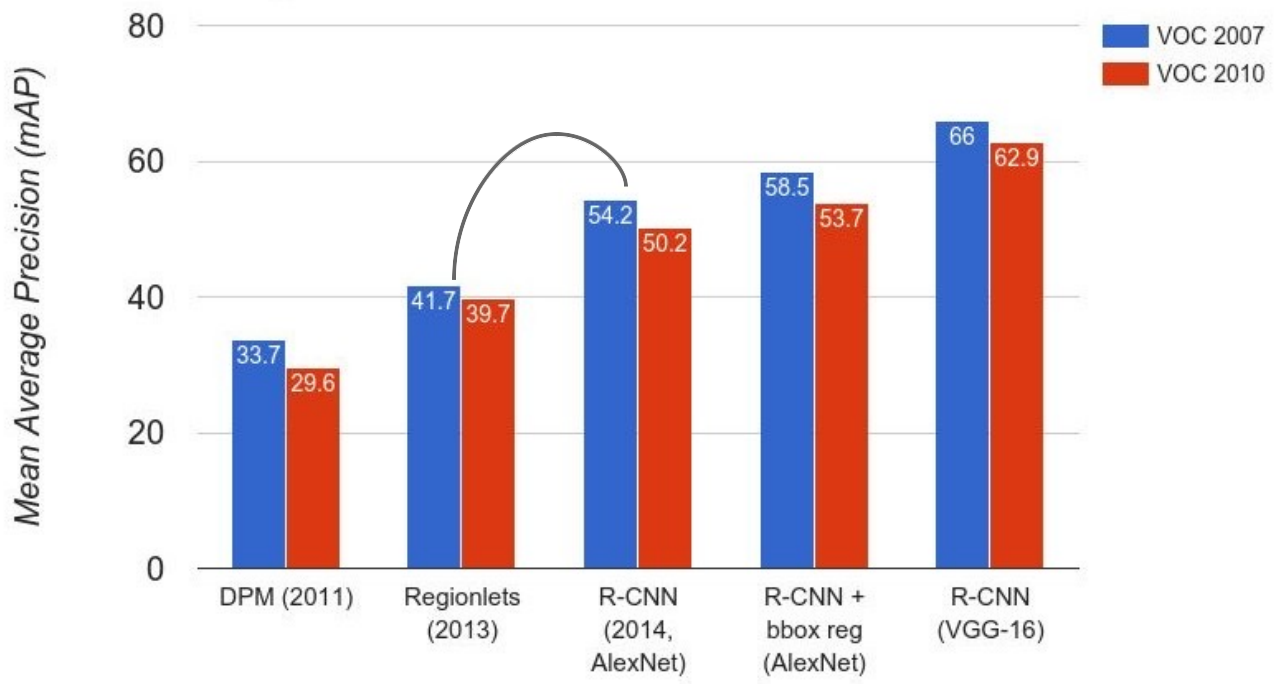
R-CNN Training

- Step 5: bounding-box regression:
  - For each class, train a linear regression model to map from features to offsets to ground-truth bounding boxes → makes up for ”slightly wrong” proposals



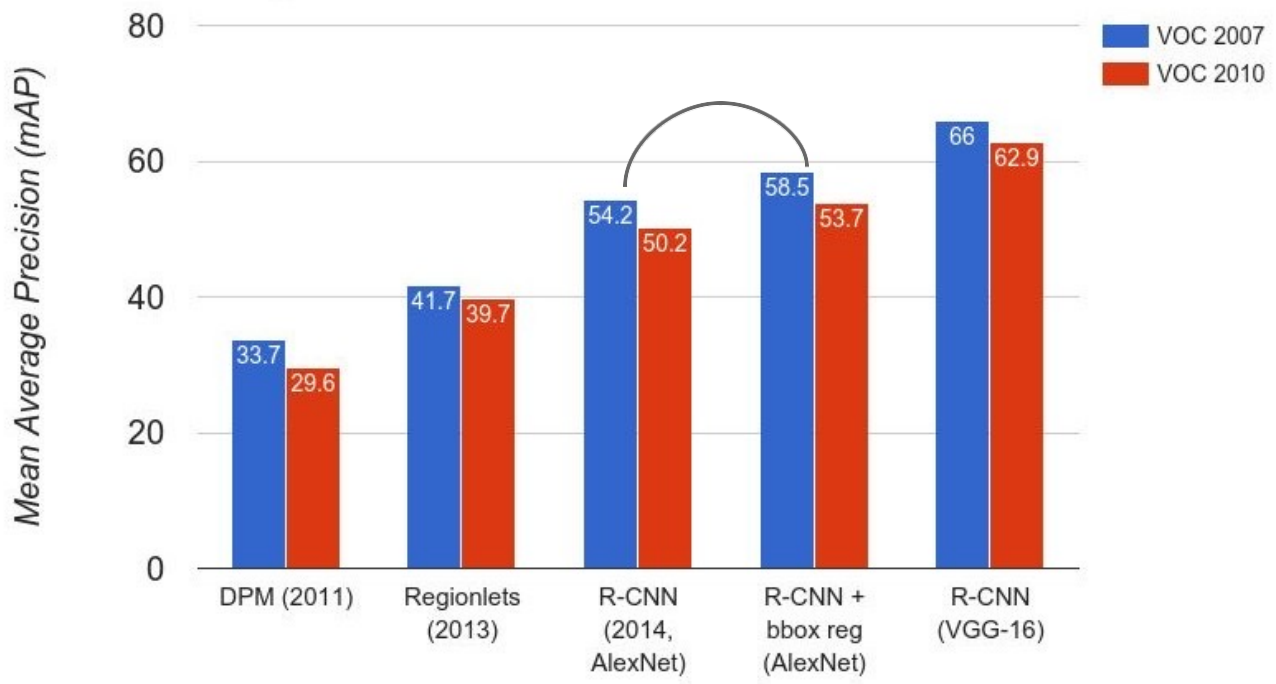
Slides modified from Ross Girshick tutorial at CVPR 2019

R-CNN Results



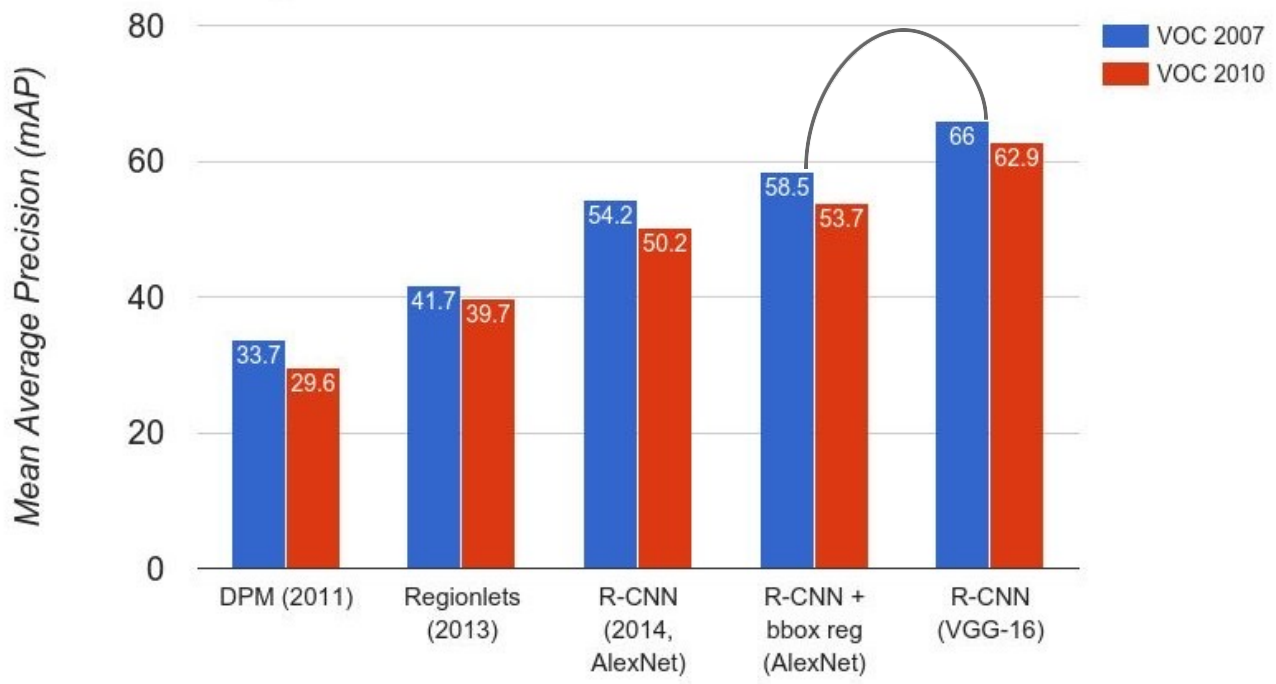
Big improvement compared to pre-CNN methods

R-CNN Results



Big improvement compared to pre-CNN methods
Bounding-box regression helps

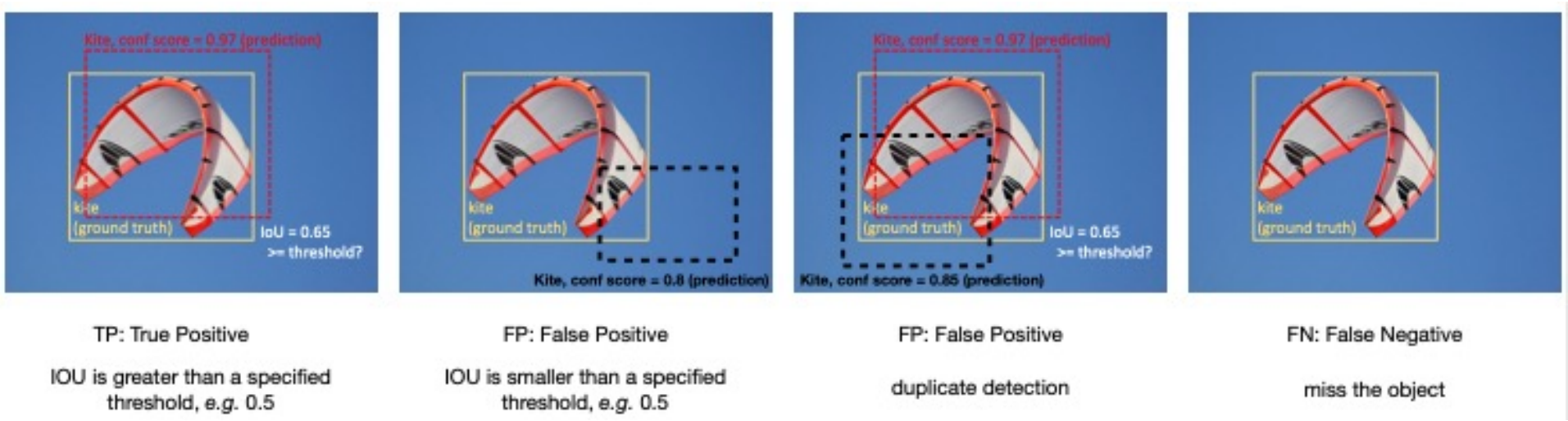
R-CNN Results



Big improvement compared to pre-CNN methods
Bounding-box regression helps
Features from deeper network help

Object Detection Terminology

- The model's prediction is bounding boxes with category confidence scores, *e.g.* person, dog, background, *etc*
- **IoU**: intersection over union between a pair of boxes, *aka.* **Jaccard Similarity**
- **Goal**: large IOU, high confidence score for the correct category

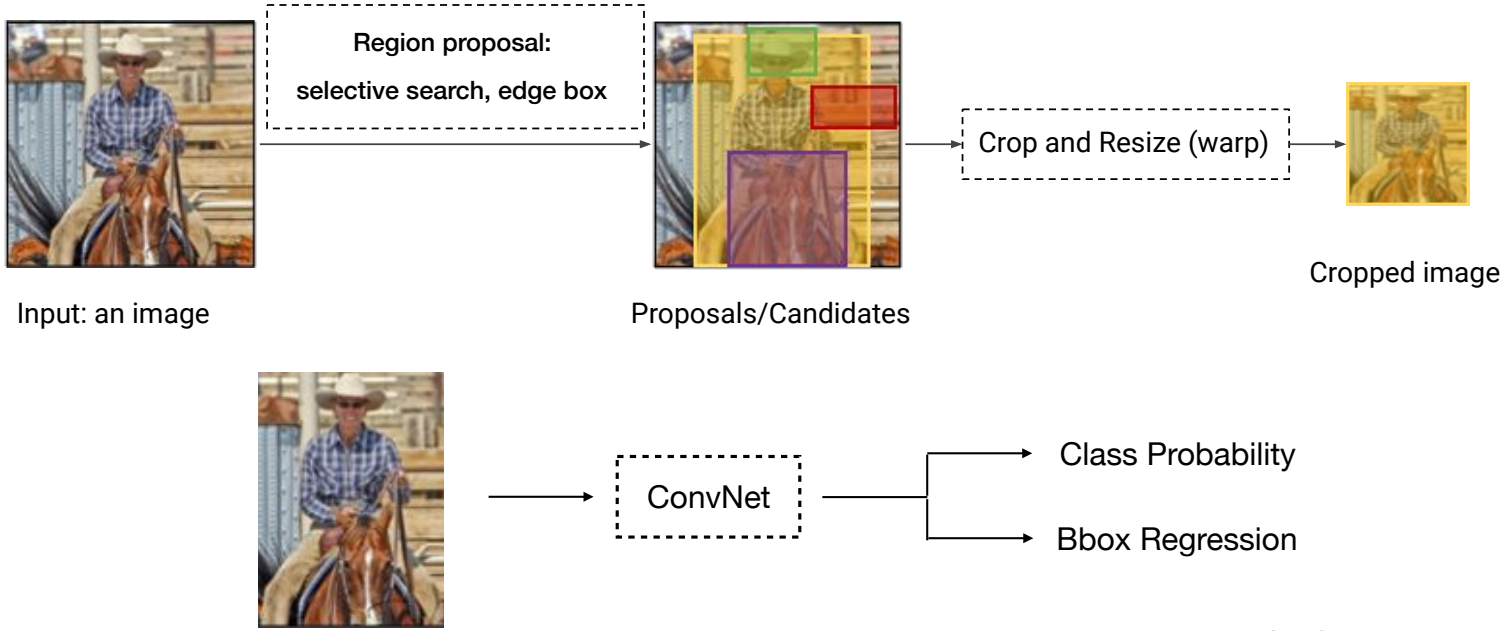


Today
Object detection

- Various Problem Formulations
- General Strategies for Object Detection
  - Single Object Localization
  - Detection as Regression
  - Detection as Classification → R-CNN
- **Fast R-CNN, Faster R-CNN**
- Comparing Boxes
- Evaluating Object Detectors

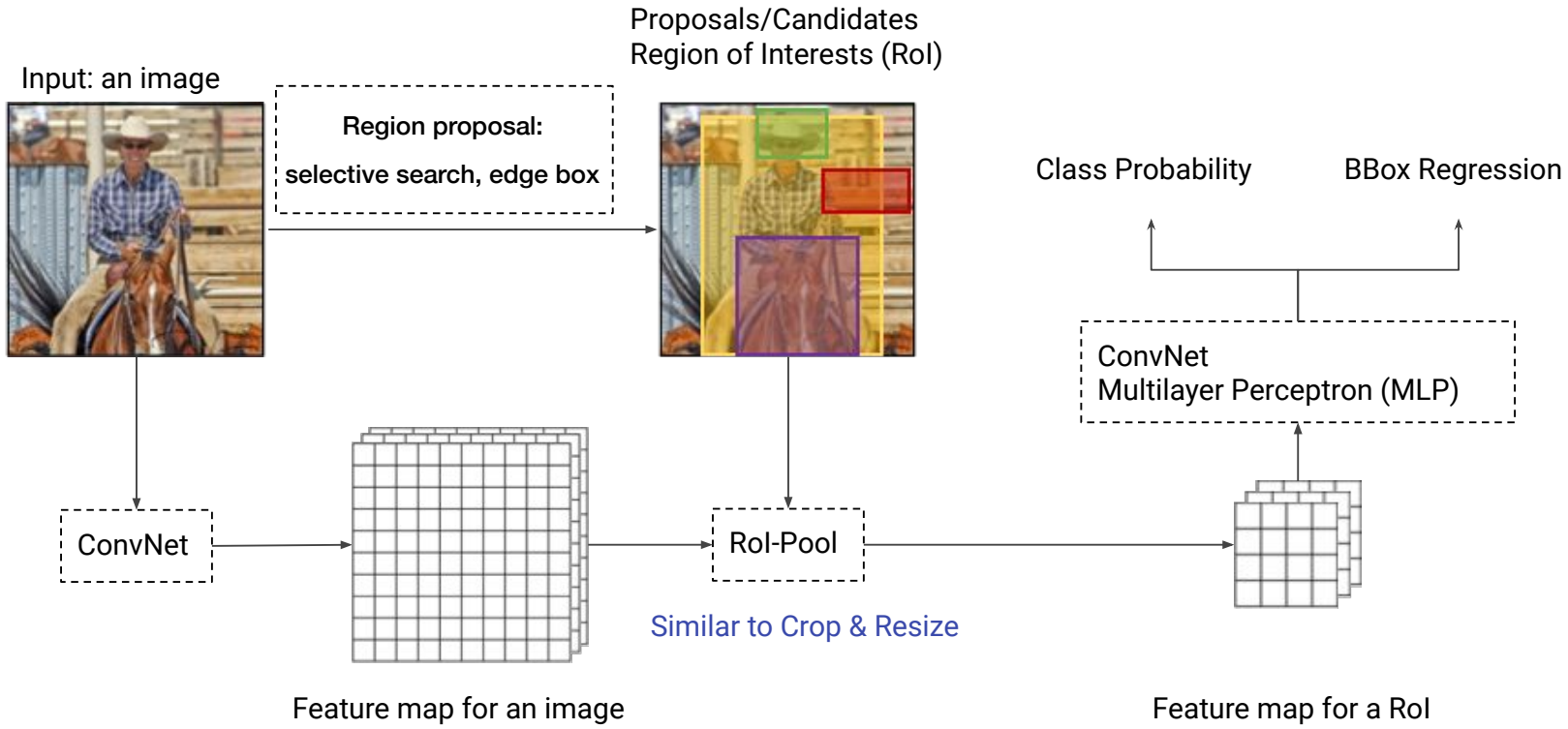
R-CNN: Region-based CNN

- Propose large number of regions potentially with objects and classify each proposed region
- Problem: Computationally expensive!
  - Need to train with all region-proposals



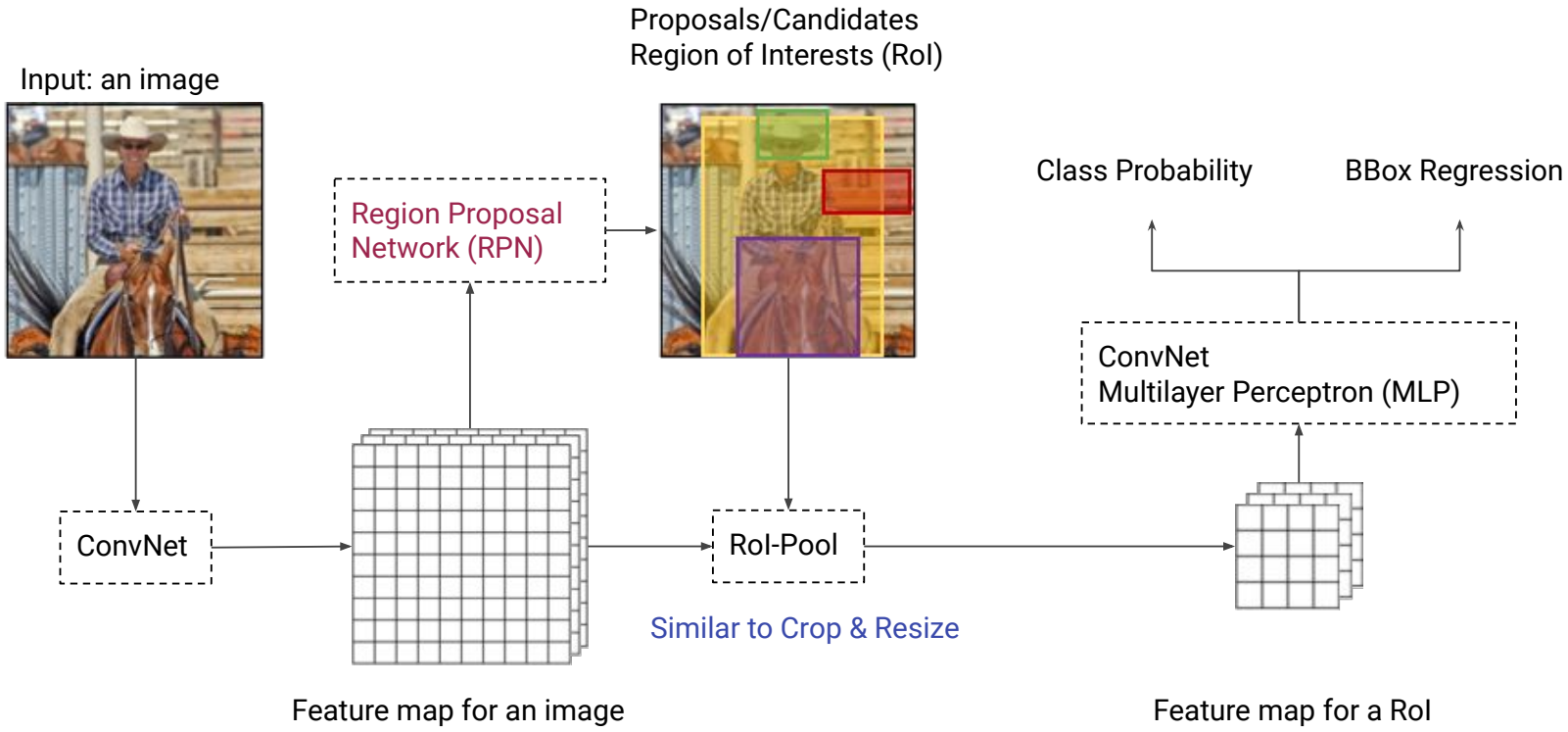
Slides modified from Ross Girshick tutorial at CVPR 2019

Fast R-CNN



Slides modified from Ross Girshick tutorial at CVPR 2019

Faster R-CNN



Slides modified from Ross Girshick tutorial at CVPR 2019

Anchor-based Object Detection

- Common characteristic among two-stage detectors
  → **anchors**

- **Anchor** = bounding box region with specific
  - pre-defined scale and
  - dimensions (width/height)

Faster R-CNN

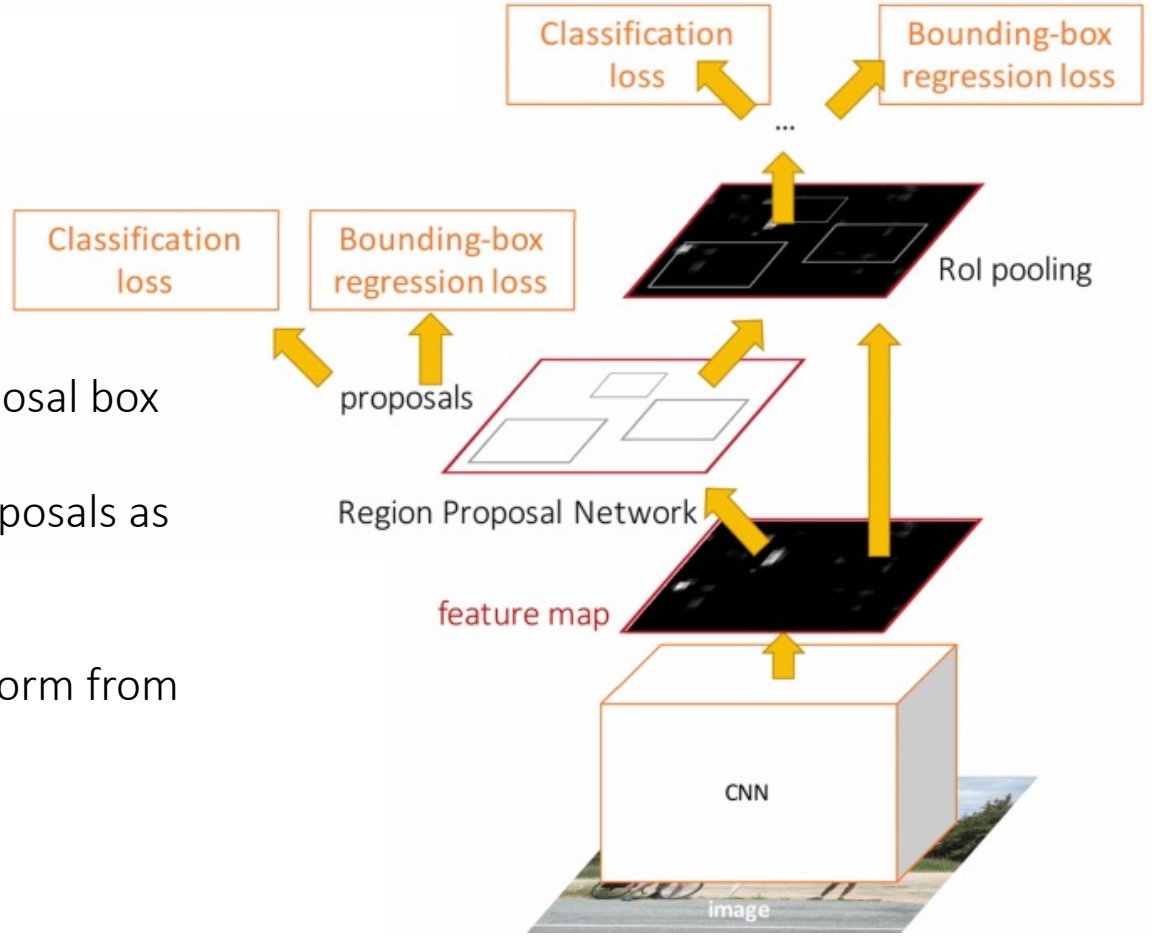
4 losses!

1. RPN classification: anchor box is object/not 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



Faster R-CNN

4 losses!

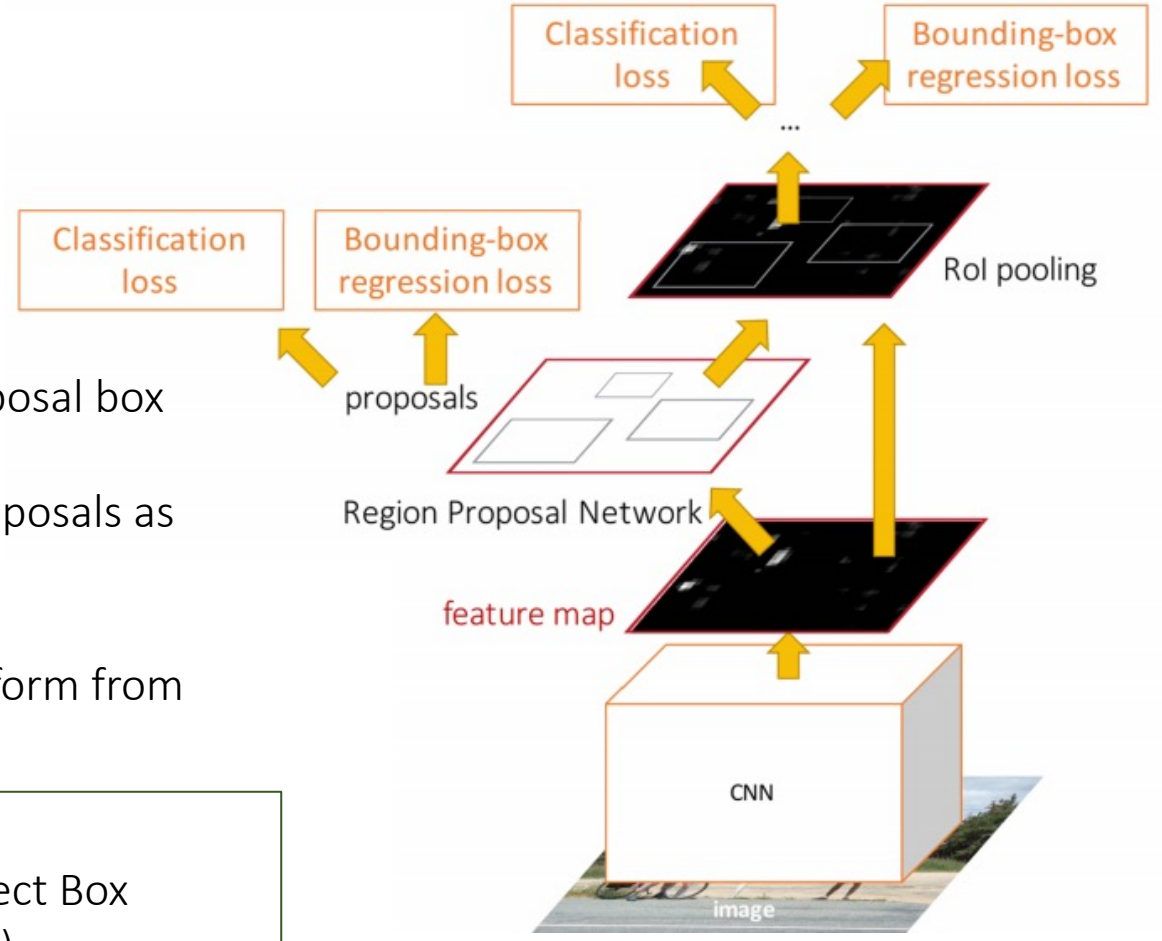
1. RPN classification: anchor box is object/not object

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

2 Stage Training:
Anchor → Region Proposal → Object Box
(Stage 1) (Stage 2)



Architectures

- **Two-stage architectures**
  - Propose large number of regions with **high recall**, meaning all potential objects have been included
  - Classify the regions as object category or background
  - Possibly slow because of the two steps
  - Examples: RCNN, Fast-RCNN, Faster-RCNN, Mask-RCNN, …

Architectures

- **Two-stage architectures**
  - Propose large number of regions with **high recall**, meaning all potential objects have been included
  - Classify the regions as object category or background
  - Possibly slow because of the two steps
  - Examples: RCNN, Fast-RCNN, Faster-RCNN, Mask-RCNN, …

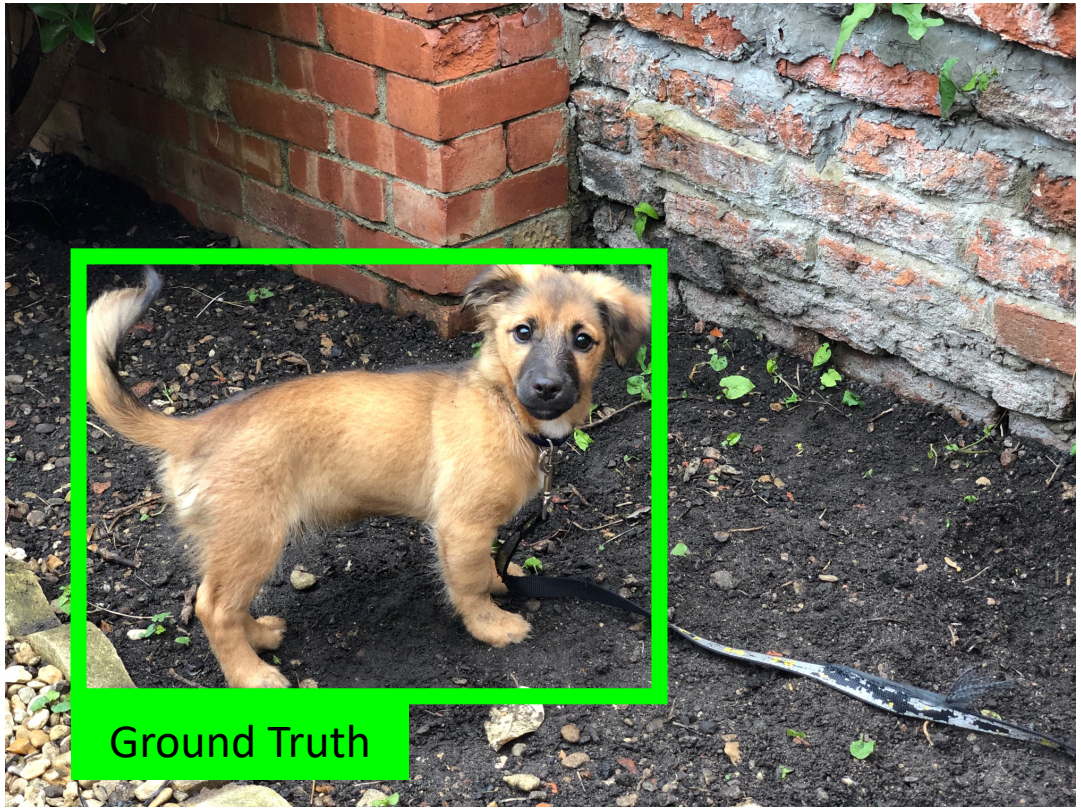
- **One-stage**
  - Regions are built into the architecture (fully convolutional layers)
  - Can be fast
  - Examples:
    - anchor based, *e.g.* YOLO, SSD, RetinaNet, EfficientDet (CVPR2020)
    - point based, *e.g.* CornerNet, CenterNet, FCOS

Today
Object detection

- Various Problem Formulations
- General Strategies for Object Detection
  - Single Object Localization
  - Detection as Regression
  - Detection as Classification → R-CNN
- Fast R-CNN, Faster R-CNN
- **Comparing Boxes**
- Evaluating Object Detectors

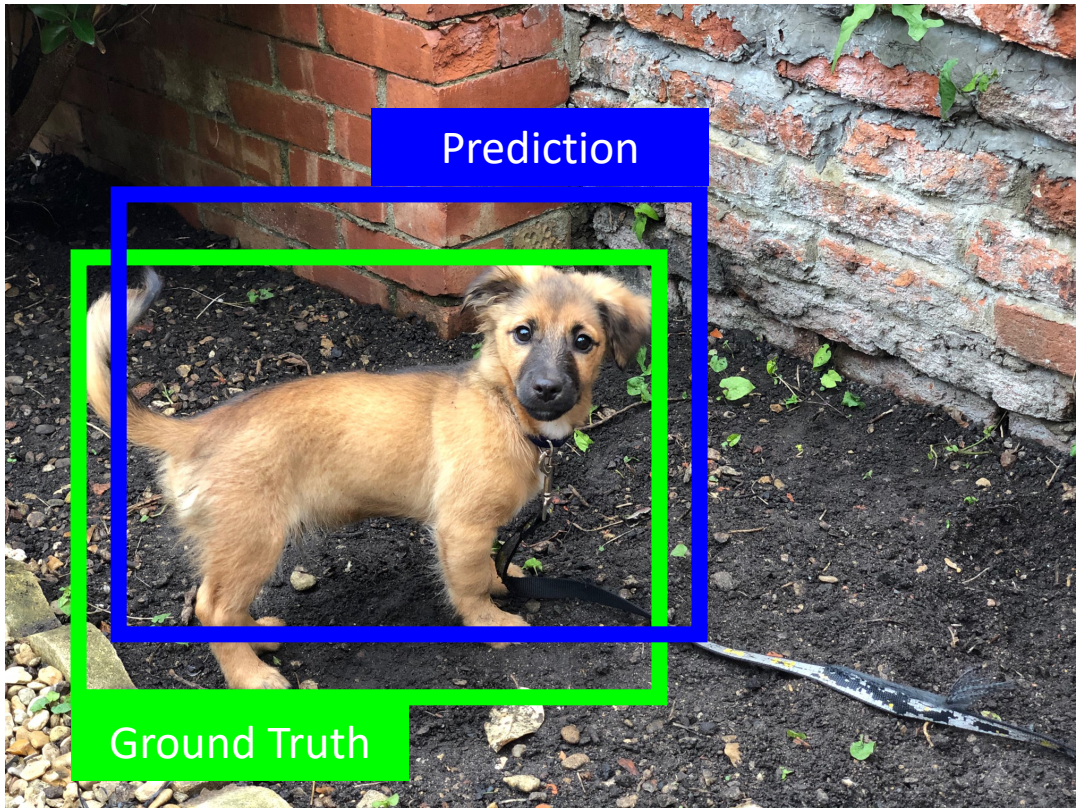
Comparing Boxes

Comparing Boxes



Ground Truth

Comparing Boxes



Ground Truth

Prediction

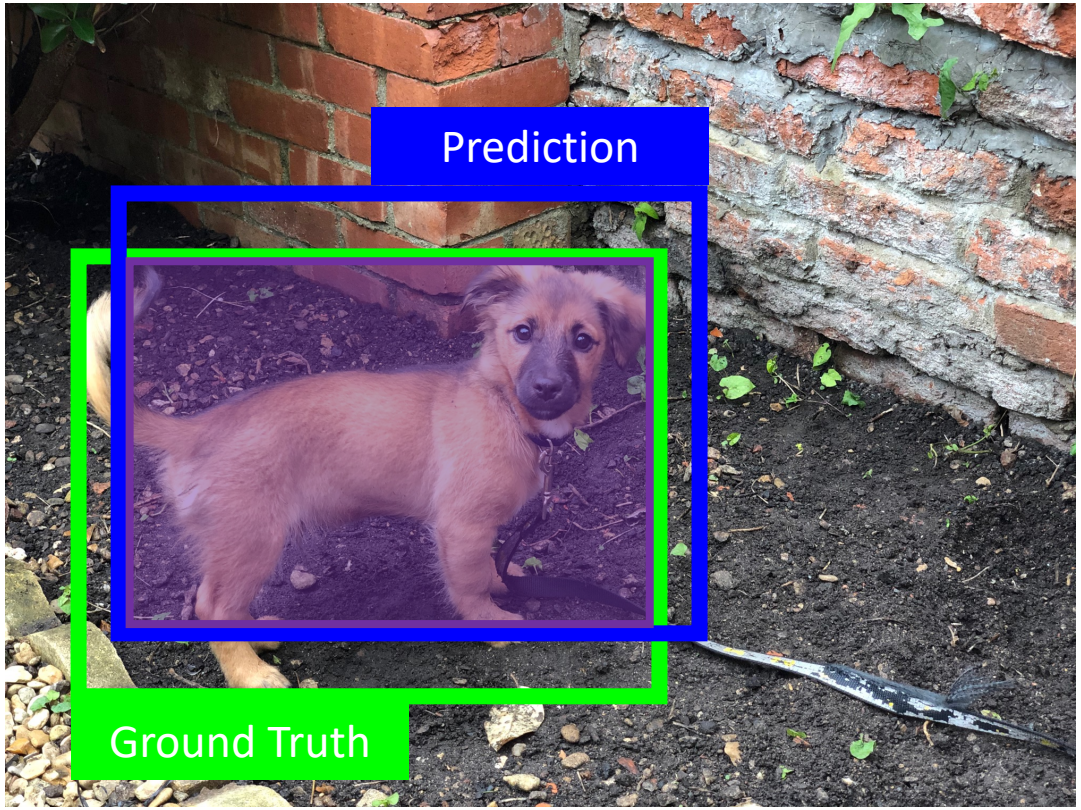
Comparing Boxes

- IoU: **Intersection over Union** (also called “Jaccard Similarity)

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

Comparing Boxes

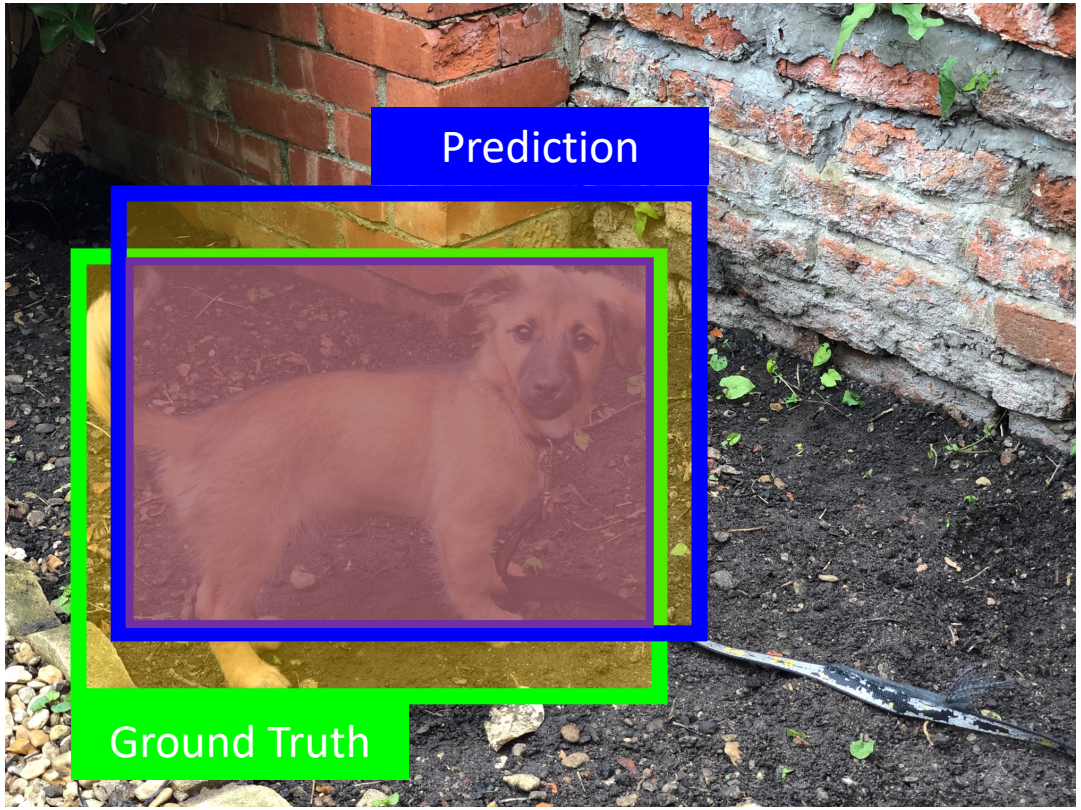
- IoU: **Intersection over Union** (also called "Jaccard Similarity)



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

Comparing Boxes

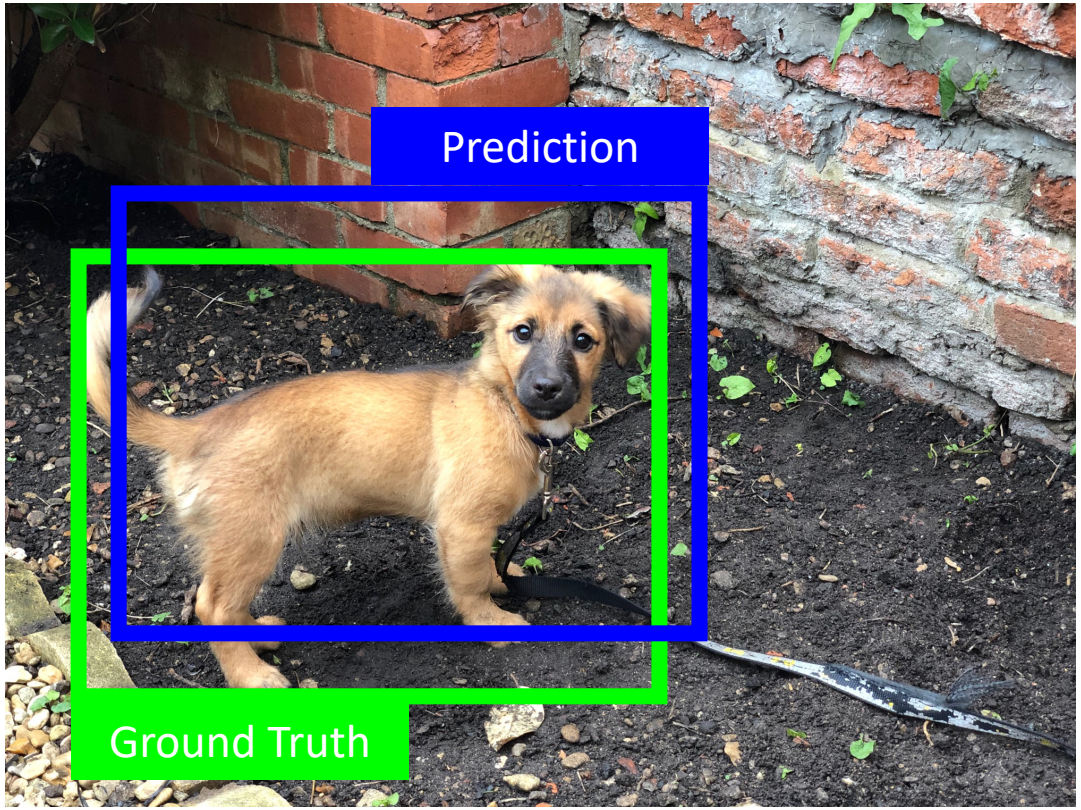
- IoU: **Intersection over Union** (also called "Jaccard Similarity)



$$IoU = \frac{Area\ of\ Intersection.}{Area\ of\ Union.} = \frac{\square \cap \square}{\square \cup \square}$$

Comparing Boxes

- IoU: **Intersection over Union** (also called “Jaccard Similarity)

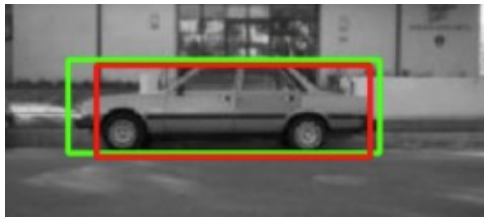


$$\text{IoU} = \frac{\textit{Area of Intersection}}{\textit{Area of Union}}$$

IoU >= 0.5 → correct prediction
IoU < 0.5 → wrong prediction

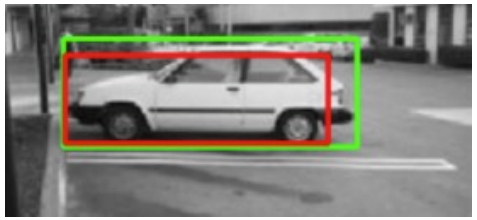
Pop quiz

Ground Truth
Prediction


1


2


3

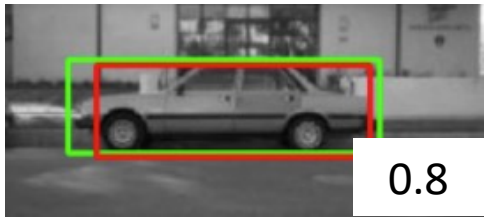

4

- What is the IoU?

(a) 0.4 (b) 0.5 (c) 0.6 (d) 0.7 (e) 0.8 (f) 0.9

Pop quiz Answers

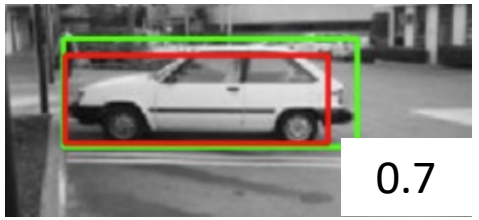
Ground Truth
Prediction



1

2

3



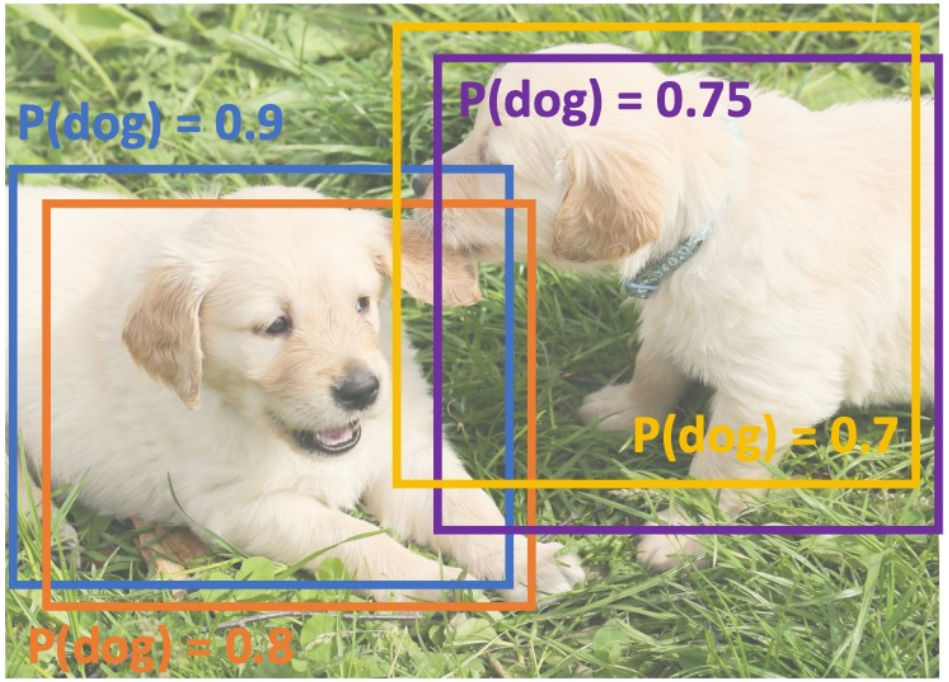
4

- What is the IoU?

(a) 0.5 (b) 0.6 (c) 0.7 (d) 0.8 (e) 0.9

Overlapping Boxes

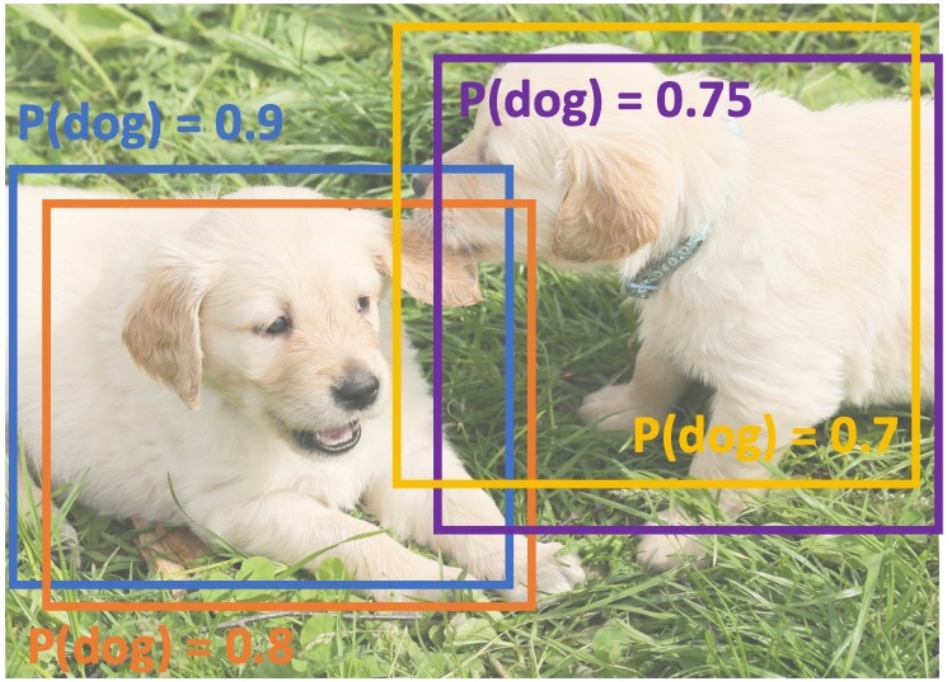
- **NMS:** Non-Max Suppression



When many overlapping predicted bounding-boxes, which one do we keep?

Overlapping Boxes

- **NMS:** Non-Max Suppression



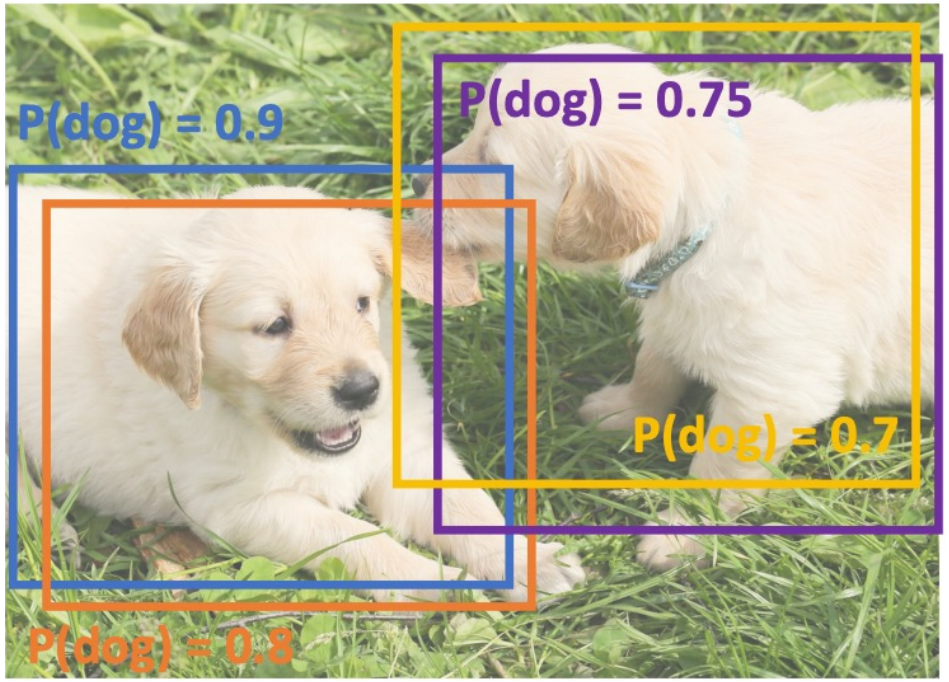
When many overlapping predicted bounding-boxes, which one do we keep?

Post-process raw detections using NMS:

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold
3. If any boxes remain, repeat

Overlapping Boxes

- **NMS:** Non-Max Suppression



When many overlapping predicted bounding-boxes, which one do we keep?

Post-process raw detections using NMS:

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold
3. If any boxes remain, repeat

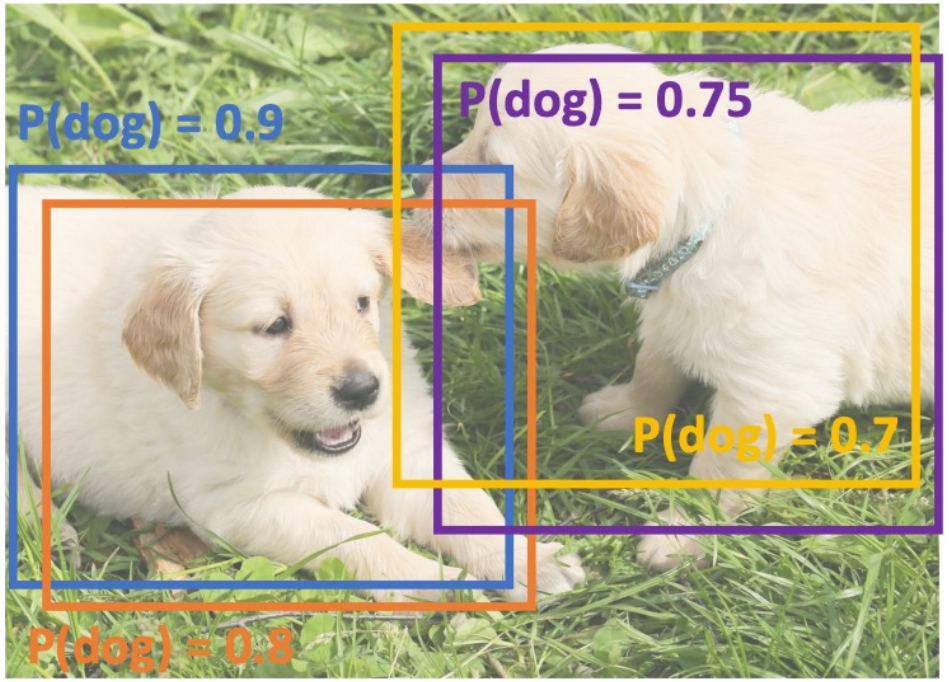
IoU(,) = 0.78

IoU(,) = 0.05

IoU(,) = 0.07

Overlapping Boxes

- **NMS:** Non-Max Suppression



When many overlapping predicted bounding-boxes, which one do we keep?

Post-process raw detections using NMS:

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold
3. If any boxes remain, repeat

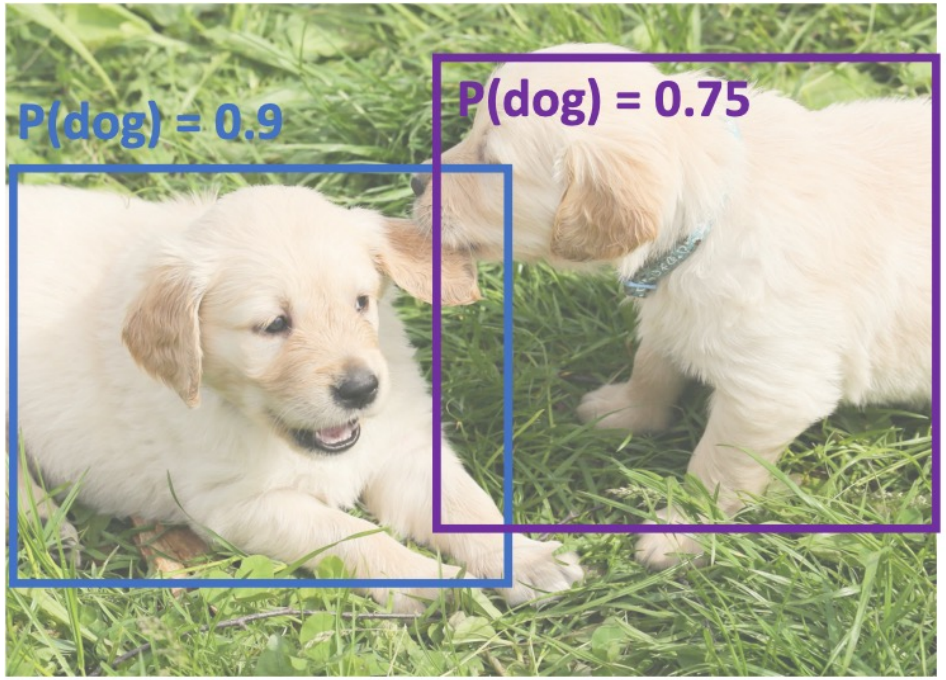
IoU(,) = **0.74**

IoU(,) = 0.05

IoU(,) = 0.03

Overlapping Boxes

- **NMS:** Non-Max Suppression



When many overlapping predicted bounding-boxes, which one do we keep?

Post-process raw detections using NMS:

1. Select next highest-scoring box
2. Eliminate lower-scoring boxes with IoU > threshold
3. If any boxes remain, repeat

Pop quiz

When can NMS fail?

Pop quiz Answers

When can NMS fail?

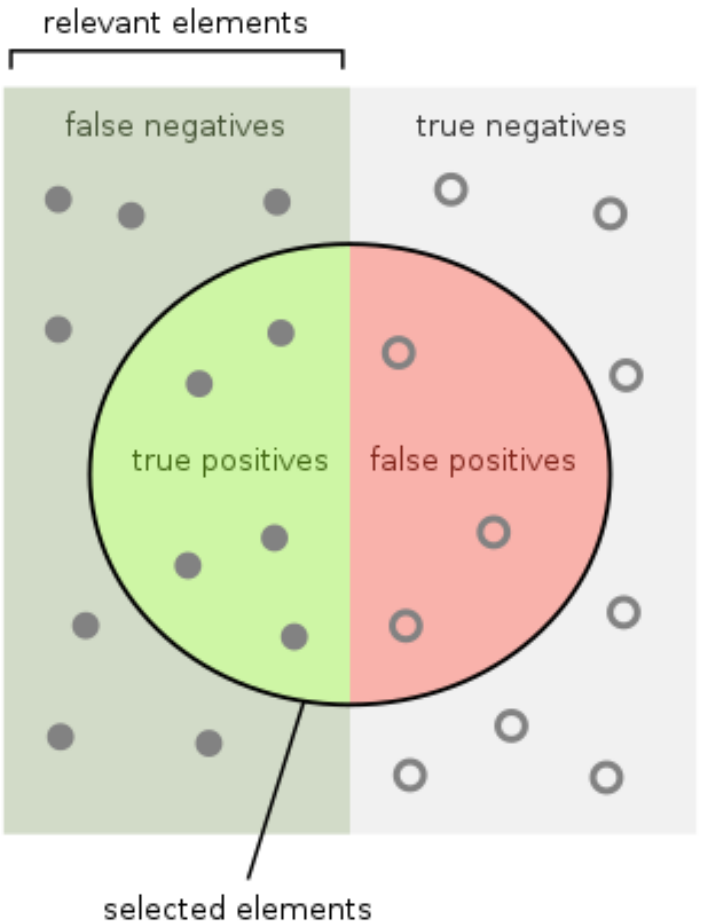
Cluttered images with highly overlapping boxes

Today
Object detection

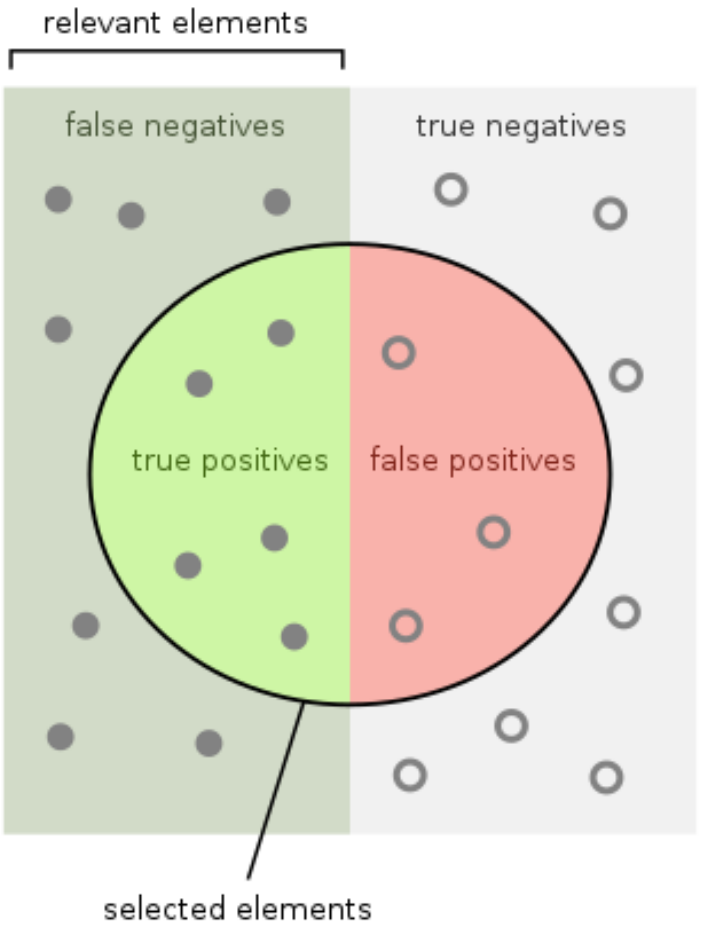
- Various Problem Formulations
- General Strategies for Object Detection
  - Single Object Localization
  - Detection as Regression
  - Detection as Classification → R-CNN
- Fast R-CNN, Faster R-CNN
- Comparing Boxes
- **Evaluating Object Detectors**

[Slides modified from Ecole Polytechnique, Computer Vision in Deep Learning]

Recall & Precision



Recall & Precision



How many selected items are relevant?

Precision = 

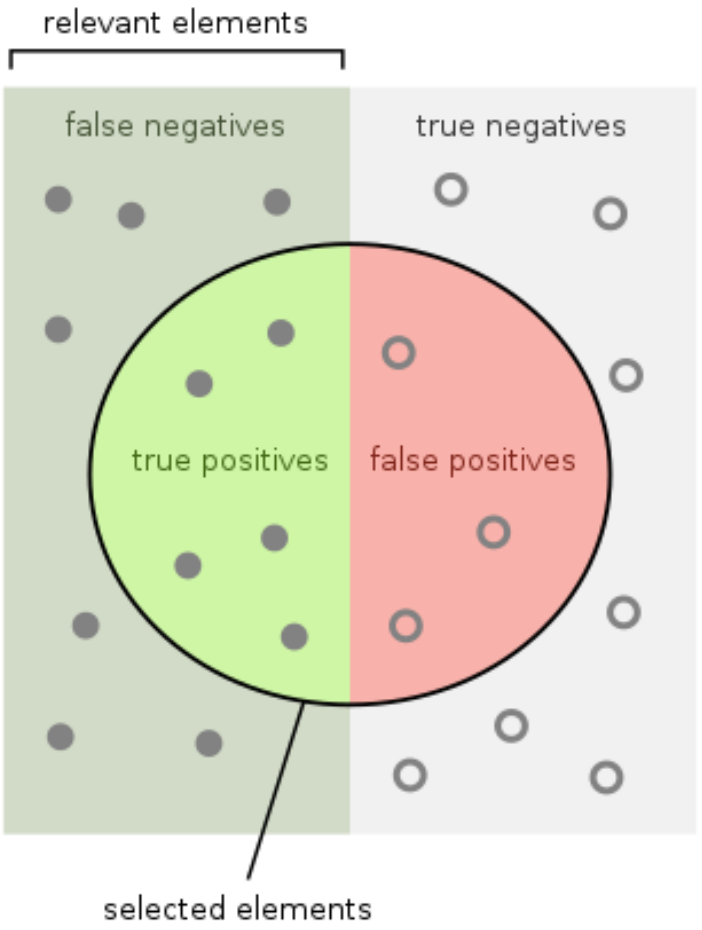
How many relevant items are selected?

Recall = 

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Recall & Precision



How many selected items are relevant?

Precision = 

How many relevant items are selected?

Recall = 

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Missed Detections (MD)

Evaluating Object Detectors

- **mAP:** Mean Average Precision
  - Run Object Detector on all test images
  - NMS on all predicted boxes
  - For each class, compute Average Precision (AP) = area under the Precision-Recall Curve, as follows:

Evaluating Object Detectors

- **mAP:** Mean Average Precision
  - Run Object Detector on all test images
  - NMS on all predicted boxes
  - For each class, compute Average Precision (AP) = area under the Precision-Recall Curve, as follows:
    - Sort detections in decreasing order of confidence scores
    - For each detection in the sorted list:
      - If it matches a ground-truth (GT) box with IoU>0.5, mark it as *positive* and *remove* the GT.
      - If not, mark it as *negative.*
      - Plot a point on PR Curve.

mAP Example

5 boxes with scores

mAP Example

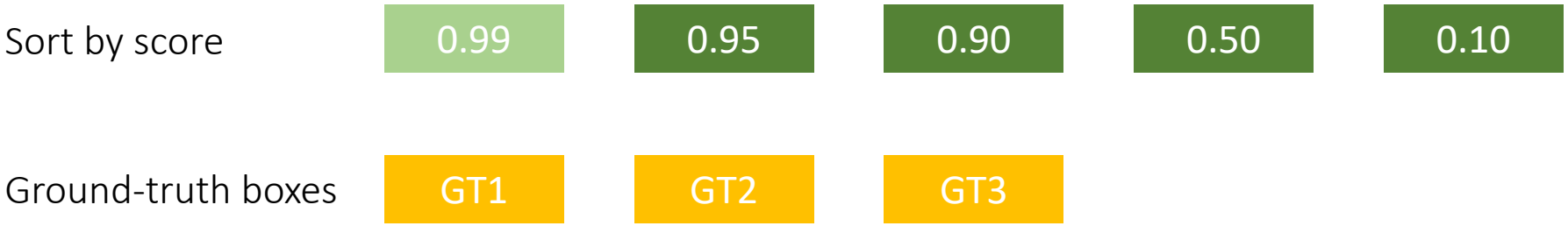
5 boxes with scores | 0.99 | 0.50 | 0.10 | 0.90 | 0.95 |

Sort by score | 0.99 | 0.95 | 0.90 | 0.50 | 0.10 |

mAP Example

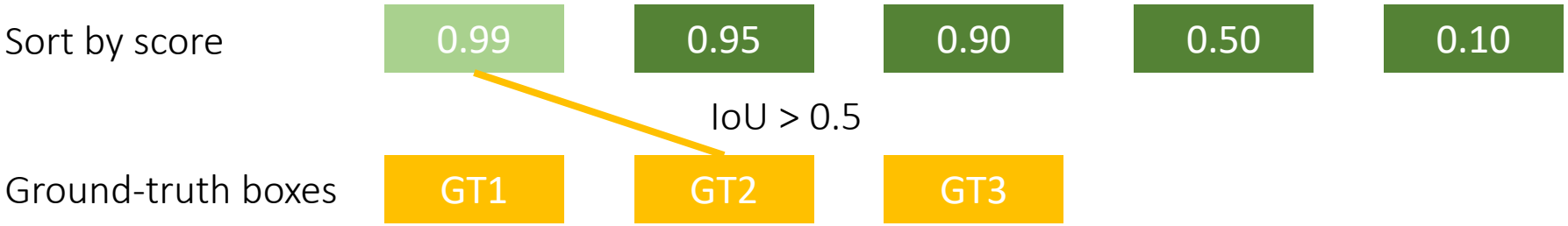

5 boxes with scores
0.99
0.50
0.10
0.90
0.95
Sort by score
0.99
0.95
0.90
0.50
0.10
Ground-truth boxes
GT1
GT2
GT3

mAP Example



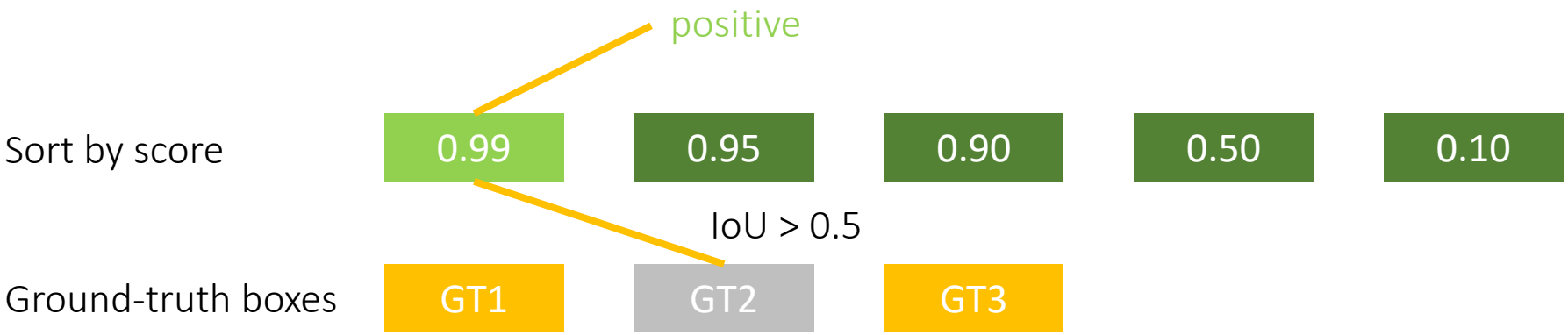
For each detection, does it match a GT with IoU > 0.5?

mAP Example



For each detection, does it match a GT with IoU > 0.5?

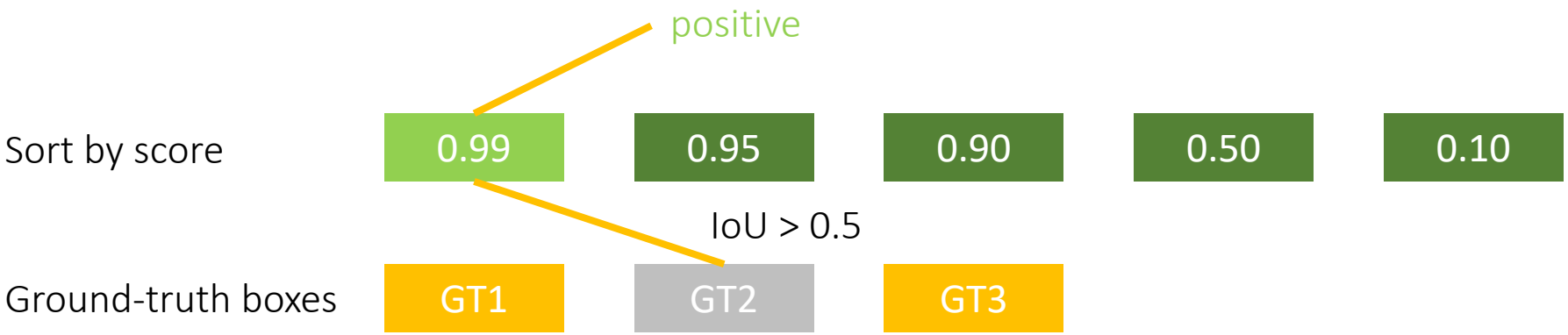
mAP Example



For each detection, does it match a GT with IoU > 0.5?

- Mark as positive
- Remove GT2

Pop quiz

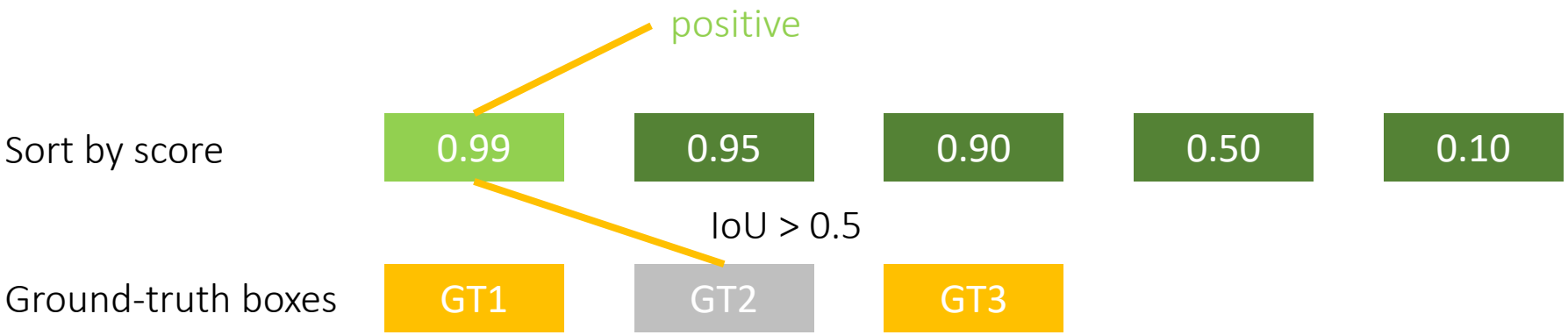


For each detection, does it match a GT with IoU > 0.5?

- Mark as positive
- Remove GT2

1. What is the Precision?
2. What is the Recall?

Pop quiz - hint



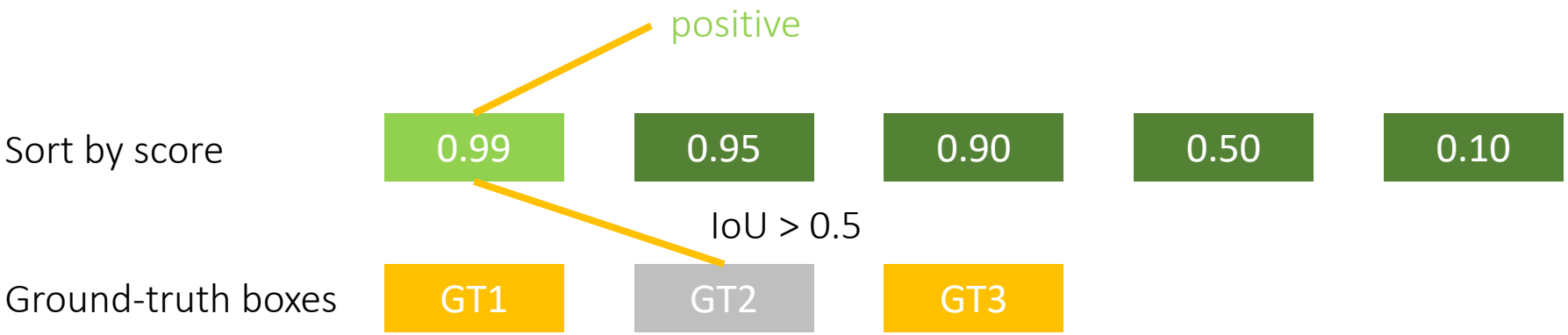
For each detection, does it match a GT with IoU > 0.5?

- Mark as positive
- Remove GT2

1. What is the Precision?
2. What is the Recall?

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

Pop quiz Answers



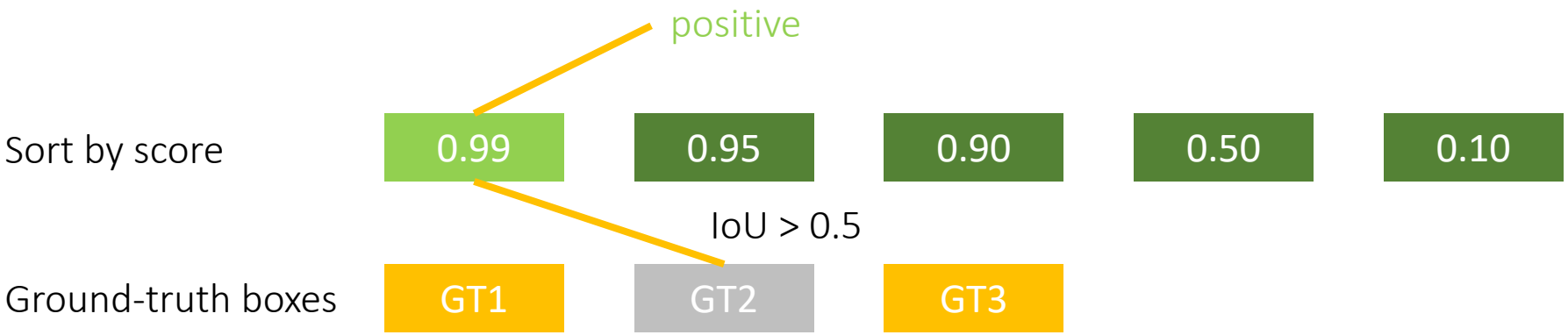
For each detection, does it match a GT with IoU > 0.5?

- Mark as positive
- Remove GT2

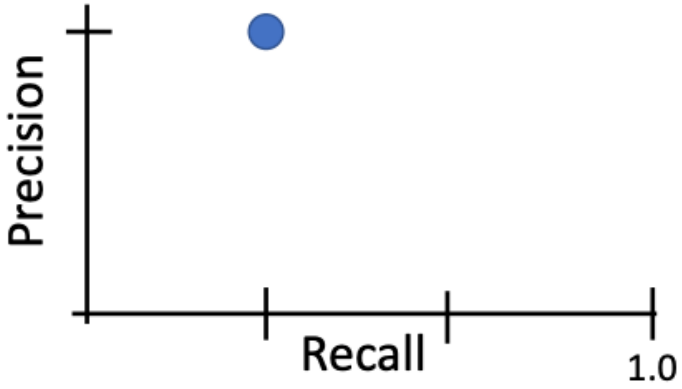
1. What is the Precision?
2. What is the Recall?

- Precision = 1/1 = 1
- Recall = 1/3 = 0.33

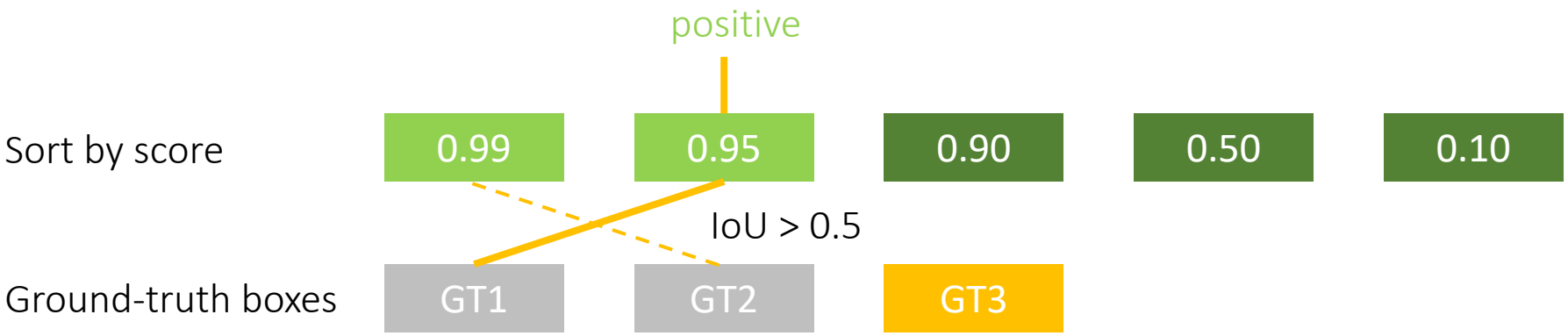
mAP Example



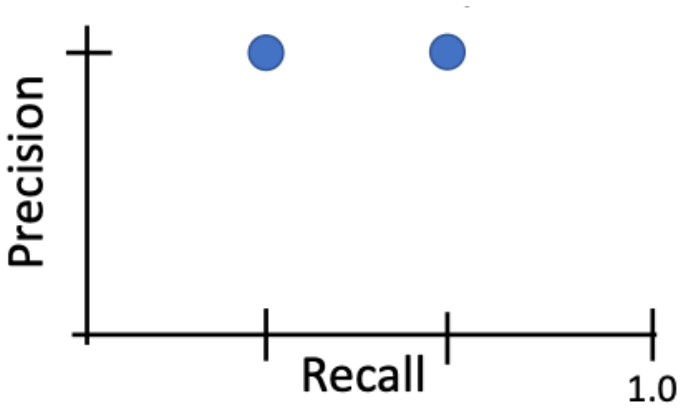
- Precision = 1/1 = 1
- Recall = 1/3 = 0.33



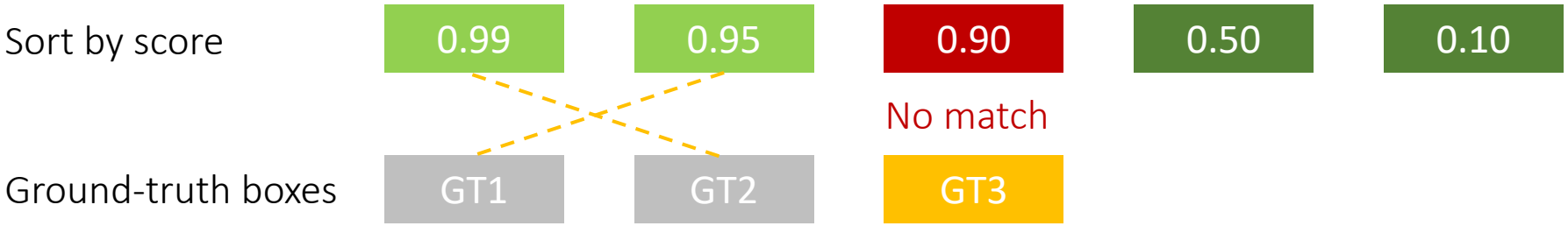
mAP Example



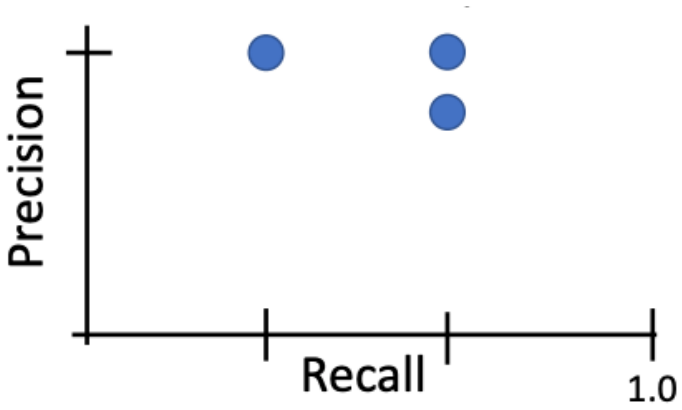
- Precision = 2/2 = 1
- Recall = 2/3 = 0.67



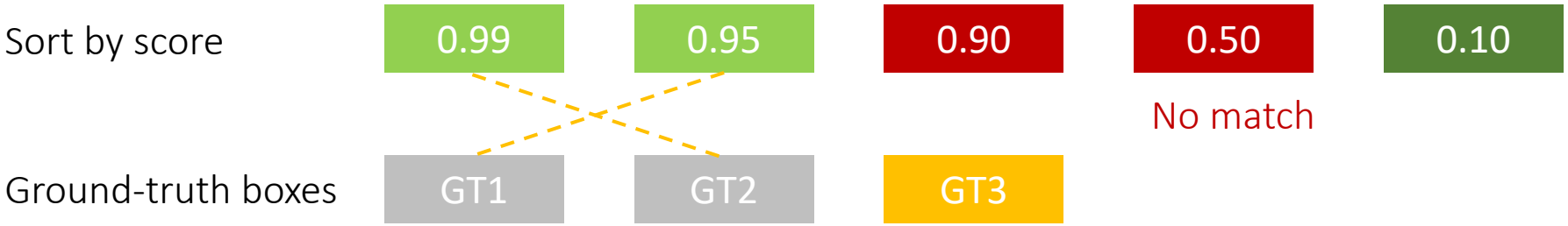
mAP Example



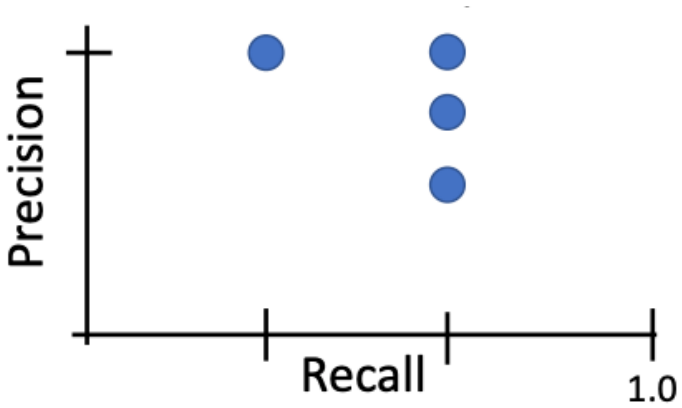
- Precision = 2/3 = 0.67
- Recall = 2/3 = 0.67



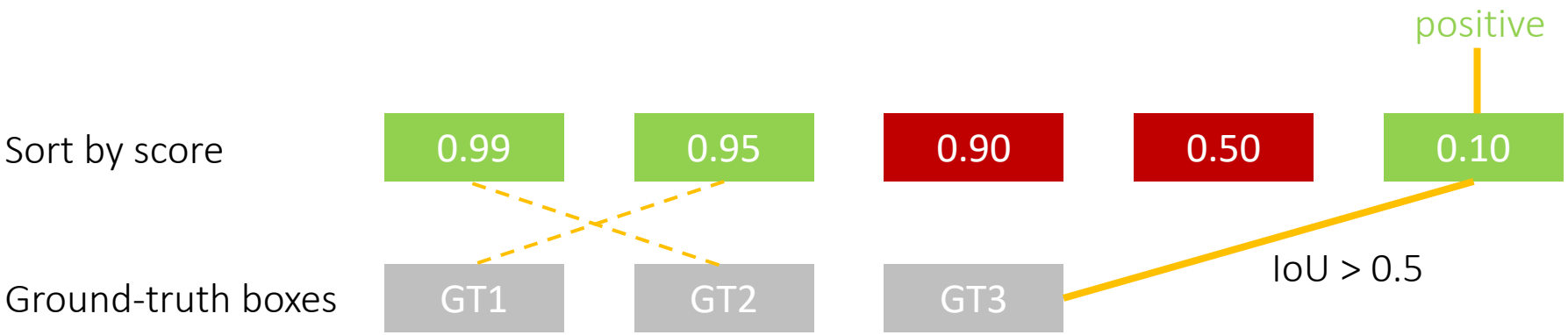
mAP Example



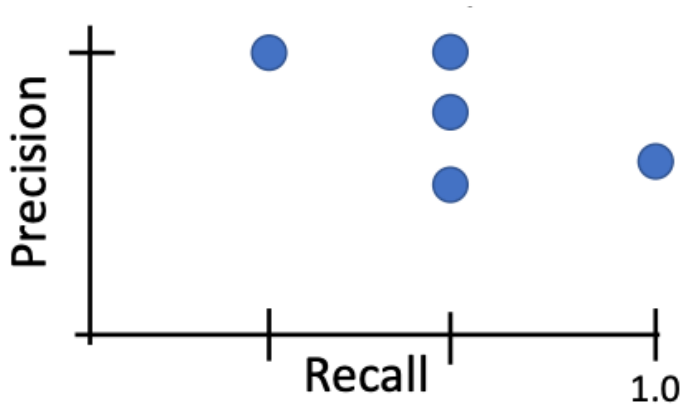
- Precision = 2/4 = 0.5
- Recall = 2/3 = 0.67



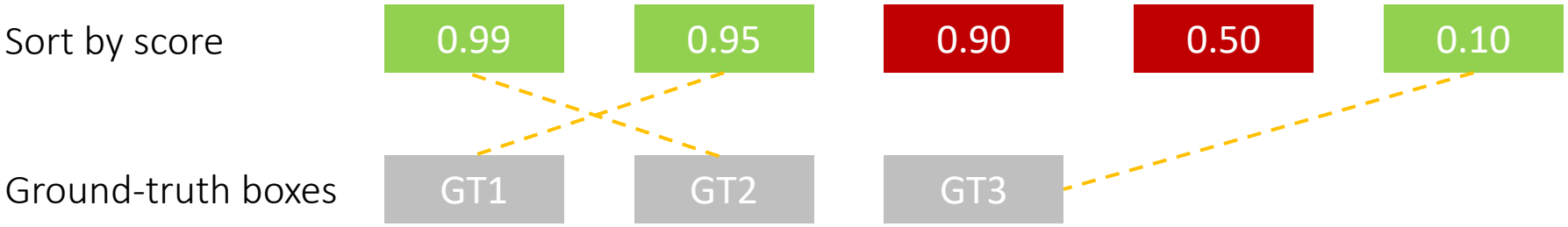
mAP Example

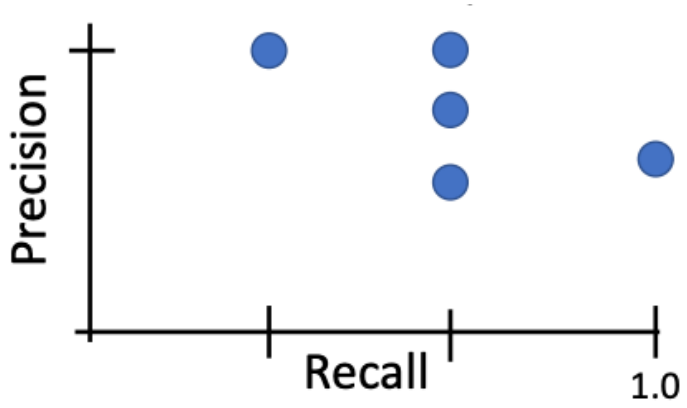


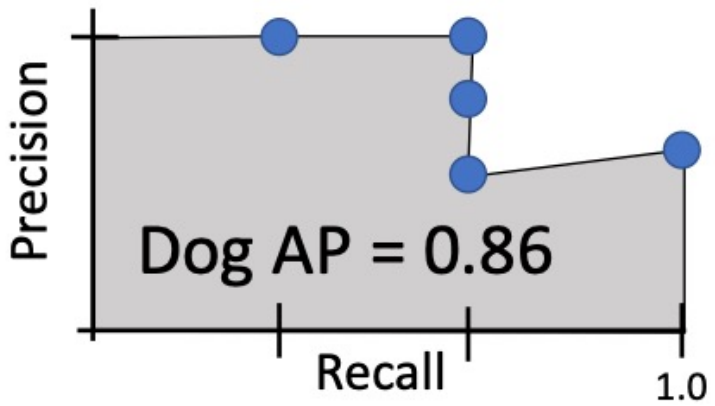
- Precision = 3/5 = 0.6
- Recall = 3/3 = 1.0



mAP Example


Sort by score
0.99
0.95
0.90
0.50
0.10
Ground-truth boxes
GT1
GT2
GT3


Precision
Recall
1.0


Precision
Dog AP = 0.86
Recall
1.0

mAP

Perfect AP = 100%

→

Hit all GT boxes with IoU >0.5 with
NO false positive (FP) detections
ranked above
any true positives (TP)

Evaluating Object Detectors

- **mAP:** Mean Average Precision
  - Run Object Detector on all test images
  - NMS on all predicted boxes
  - For each class, compute Average Precision (AP) = area under the Precision-Recall Curve, as follows:
    - Sort detections in decreasing order of confidence scores
    - For each detection in the sorted list:
      - If it matches a ground-truth (GT) box with IoU>0.5, mark it as *positive* and *remove* the GT.
      - If not, mark it as *negative.*
      - Plot a point on PR Curve.
  - **Average over all classes to get "mean"**

Evaluating Object Detectors

- **mAP:** Mean Average Precision
  - Run Object Detector on all test images
  - NMS on all predicted boxes
  - For each class, compute Average Precision (AP) = area under the Precision-Recall Curve, as follows:
    - Sort detections in decreasing order of confidence scores
    - For each detection in the sorted list:
      - If it matches a ground-truth (GT) box with IoU>0.5, mark it as *positive* and *remove* the GT.
      - If not, mark it as *negative.*
      - Plot a point on PR Curve.
  - **Average over all classes to get "mean"**

Dog AP = 0.86
Cat AP = 0.80
Car AP = 0.65
mAP@0.5 = 0.77

Questions??

Project 4

Detecting waste in the wild

Project 4

Detecting waste in the wild

Tasks for simple object detector:

- Extract object proposals
- Finetune a CNN for object detector on object proposals (replace last layer)
- Apply the model onE test images
- Implement NMS
- Evaluate the object detection performance

Save the environment: Detecting waste in the wild

Project 1.2

Deep Learning in Computer Vision

June 2022

Litter has been accumulating around us as most local governments and international organizations fail to tackle this crisis, which is having a catastrophic impact on biodiversity and marine animals. In this project, you are asked to build a deep learning object detection system that can automatically detect trash and litter and in images in the wild. This object detection can then be deployed in robotic machines that can scan areas and collect and clean beaches, forests and roads.

Figure 1: Examples from the TACO dataset.

Project 4

Detecting waste in the wild

Tasks for simple object detector:

- Extract object proposals
- Finetune a CNN for object detector on object proposals (replace last layer)
- Apply the model on test images
- Implement NMS
- Evaluate the object detection performance

OPTIONAL 1: Improve the efficiency of the simple model (i.e., ROI pooling layer)

OPTIONAL 2: Implement an RPN to replace the object proposal algorithm

Save the environment: Detecting waste in the wild

Project 1.2

Deep Learning in Computer Vision

June 2022

Litter has been accumulating around us as most local governments and international organizations fail to tackle this crisis, which is having a catastrophic impact on biodiversity and marine animals. In this project, you are asked to build a deep learning object detection system that can automatically detect trash and litter and in images in the wild. This object detection can then be deployed in robotic machines that can scan areas and collect and clean beaches, forests and roads.

Figure 1: Examples from the TACO dataset.

Feedback ☺

Join at **menti.com** use code **6137 9549**

Thank you!!!