

Computed Vision

Naeemullah Khan

naeemullah.khan@kaust.edu.sa



جامعة الملك عبد الله
للغعلوم والتكنولوجية
King Abdullah University of
Science and Technology

KAUST Academy
King Abdullah University of Science and Technology

February 7, 2025

Table of Contents

1. Object Detection
2. Region Proposal and R-CNN
3. YOLO and Single Shot Detection
4. Instance Segmentation
5. Mask R-CNN
6. Panoptic Segmentation

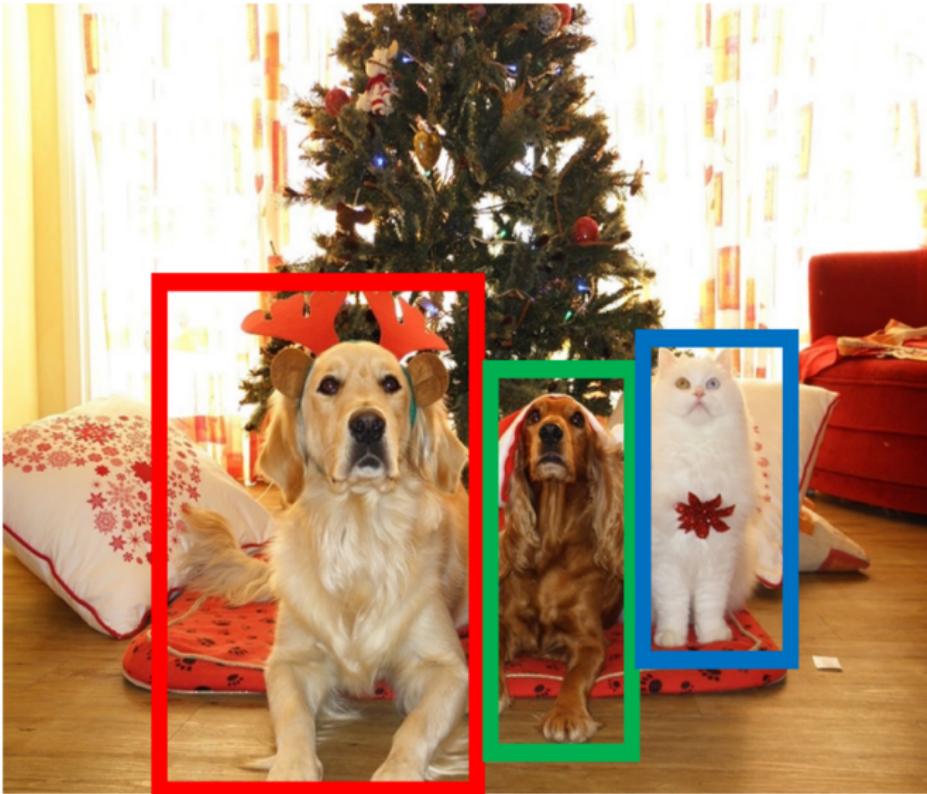
Learning Outcomes

- ▶ Understand the fundamentals of object detection and its challenges.
- ▶ Learn different region proposal techniques such as R-CNN and Faster R-CNN.
- ▶ Explore one-stage object detection approaches like YOLO.
- ▶ Understand the concepts of instance segmentation and Mask R-CNN.
- ▶ Differentiate between semantic, instance and panoptic segmentation.

Object Detection

- ▶ **Input:** Single RGB Image
- ▶ **Output:** A set of detected objects. For each object predict:
 - Category label (from fixed, known set of categories)
 - Bounding box (four numbers: x, y, width, height)

Object Detection (cont.)



Object Detection: Challenges

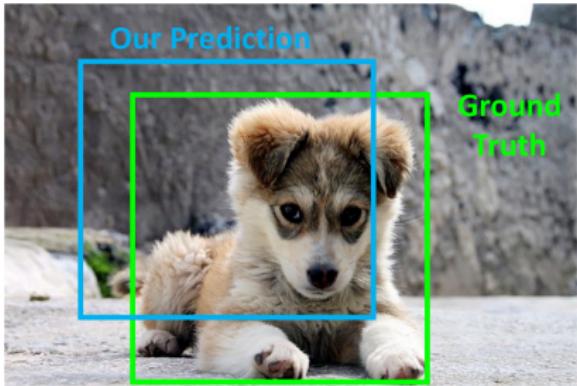
- ▶ **Multiple outputs:** Need to output variable numbers of objects per image
- ▶ **Multiple types of output:** Need to predict "what" (category label) as well as "where" (bounding box)
- ▶ **Large images:** Classification works at 224x224; need higher resolution for detection, often $\sim 800 \times 600$

Comparing Boxes: Intersection over Union (IoU)



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

The correct
selection we
placed in Data

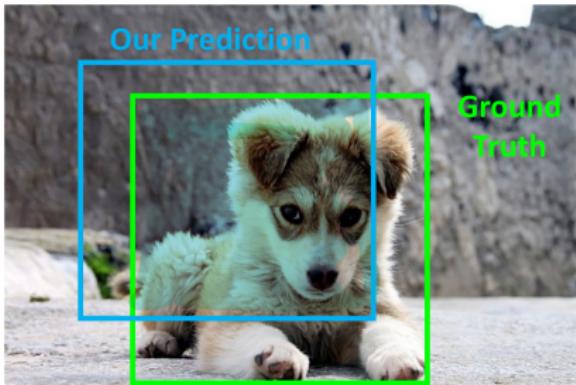


Comparing Boxes: Intersection over Union (IoU)

- ▶ How can we compare our prediction to the ground-truth box?
 - ▶ **Intersection over Union** (IoU)
(Also called "Jaccard similarity" or "Jaccard index"):

performance metric = $\frac{\text{Area of Intersection}}{\text{Area of Union}}$

(our prediction + Ground Truth) - Intersection



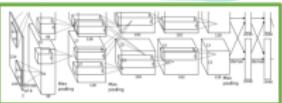
Detecting a Single Object (cont.)

classification +

Bounding
Box
(regression)



Often pretrained
on ImageNet
(Transfer learning)



This image is CC0 public domain

Treat localization as a
regression problem!

Problem: Images can have
more than one object!

Vector:
4096

Fully
Connected:
4096 to 1000

“What”

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

Fully
Connected:
4096 to 4

Box
Coordinates
(x, y, w, h)

one
type
(regression)
Trying to
localization
then

Computed Vision

Correct label:

Cat

Classification

Softmax

Loss

Multitask
Loss

Weighted
Sum

Loss

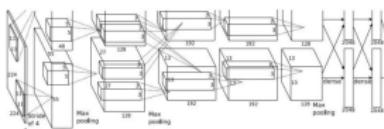
$$L = L_{cls} + \lambda L_{reg}$$

regression
→ L2 Loss
→ MSE Loss (ridge)
(mean Squared Error)
Usually use
L1 (Lasso)

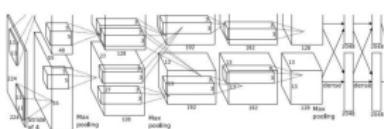
Correct box:
(x', y', w', h')

Lambda is Control parameter
(hyperparameter can be tune)

Multiple Objects



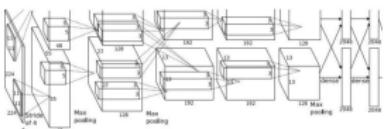
CAT: (x, y, w, h)



DOG: (x, y, w, h)

DOG: (x, y, w, h)

CAT: (x, y, w, h)



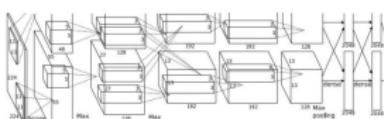
DUCK: (x, y, w, h)

DUCK: (x, y, w, h)

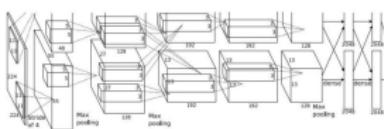
....

Multiple Objects (cont.)

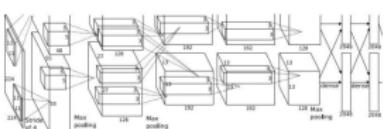
Each image needs a different number of outputs!



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



DOG: (x, y, w, h)
DOG: (x, y, w, h) **12 numbers**
CAT: (x, y, w, h)



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

....

Detecting Multiple Objects: Sliding Window (cont.)



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 $H \times W$?

Consider a box of size $h \times w$:

Possible x positions: $W - w + 1$

Possible y positions: $H - h + 1$

Possible positions:

$$(W - w + 1) * (H - h + 1)$$

Total possible boxes:

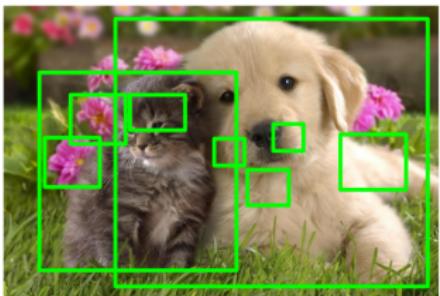
$$\sum_{h=1}^{\max} \sum_{w=1}^{\max} (W - w + 1)(H - h + 1)$$

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

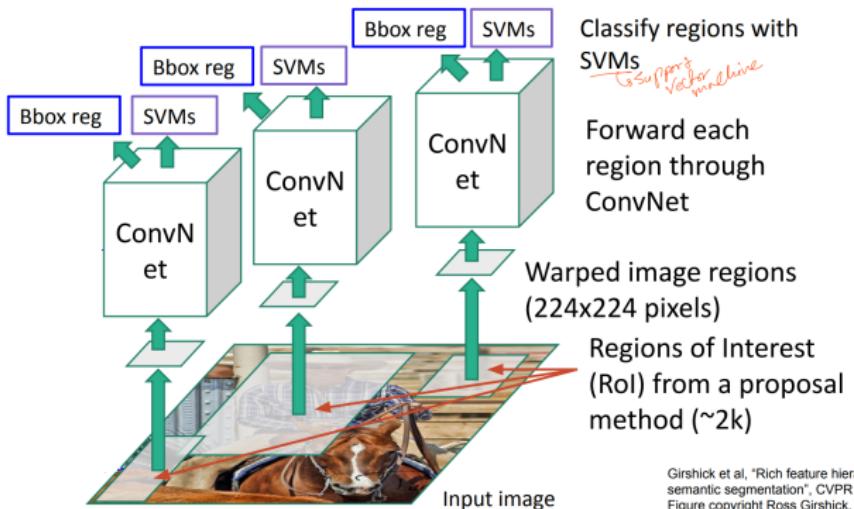
minimum height & width = 1
Maximum $\sim \sim$ = Whole image

Region Proposal

- ▶ 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



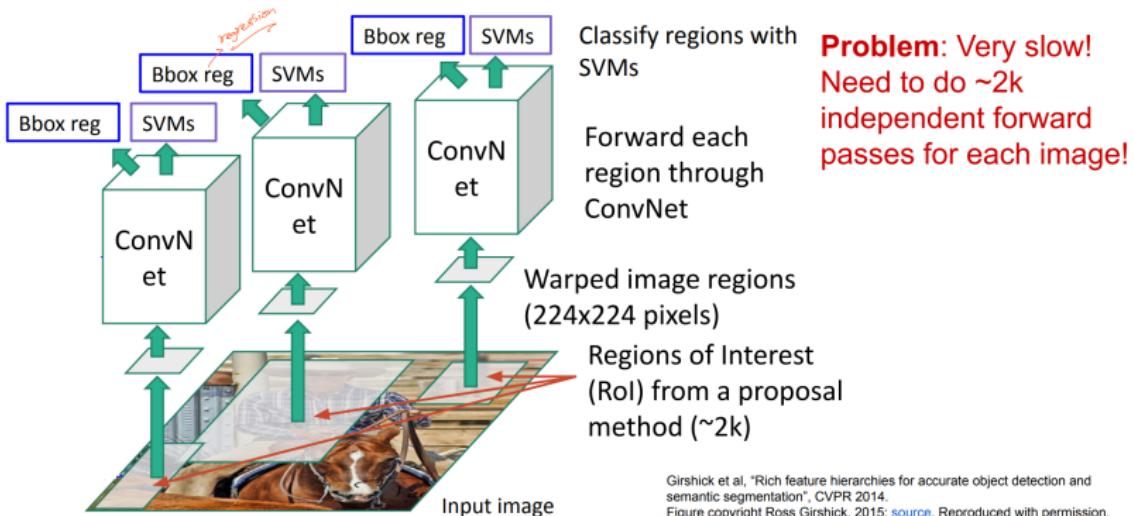
Predict “corrections” to the RoI: 4 numbers: (dx, dy, dw, dh)



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 (cont.)

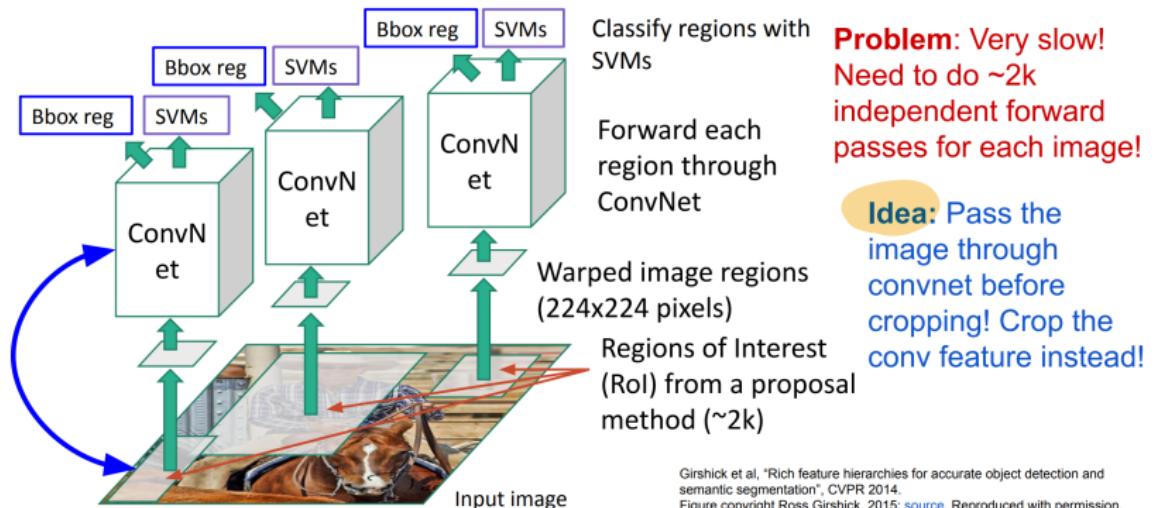
Predict “corrections” to the RoI: 4 numbers: (dx, dy, dw, dh)



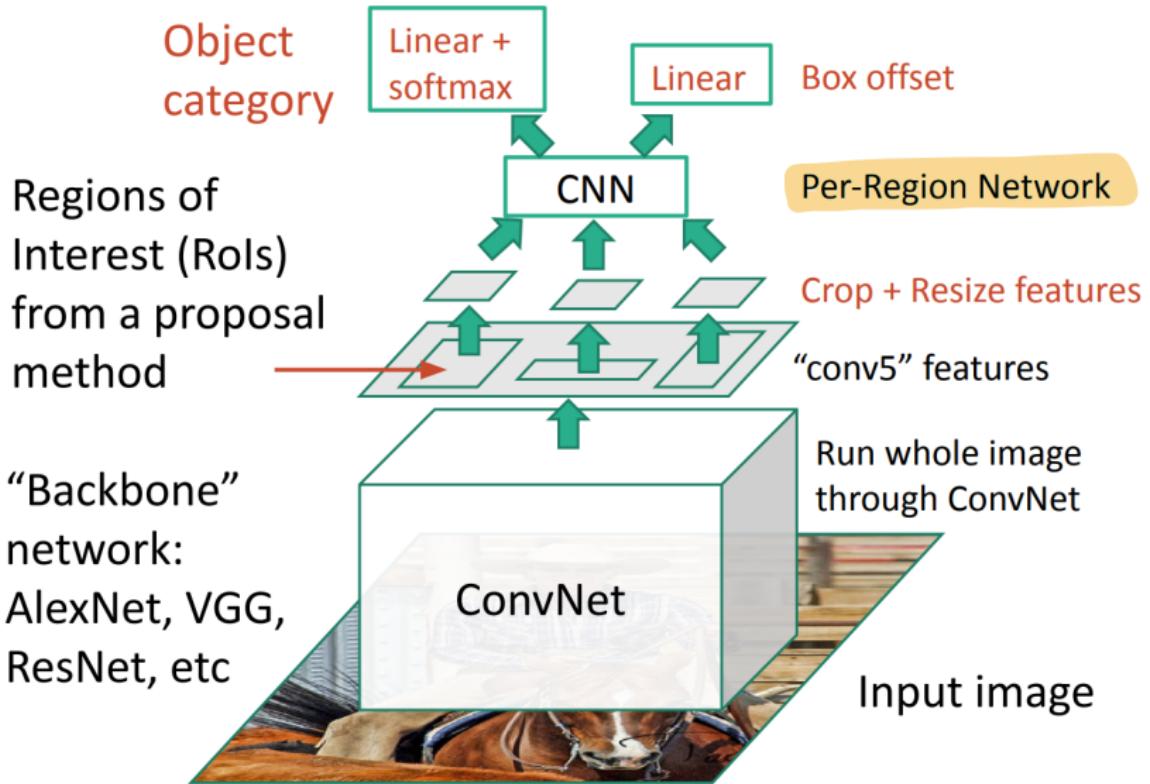
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.

R-CNN (cont.)

Predict “corrections” to the RoI: 4 numbers: (dx, dy, dw, dh)



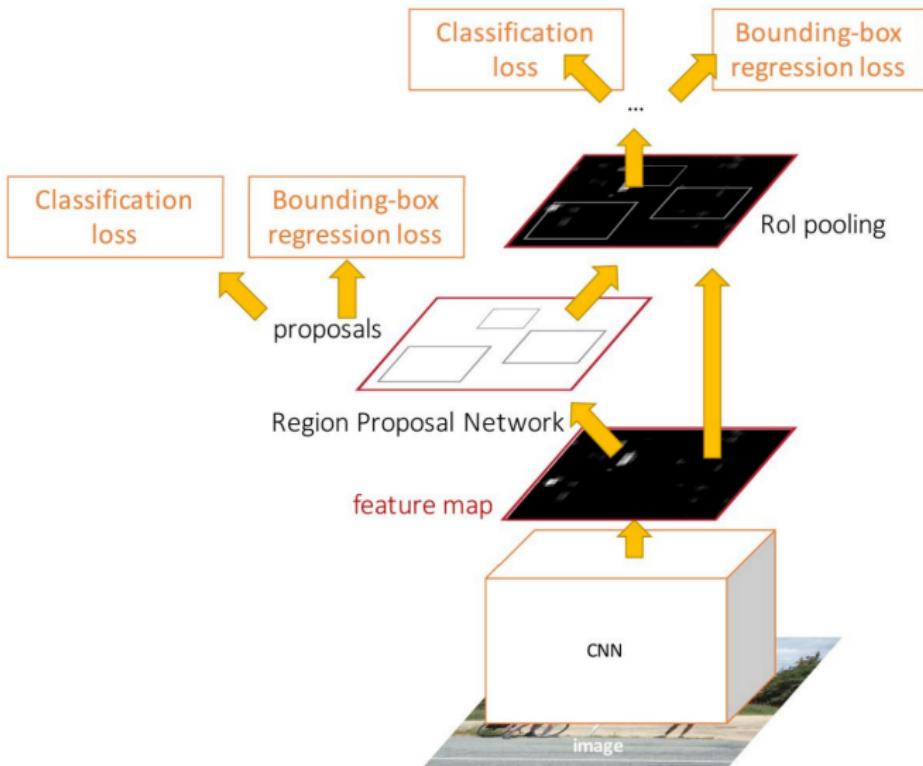
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; [source](#). Reproduced with permission.



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

- ▶ Make CNN do proposals!
- ▶ Insert Region Proposal Network (RPN) to predict proposals from features
- ▶ Jointly train on 4 losses:
 - **RPN classification:** anchor box is object / not an object
 - **RPN regression:** predict transform from anchor box to proposal box
 - **Object classification:** classify proposals as background / object class
 - **Object regression:** predict transform from proposal box to object box

Faster R-CNN



Faster R-CNN: Make CNN do proposals!

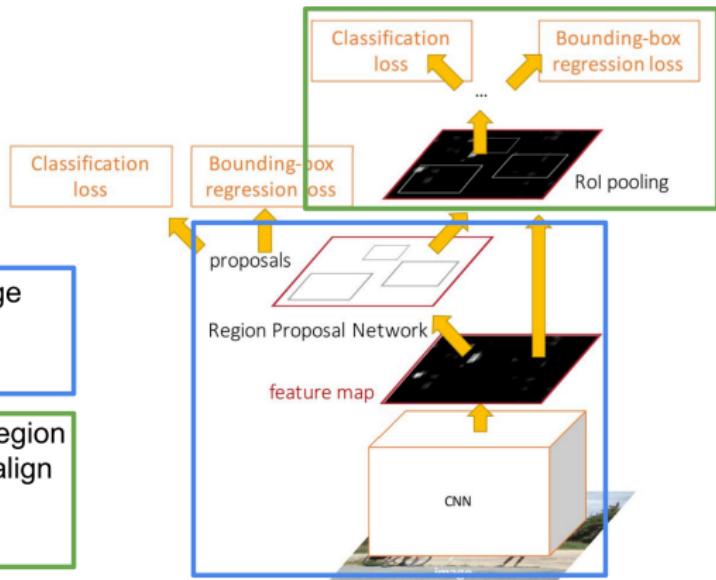
Faster R-CNN is a
Two-stage object detector

First stage: Run once per image

- Backbone network
- Region proposal network

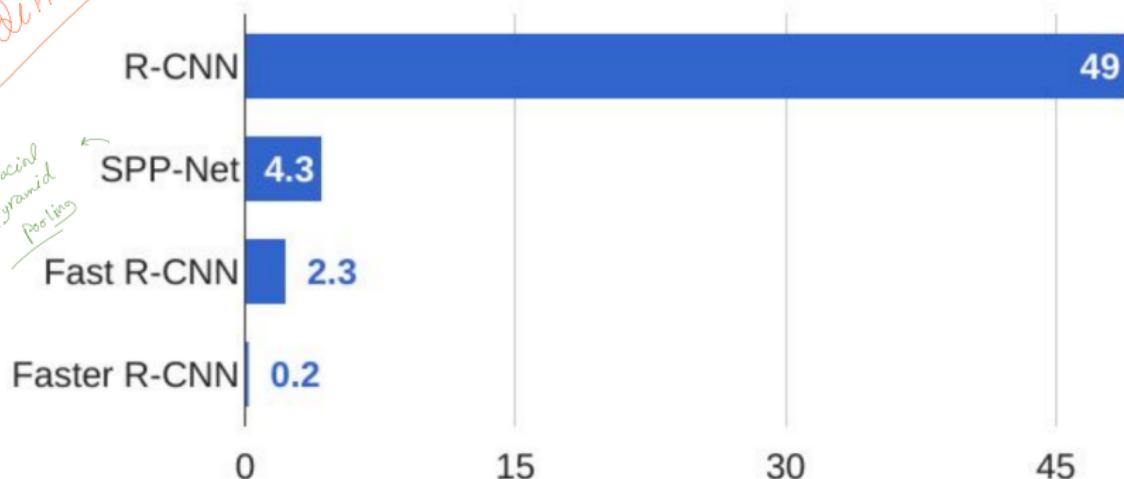
Second stage: Run once per region

- Crop features: RoI pool / align
- Predict object class
- Prediction bbox offset



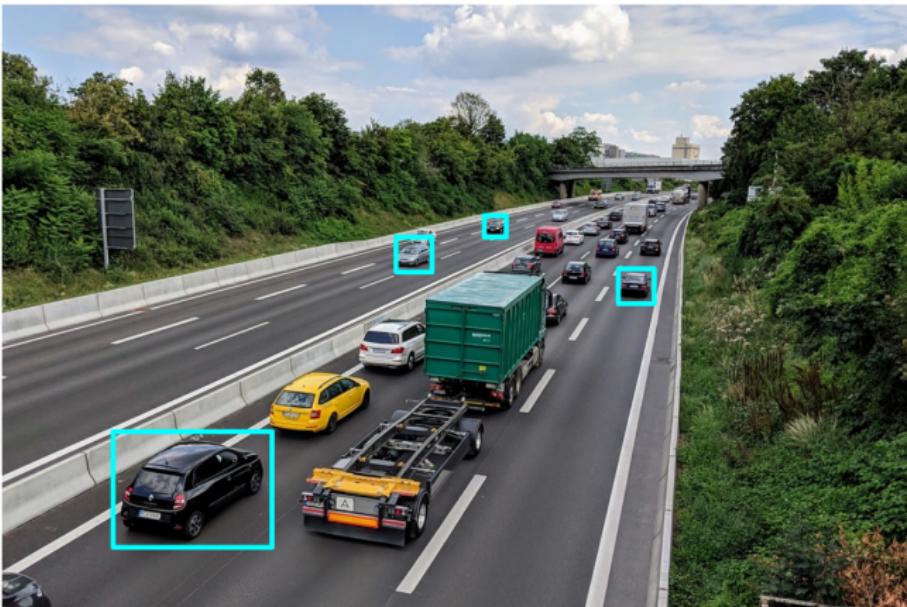
Check
the differences

R-CNN Test-Time Speed



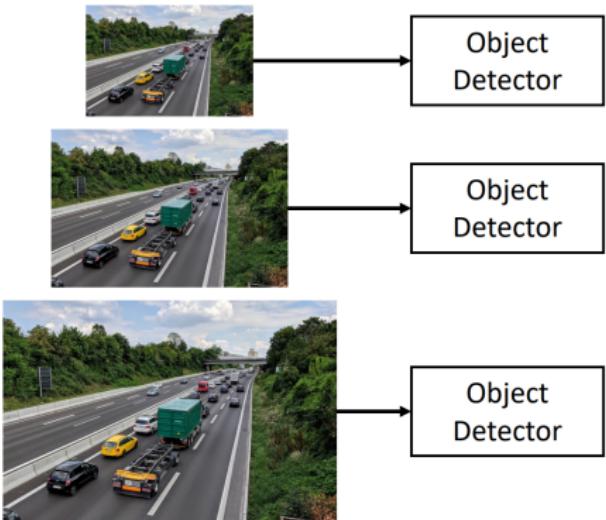
Dealing with Scale

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



Dealing with Scale: Image Pyramid

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



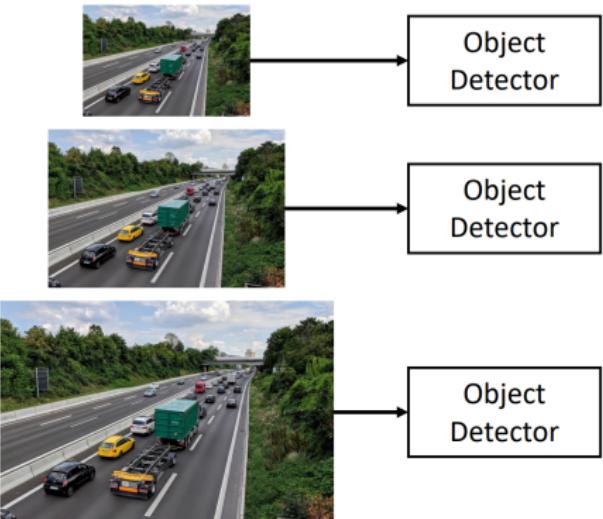
Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

Dealing with Scale: Image Pyramid (cont.)

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

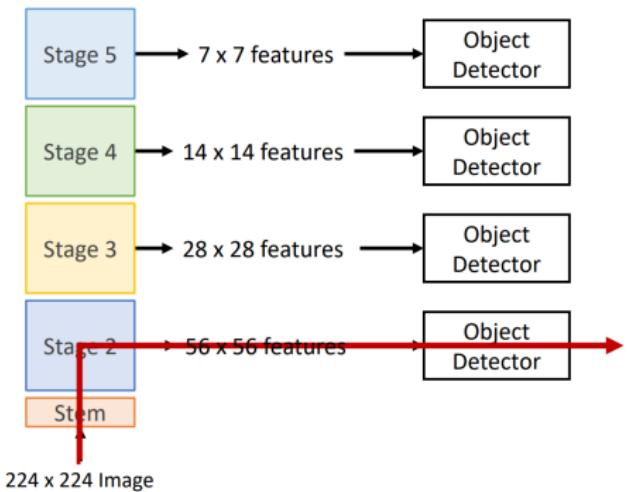
Problem: Expensive! Don't share any computation between scales

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



Dealing with Scale: Image Pyramid

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



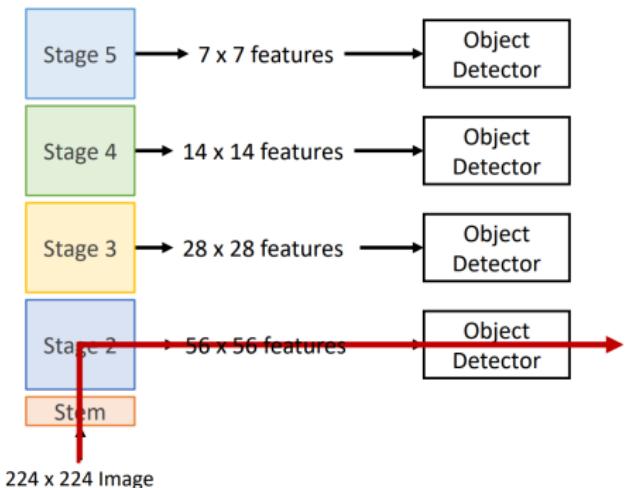
Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

Dealing with Scale: Image Pyramid (cont.)

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

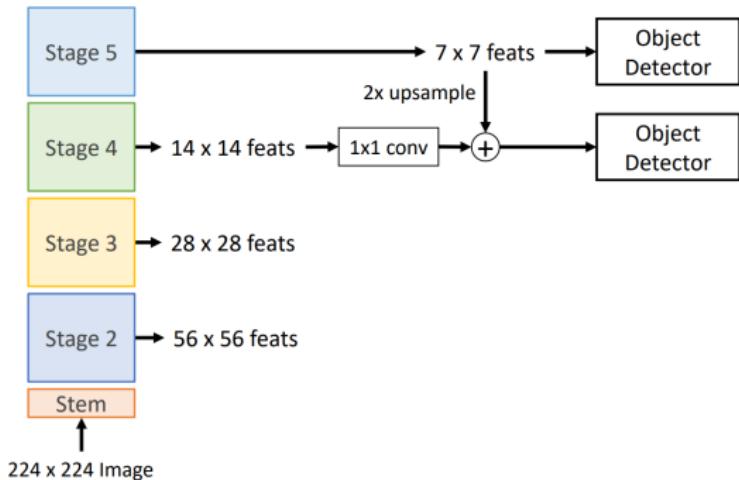
Problem: detector on early features doesn't make use of the entire backbone; doesn't get access to high-level features

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



Dealing with Scale: Feature Pyramid Network

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



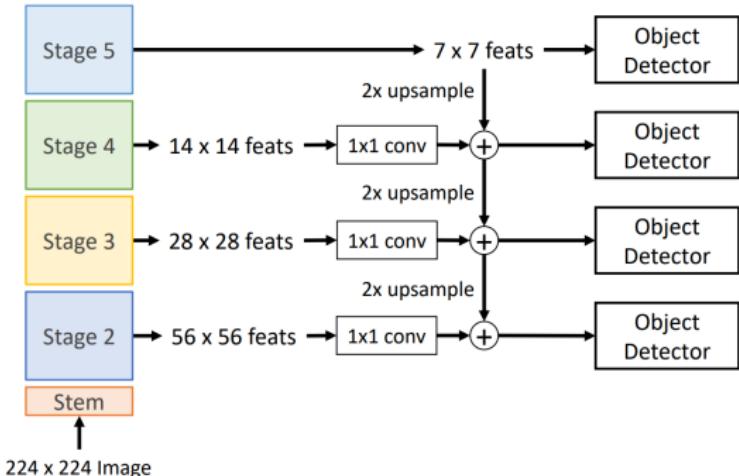
Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

Dealing with Scale: Feature Pyramid Network (cont.)

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

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

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

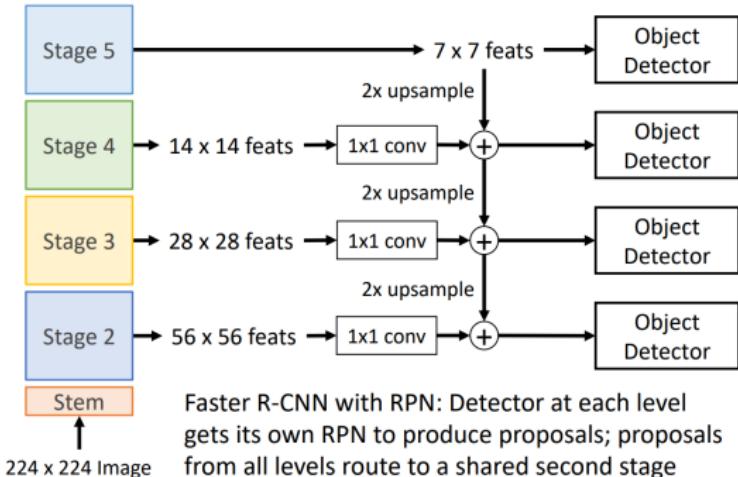


Dealing with Scale: Feature Pyramid Network (cont.)

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

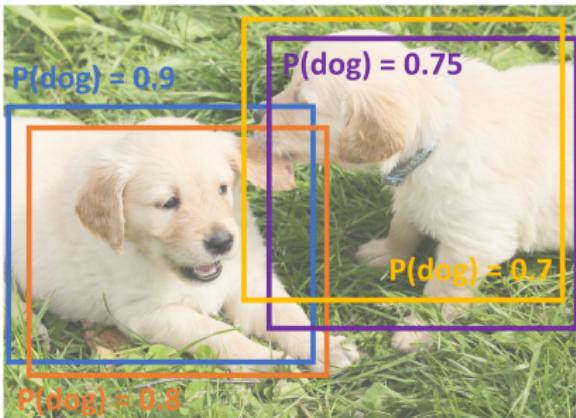
Efficient multiscale features where all levels benefit from the whole backbone! Widely used in practice

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



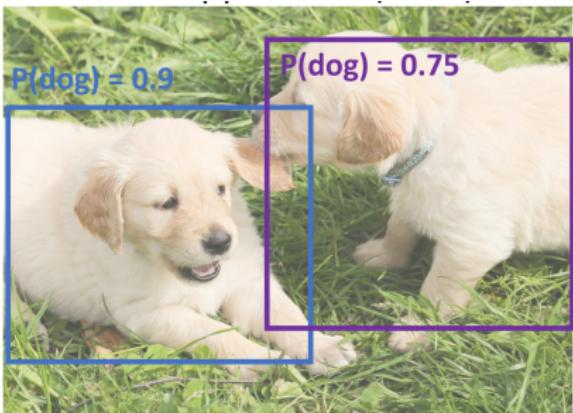
Overlapping Boxes: Non-Max Suppression (NMS)

- ▶ **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 $\text{IoU} > \text{threshold}$ (e.g. 0.7)
 3. If any boxes remain, GOTO 1



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 $\text{IoU} > \text{threshold}$ (e.g. 0.7)
 3. If any boxes remain, GOTO 1



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
 3. with $\text{IoU} > \text{threshold}$ (e.g. 0.7)
 4. If any boxes remain, GOTO 1
- ▶ **Problem:** NMS may eliminate "good" boxes when objects are highly overlapping... no good solution =(



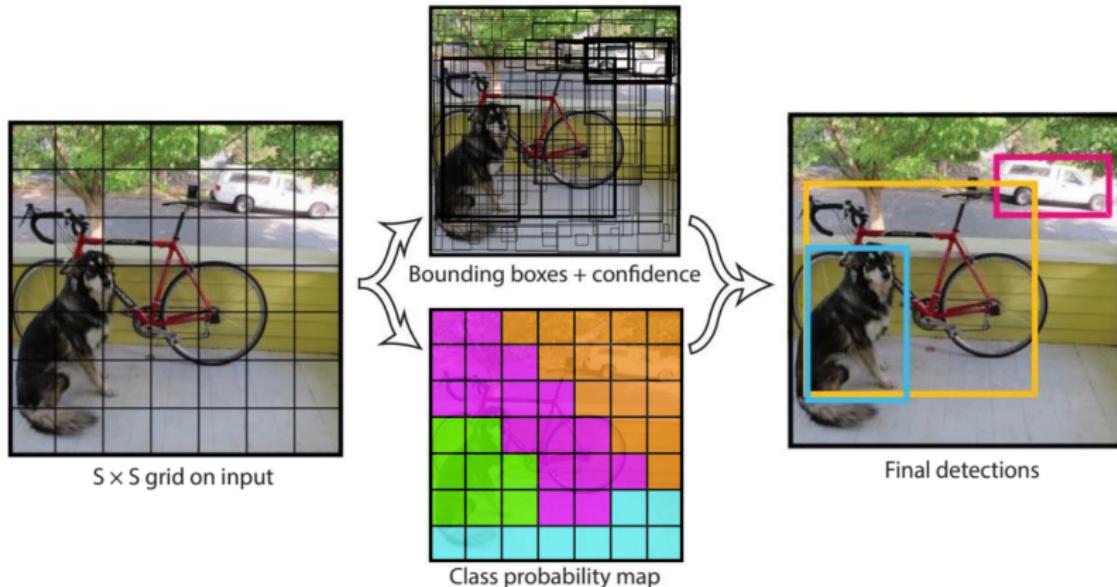
Single Shot Object Detection



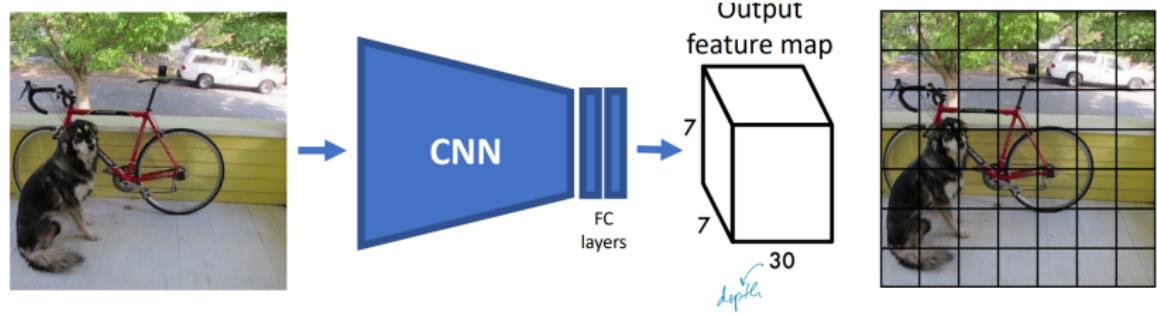
Single Shot:
SSD, YOLO ...

Fast
High false rate

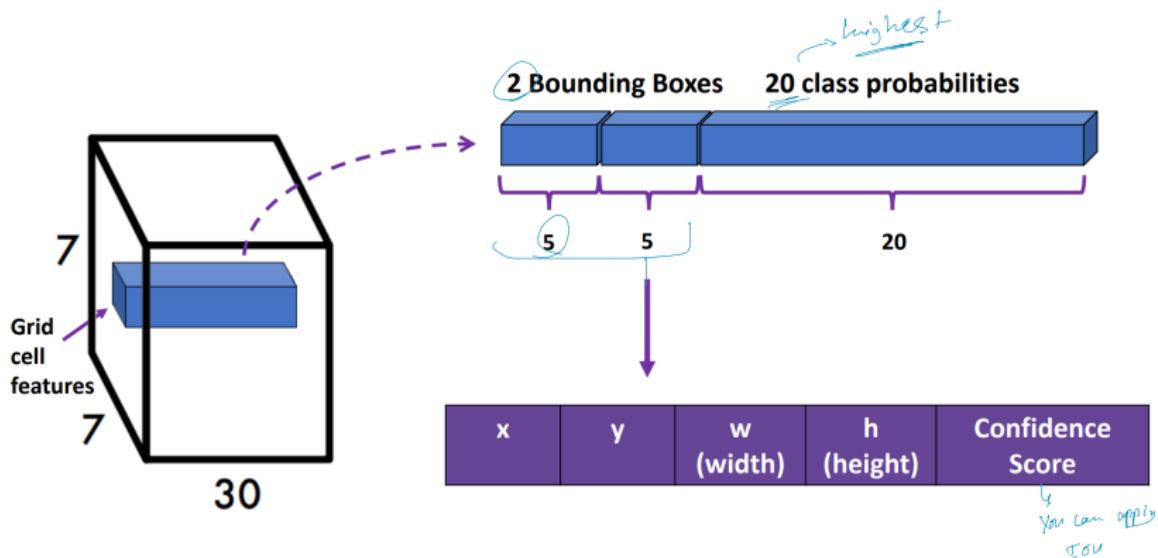
YOLO - Overview



YOLO - Overview



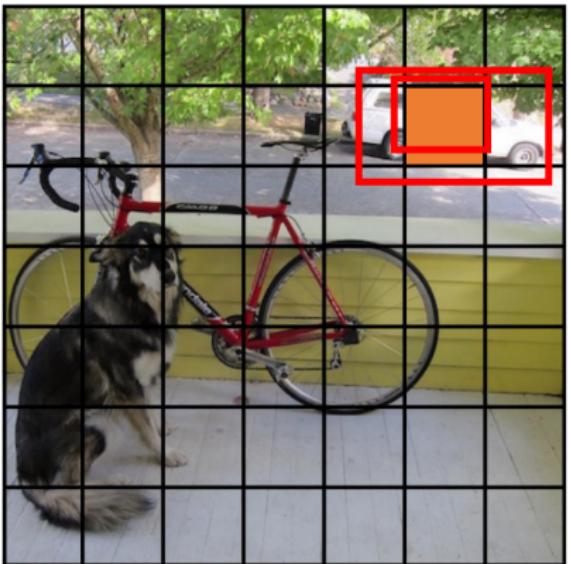
YOLO - Overview





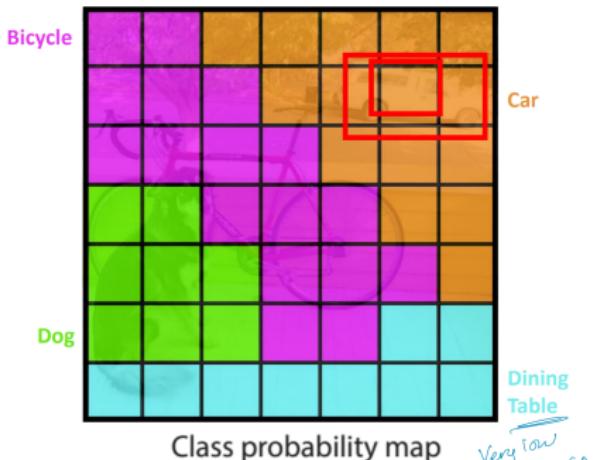
Each cell predicts

- ▶ $B = 2$ bounding boxes
 $(x, y, w, h) +$ confidence score
- ▶ $C = 20$ class probabilities



Each cell predicts

- ▶ $B = 2$ bounding boxes
 $(x, y, w, h) +$ confidence score
- ▶ $C = 20$ class probabilities





$$\cancel{7 \times 7 = 49}$$
$$49 \times 2 =$$
$$\frac{49 \times}{\cancel{49} \times}$$

If Should be $49 \times$

SxSxB Bounding-Boxes ($S=7, B=2 \rightarrow 96$ Bboxes)



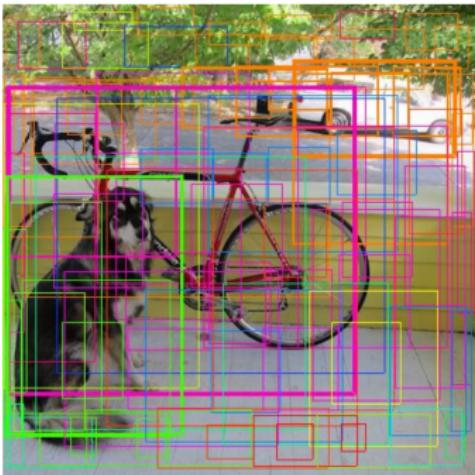
$S \times S$ grid on input

Each cell predicts

- ▶ $B = 2$ bounding boxes
 $(x, y, w, h) +$ confidence score
- ▶ $C = 20$ class probabilities

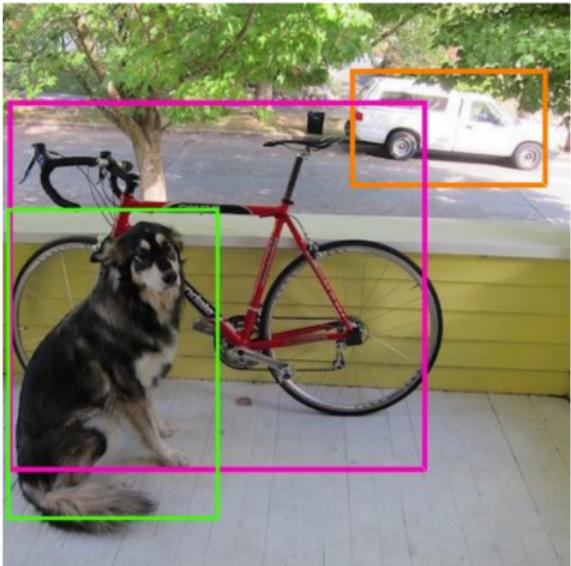
(colorful)

SxSxB Bounding-Boxes ($S=7, B=2 \rightarrow 96$ Bboxes)



Each cell predicts

- ▶ $B = 2$ bounding boxes
 $(x, y, w, h) +$ confidence score
- ▶ $C = 20$ class probabilities
- ▶ Apply Non-Maximum Suppression



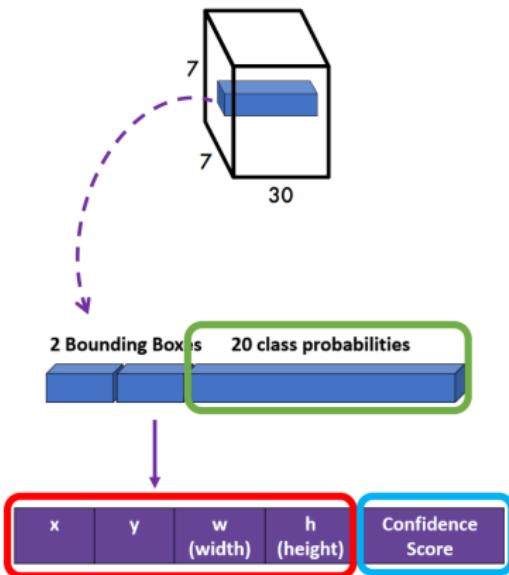
YOLO - Loss Function

YOLO – Loss function

$$\mathcal{L} = \mathcal{L}_{Localization\ Loss}$$

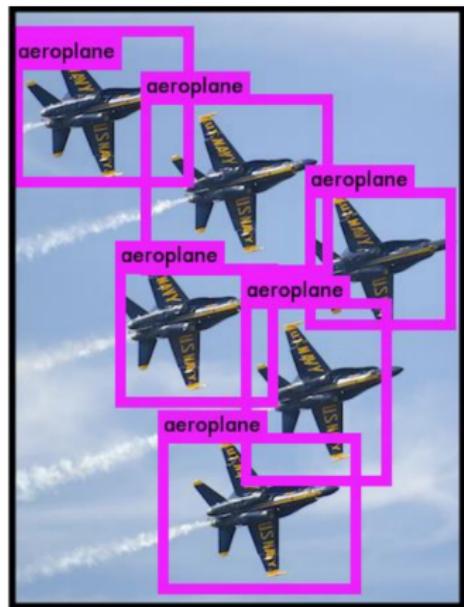
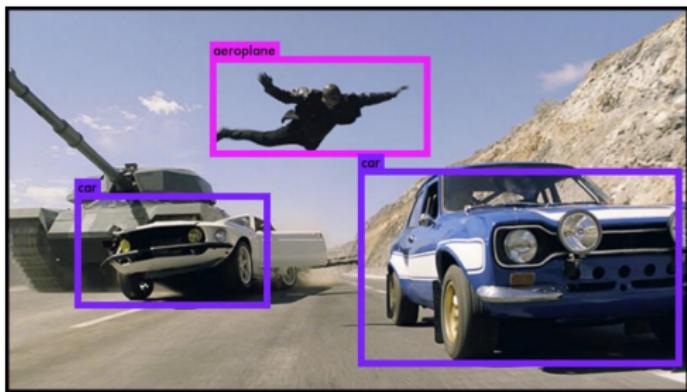
$$+ \mathcal{L}_{\text{Confidence Loss}}$$

+ $\mathcal{L}_{Classification\ Loss}$



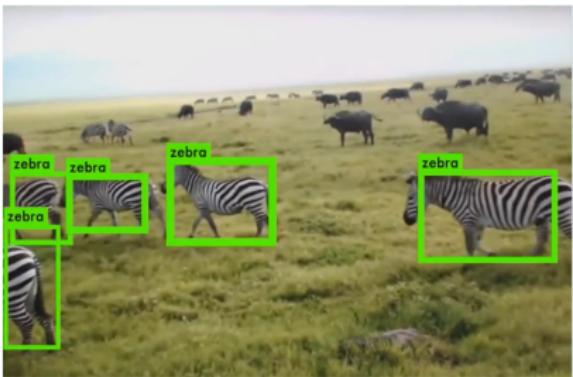
YOLO - Benefits

- ▶ Fast. Good for real-time processing
- ▶ End-to-end training



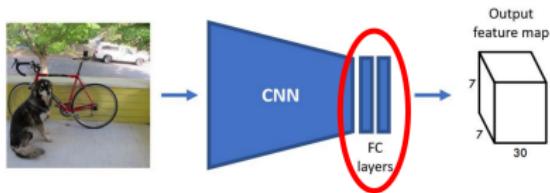
YOLO - Limitations

- ▶ Difficult to detect small objects
 - ▶ Coarse predictions



YOLO - Limitations

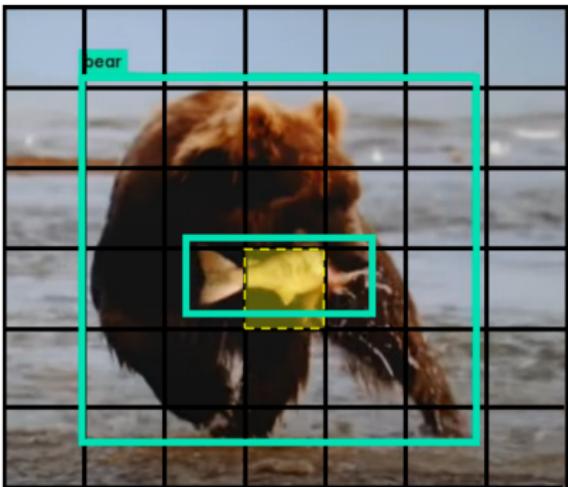
- ▶ Difficult to detect small objects
 - ▶ Coarse predictions
 - ▶ Fixed input size



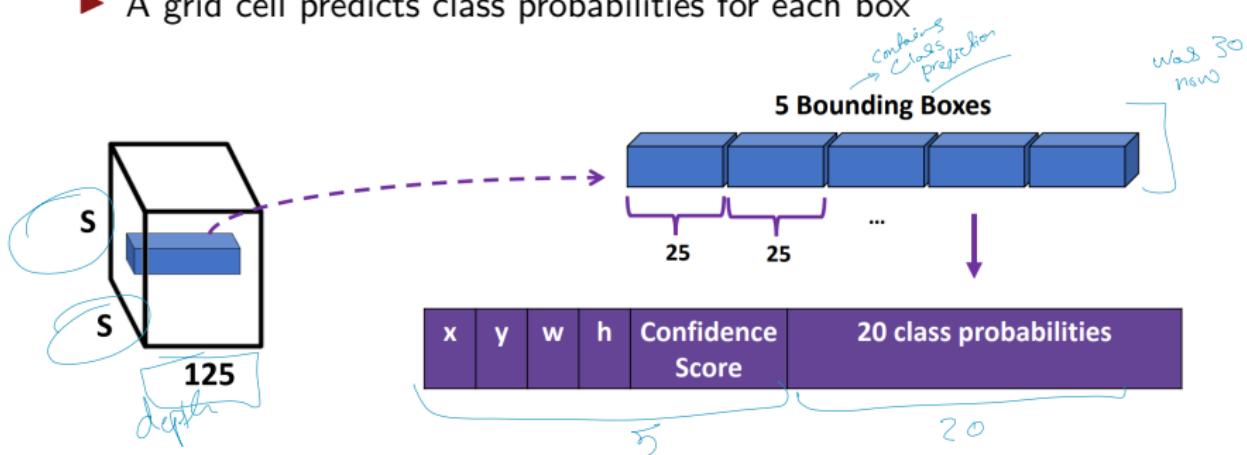
YOLO - Limitations



- ▶ Difficult to detect small objects
 - ▶ Coarse predictions
 - ▶ Fixed input size
 - ▶ A grid cell can predict only one class
 - ▶ Solutions:
 - Remove fc layers!
 - Predict class per bbox (not per cell)



- ▶ Removed fully connected layers
- ▶ A grid cell predicts class probabilities for each box



► YOLOv3

- J. Redmon, A. Farhadi. Yolov3: An incremental improvement, 2018

► YOLOv4

- A. Bochkovskiy, C. Wang, H. Liao. Yolov4: Optimal speed and accuracy of object detection (Feb. 2020)

► YOLOv5

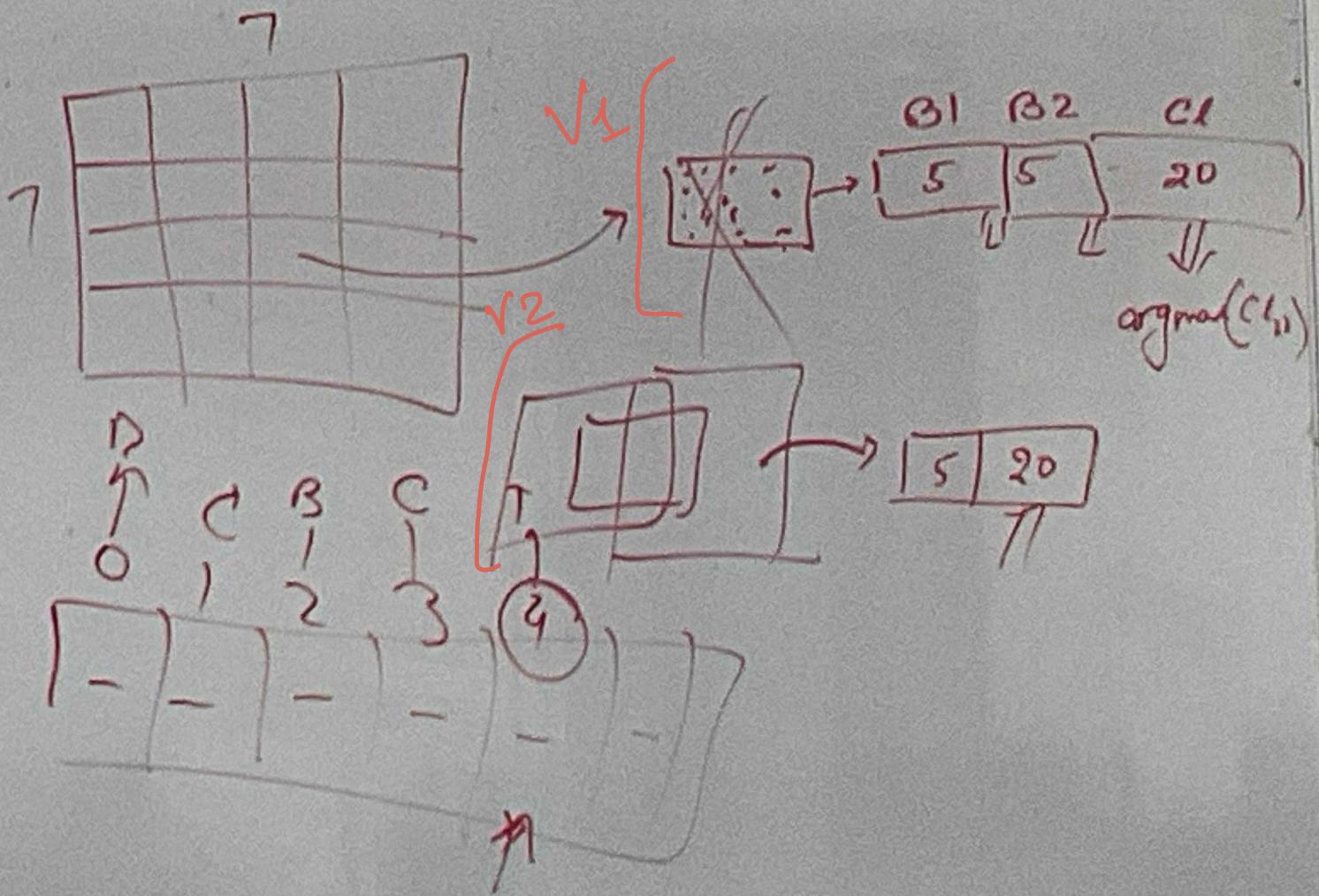
- YOLOv5 by ultralytics (June 2020)

► PP-YOLO

- X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, S. Wen. Pp-yolo: An effective and efficient implementation of object detector (June 2020)

► PP-YOLOv2 (2021)

- J. X. Huang, X. Wang, W. Lv, X. Bai, X. Long, K. Deng, Q. Dang, S. Han, Q. Liu, X. Hu, D. Yu, Y. Ma, O. Yoshie. PP-YOLOv2: A Practical Object Detector (2021)



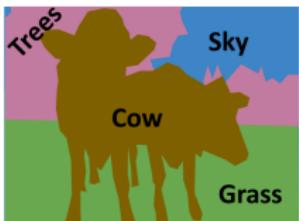
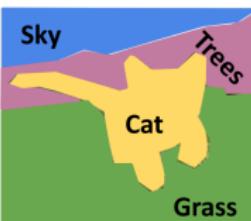
Things and Stuff



- ▶ **Things:** Object categories that can be separated into object instances (e.g. cats, cars, person)
- ▶ **Stuff:** Object categories that cannot be separated into instances (e.g. sky, grass, water, trees)

Interest
Not interest

Interest
In



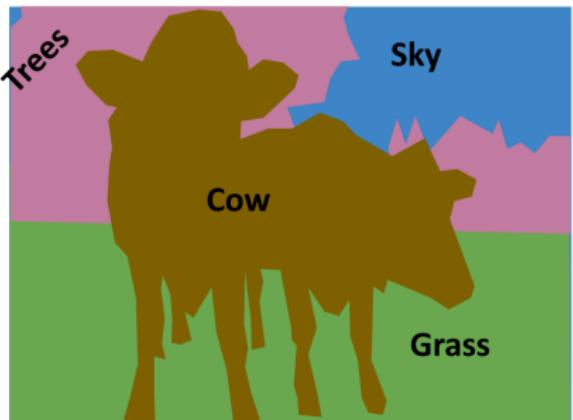
Semantic → meaning of each pixel
Instance →

Computer Vision Tasks

- ▶ **Object Detection:** Detects individual object instances, but only gives box(Only things!)

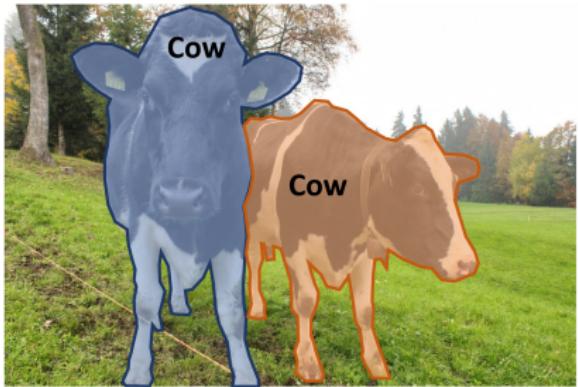


- ▶ **Semantic Segmentation:** Gives per-pixel labels, but merges instances (Both things and stuff)



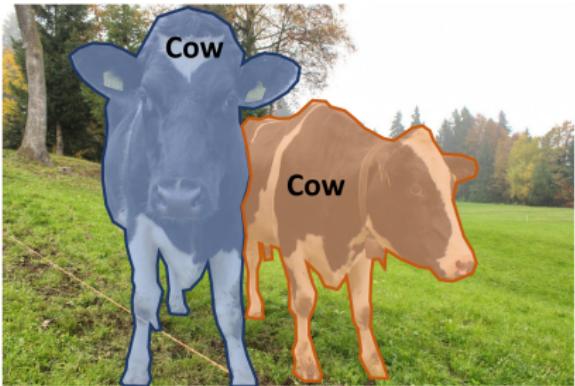
Instance Segmentation

- ▶ Detect all objects in the image, and identify the pixels that belong to each object (Only things!)



Instance Segmentation

- ▶ Detect all objects in the image, and identify the pixels that belong to each object (Only things!)
- ▶ **Approach:** Perform object detection, then predict a segmentation mask for each object!

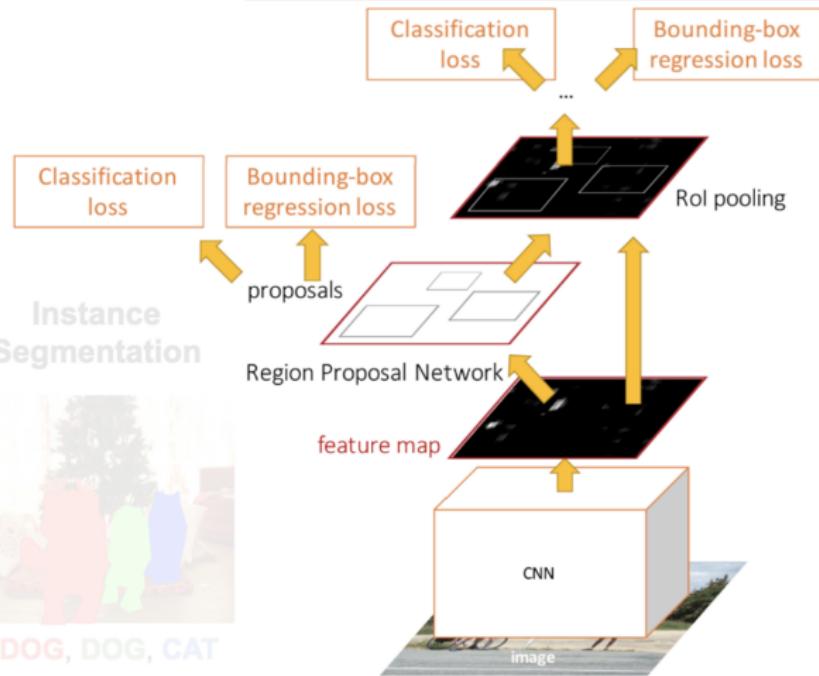


Object Detection: Faster R-CNN

Object Detection



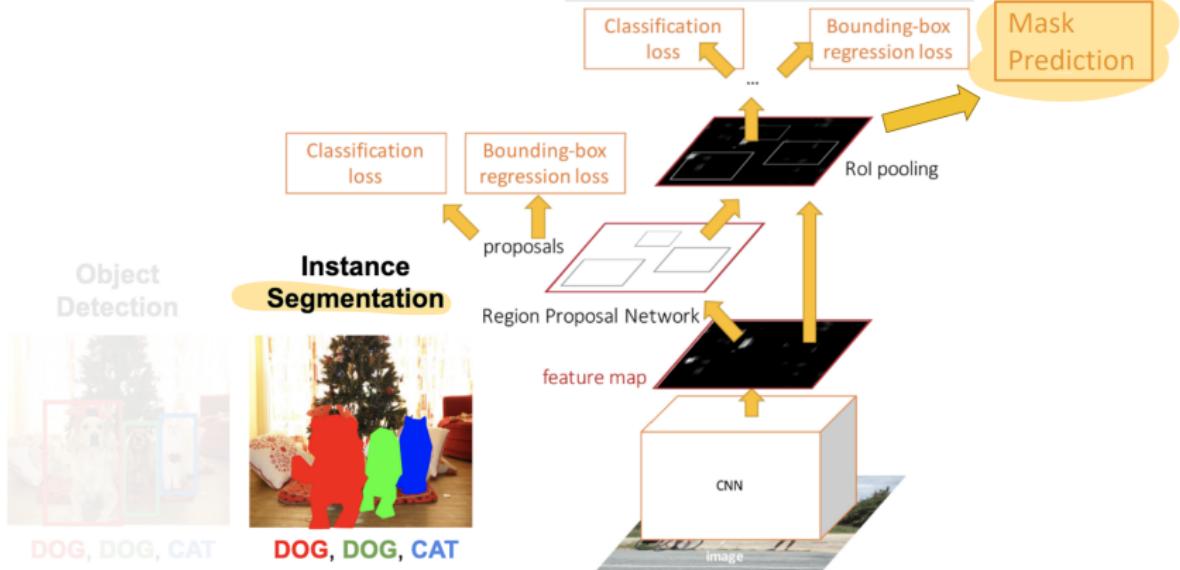
DOG, DOG, CAT



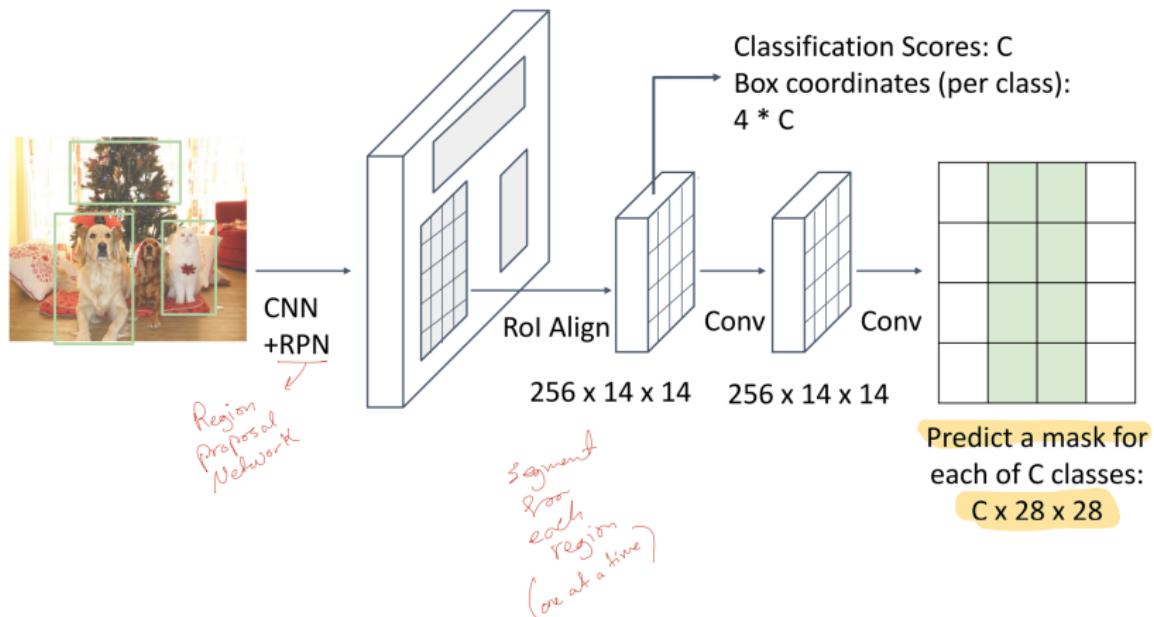
Instance Segmentation: Mask R-CNN



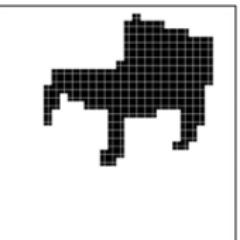
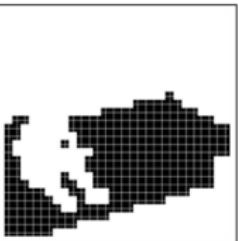
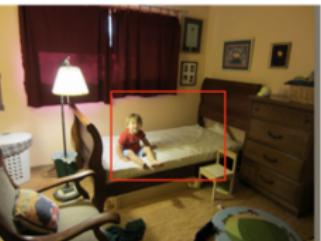
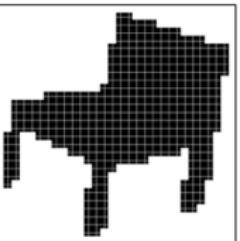
extra layer faster = RCNN



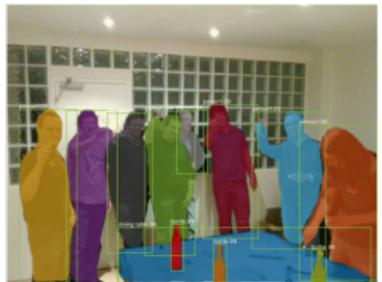
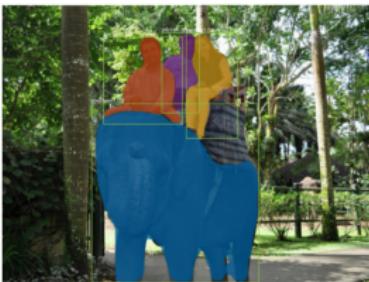
Instance Segmentation: Mask R-CNN



Mask R-CNN: Example Training Targets

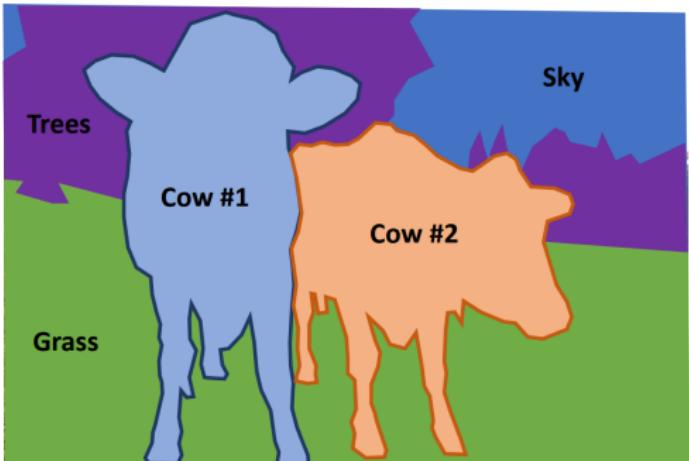


Mask R-CNN: Very Good Results!



Beyond Instance Segmentation: Panoptic Segmentation

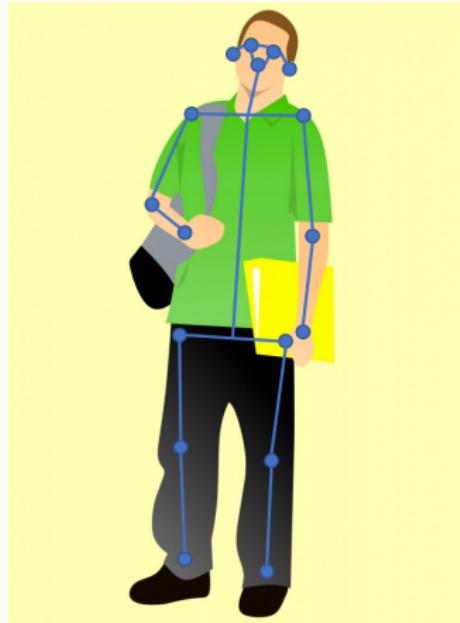
- ▶ Label all pixels in the image (both things and stuff)
- ▶ For "thing" categories also separate into instances



Beyond Instance Segmentation: Panoptic Segmentation

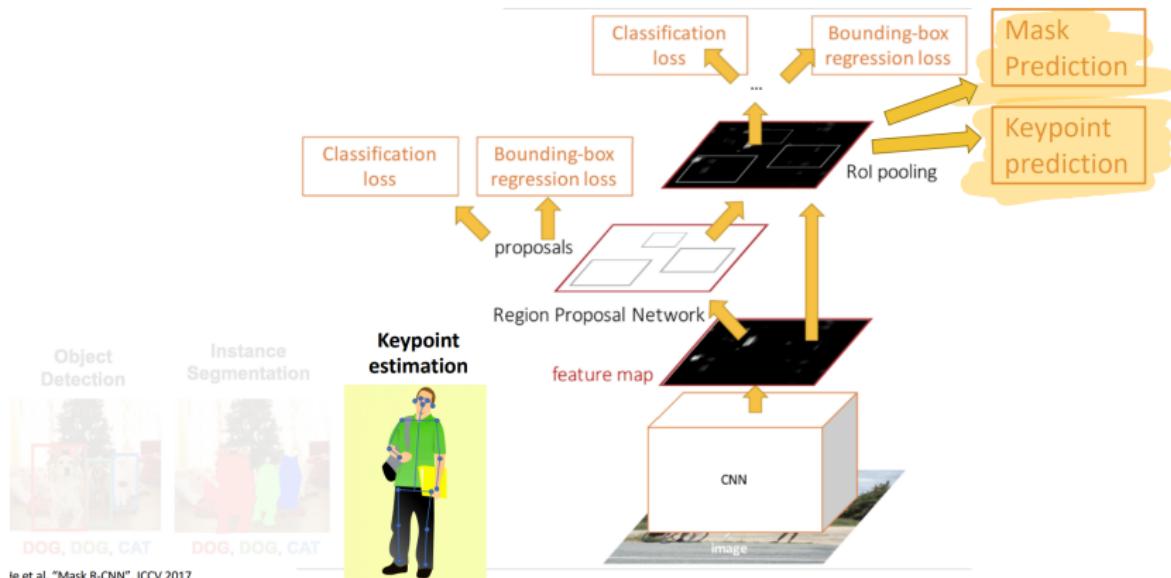


Beyond Instance Segmentation: Human Keypoints



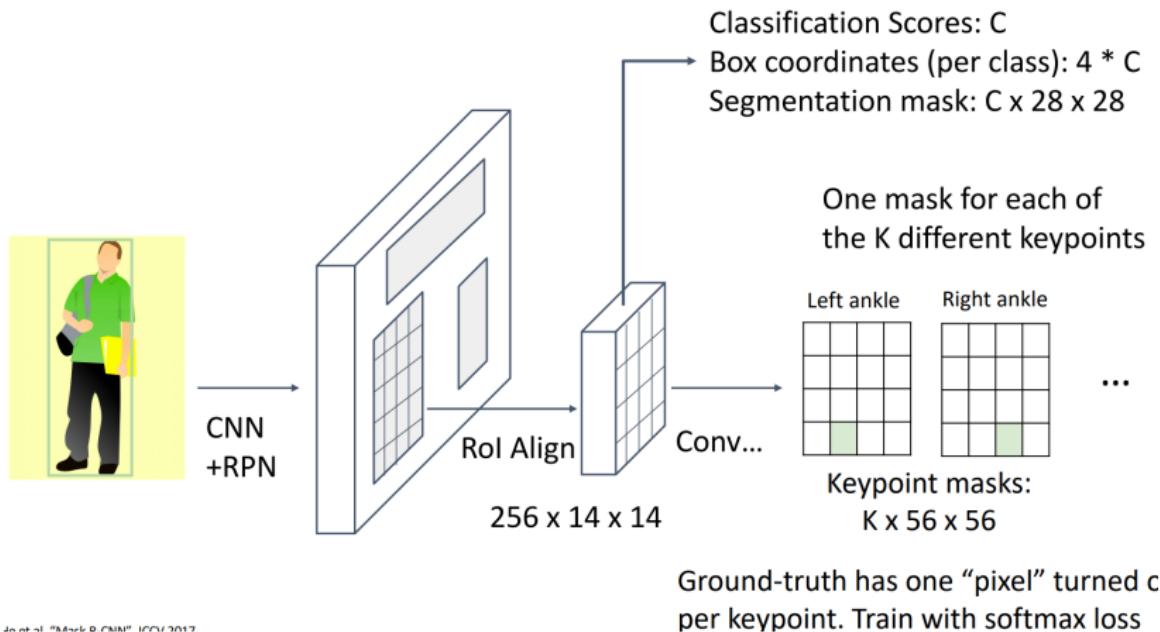
- ▶ Represent the pose of a human by locating a set of keypoint se.g. 17 keypoints:
 - ▶ Nose
 - ▶ Left / Right eye
 - ▶ Left / Right earLeft / Right shoulder
 - ▶ Left / Right elbow
 - ▶ Left / Right wrist

Mask R-CNN: Keypoint Estimation



Ie et al., "Mask R-CNN", ICCV 2017

Mask R-CNN: Keypoint Estimation

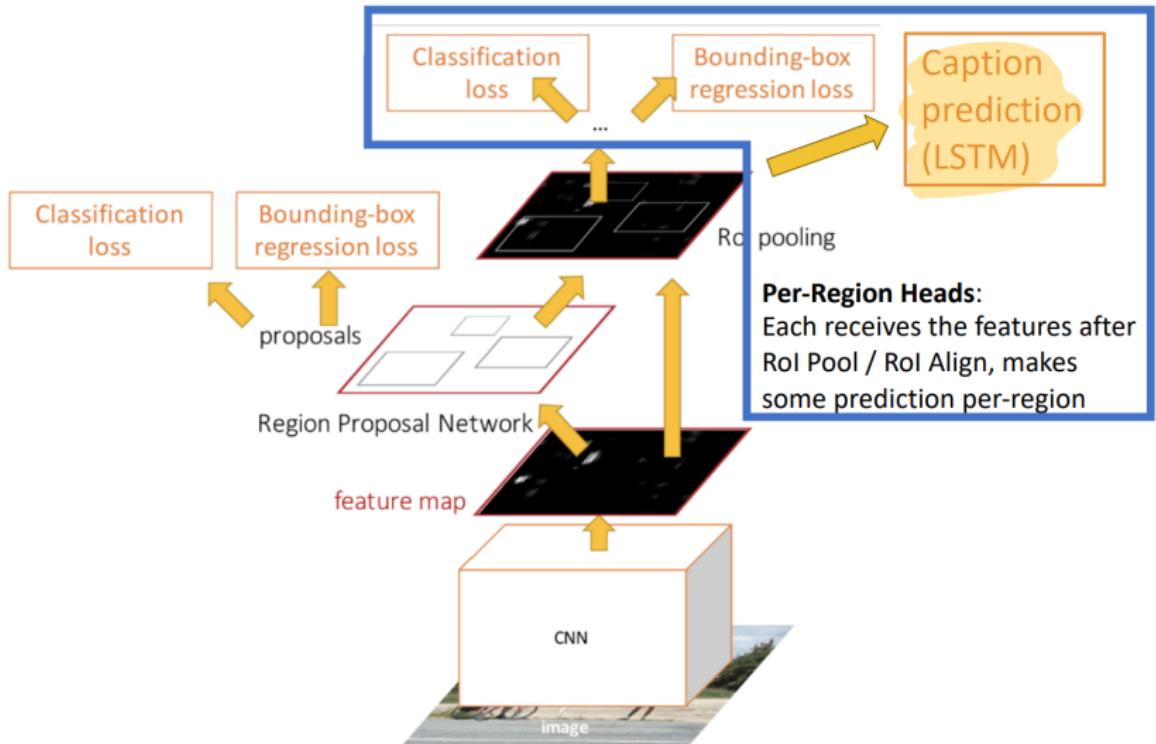


Jia et al. "Mask R-CNN". ICML 2017

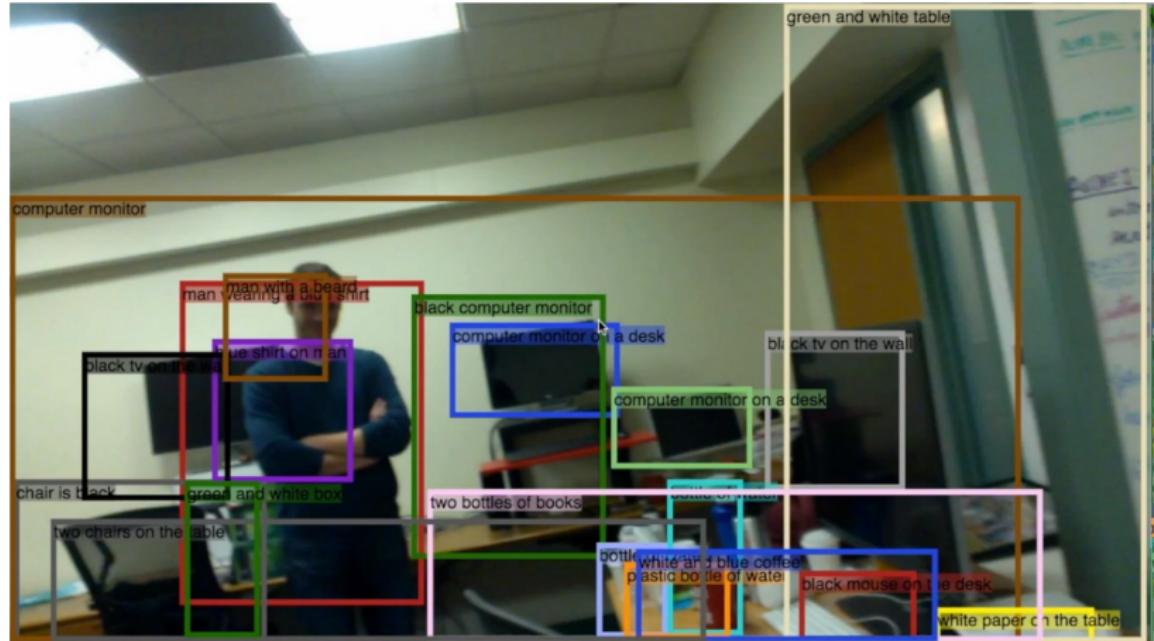
Joint Instance Segmentation and Pose Estimation



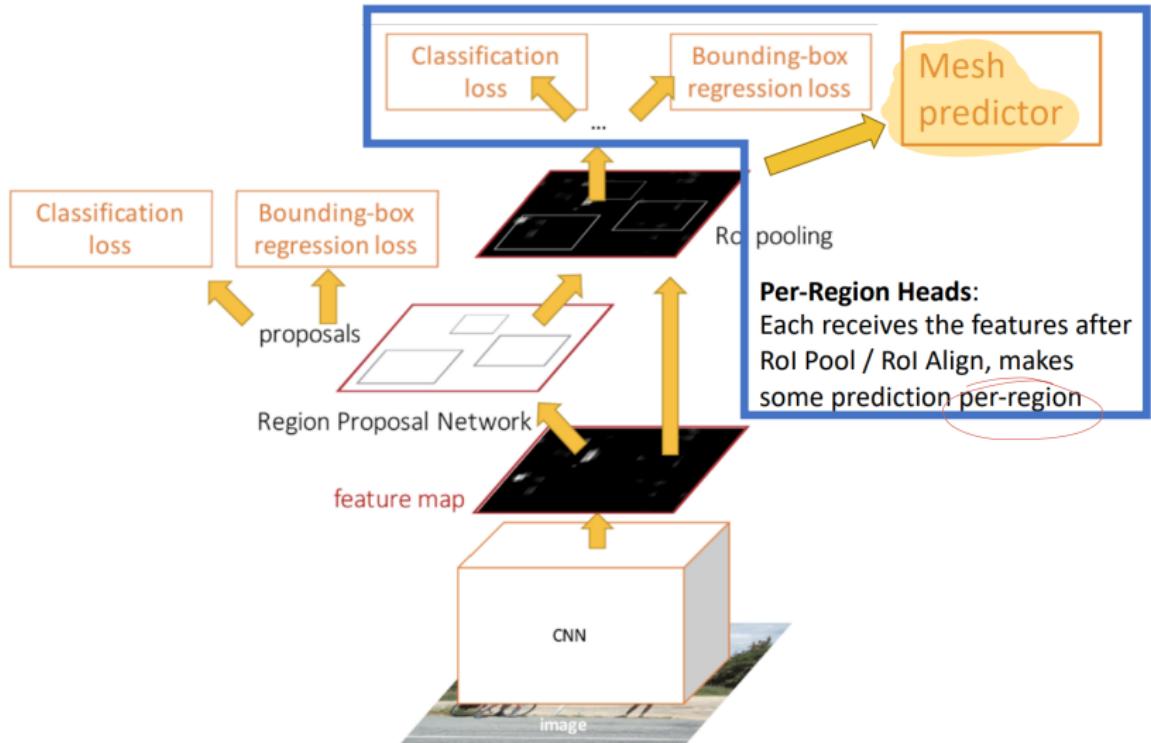
Captioning: Predict a caption per region!



Captioning: Predict a caption per region!

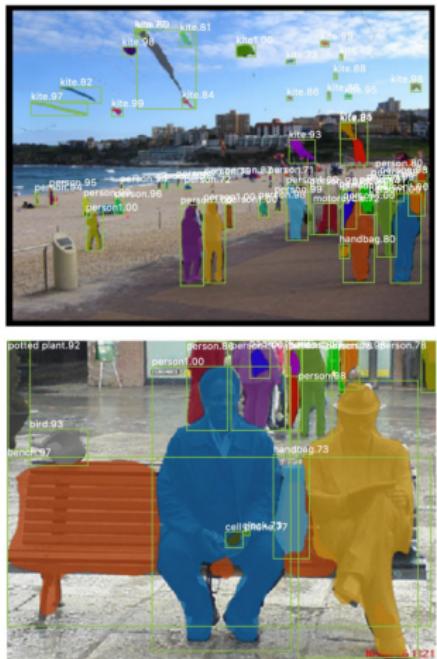


Johnson, Karpathy, and Fei-Fei, "DenseCap: Fully Convolutional Localization Networks for Dense Captioning", CVPR 2016

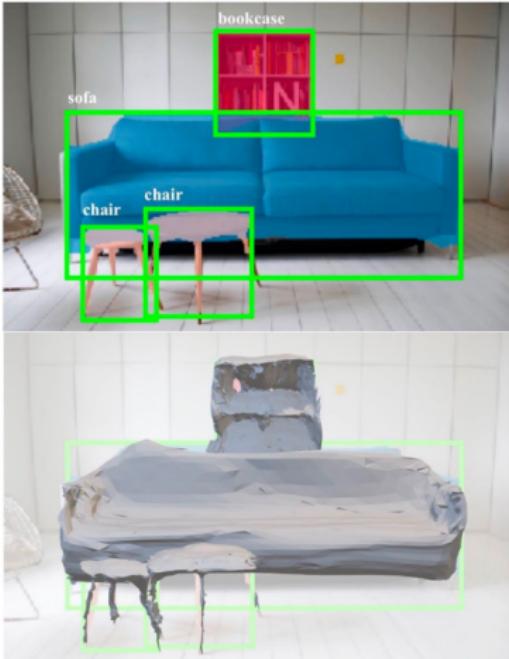


3D Shape Prediction

Mask R-CNN:
2D Image -> 2D shapes



Mesh R-CNN: 2D Image \rightarrow **3D shapes**



Gkioxari, Malik, and Johnson, "Mesh R-CNN", ICCV 2019

Temporal:

- ▶ **Goal:** Track objects over a sequence of photos or a video
- ▶ Exceedingly challenging in multi-object tracking scenarios
- ▶ Need to take care of not mixing up or losing objects midway
- ▶ **One Solution:** Perform object detection and assign IDs to each object and store its feature vector. Then track the objects based on its ID and feature vector

Object Tracking

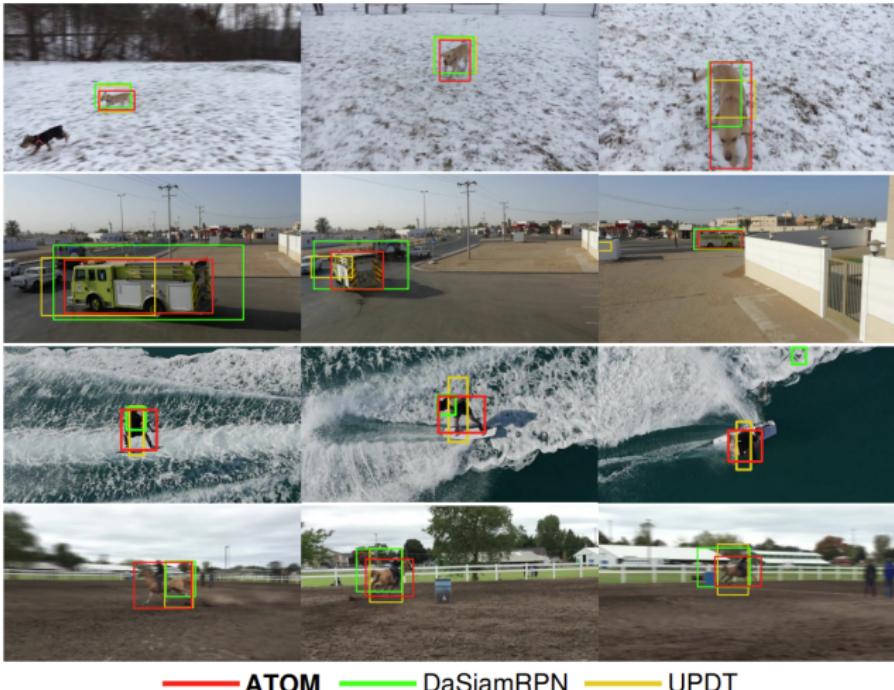


Figure 2: Comparison of 3 approaches for object tracking

Danelljan et al.

These slides have been adapted from

- ▶ Fei-Fei Li, Yunzhu Li & Ruohan Gao, Stanford CS231n: Deep Learning for Computer Vision
- ▶ Assaf Shocher, Shai Bagon, Meirav Galun & Tali Dekel, WAIC DL4CV Deep Learning for Computer Vision: Fundamentals and Applications
- ▶ Justin Johnson, UMich EECS 498.008/598.008: Deep Learning for Computer Vision