Computed Vision

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

KAUST Academy
King Abdullah University of Science and Technology

February 19, 2025

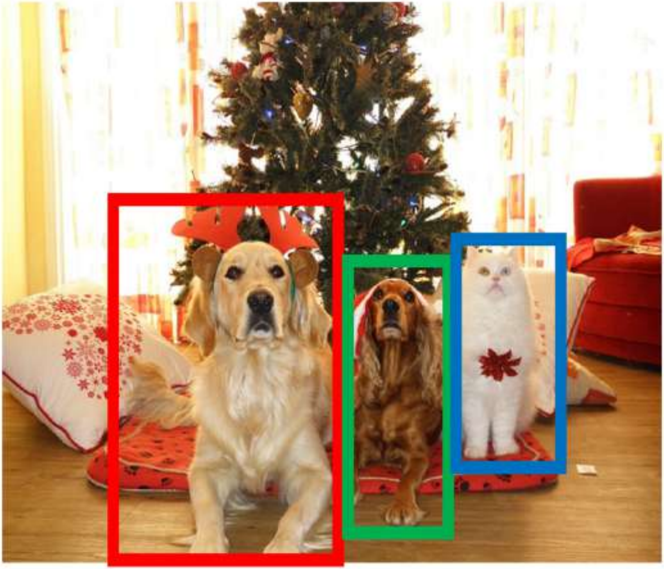
Table of Contents

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.

- **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)

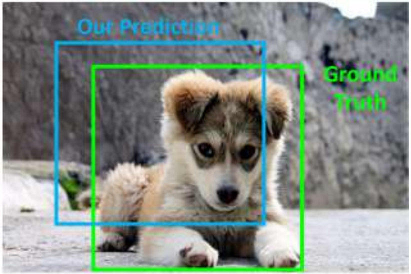
Object Detection (cont.)



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 $\sim 800 \times 600$

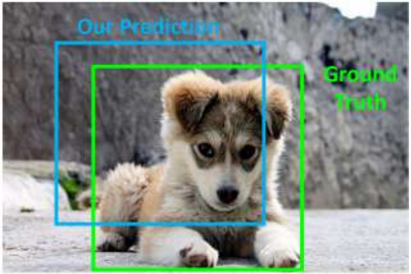
- How can we compare our prediction to the ground-truth box?



Comparing Boxes: Intersection over Union (IoU)

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

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



Detecting a Single Object

Often pretrained on ImageNet (Transfer learning)

Vector: 4096

Treat localization as a regression problem!

Detecting a Single Object (cont.)

“What”

Correct label:
Cat

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

Softmax
Loss

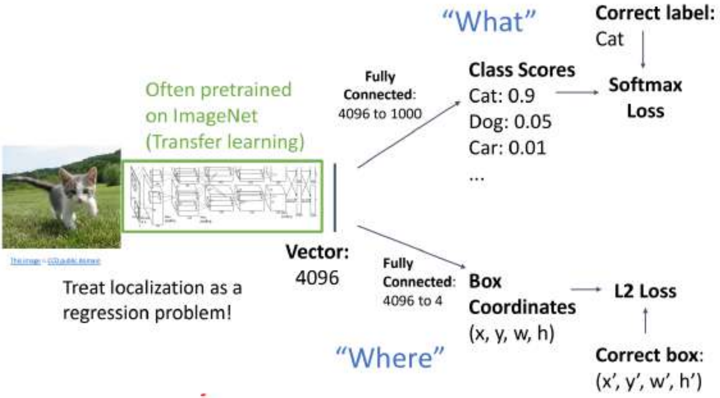
Fully
Connected:
4096 to 1000

Often pretrained
on ImageNet
(Transfer learning)

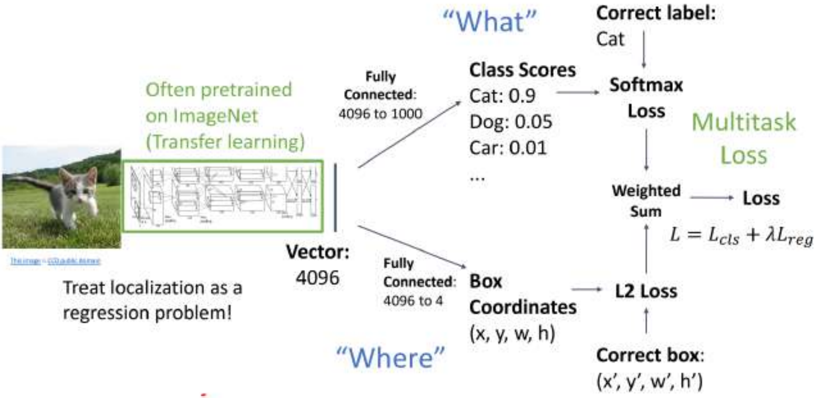
Vector:
4096

Treat localization as a
regression problem!

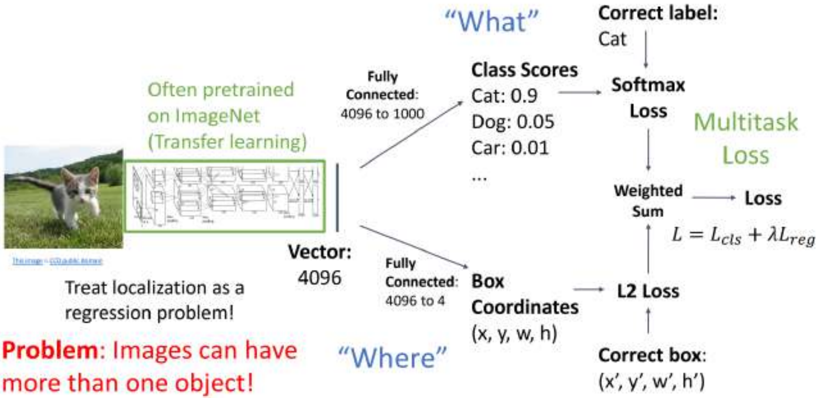
Detecting a Single Object (cont.)



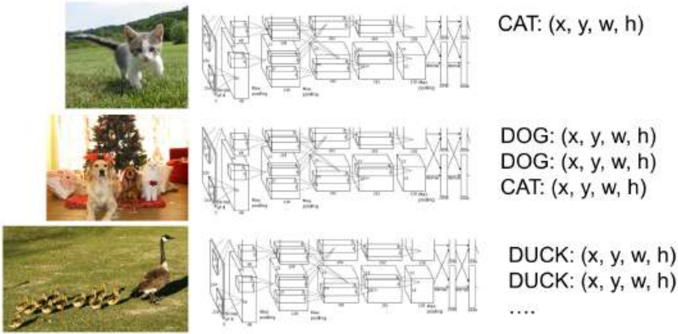
Detecting a Single Object (cont.)



Detecting a Single Object (cont.)

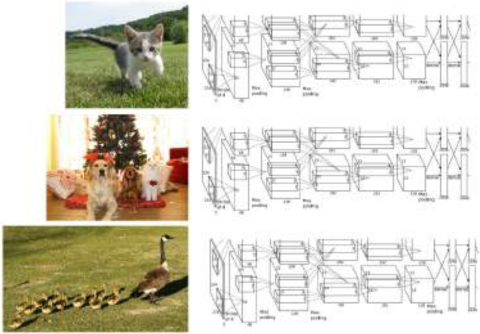


Multiple Objects



Multiple Objects (cont.)

Each image needs a different number of outputs!

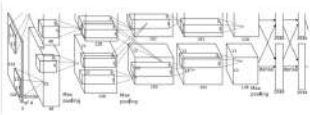


CAT: (x, y, w, h) — 4 numbers

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

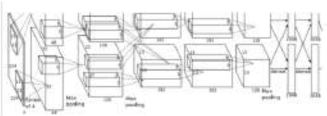
DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....
Many numbers!

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



Dog? NO
Cat? NO
Background? YES

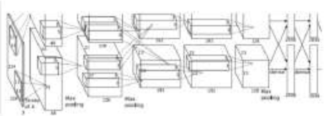
Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? YES
Cat? NO
Background? NO

Detecting Multiple Objects: Sliding Window (cont.)

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 (cont.)

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?

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 (cont.)

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 (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 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)

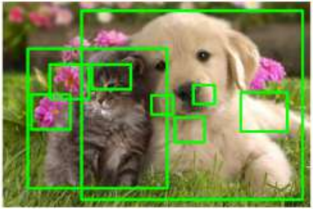
Total possible boxes:

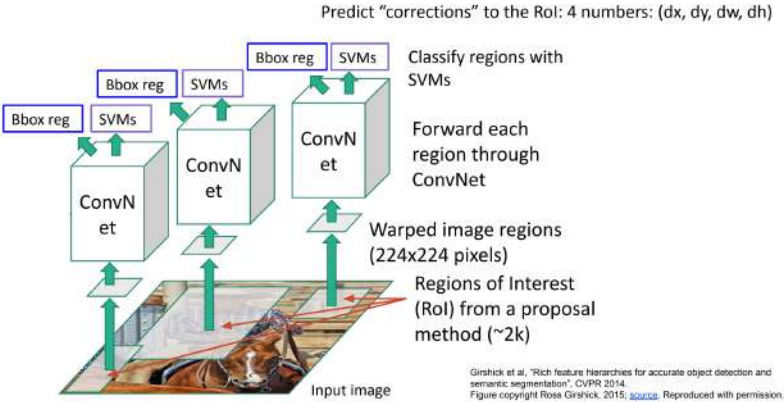
$$\sum_{h=1}^{H}\sum_{w=1}^{W}(W-w+1)(H-h+1)$$

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

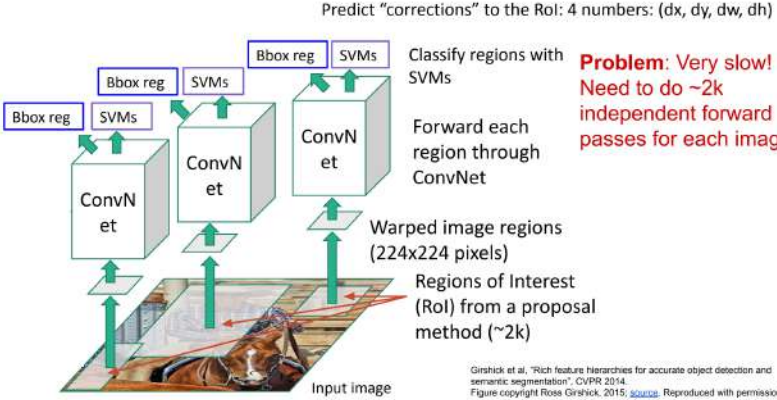
Region Proposal

- 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





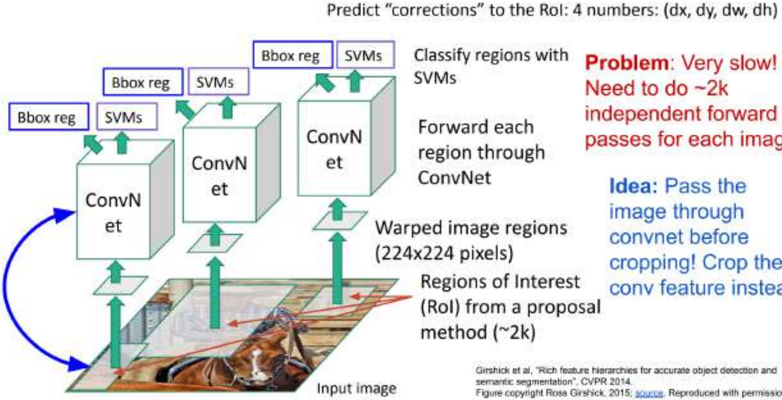
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.



Problem: Very slow! Need to do ~2k independent forward passes for each 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 (cont.)

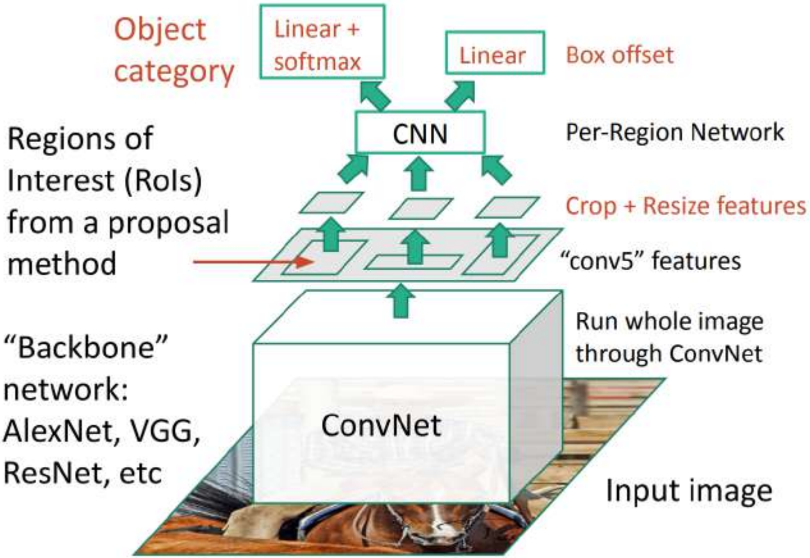


Problem: Very slow! Need to do ~2k independent forward passes for each image!

Idea: Pass the image through convnet before cropping! Crop the conv feature instead!

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

Fast R-CNN



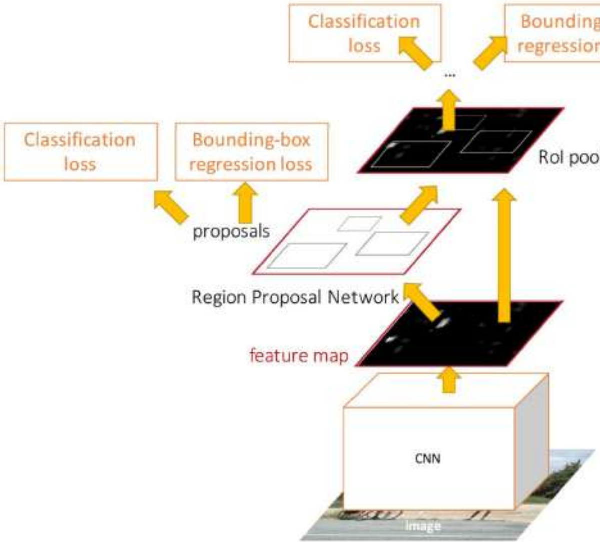
Girshick, "Fast R-CNN", ICCV 2015. Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

Faster R-CNN

- Make CNN do proposals!
- Insert Region Proposal Network (RPN) to predict proposals from features

- 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


Classification loss
Bounding-box regression loss
Classification loss
Bounding-box regression loss
RoI pooling
proposals
Region Proposal Network
feature map
CNN
image

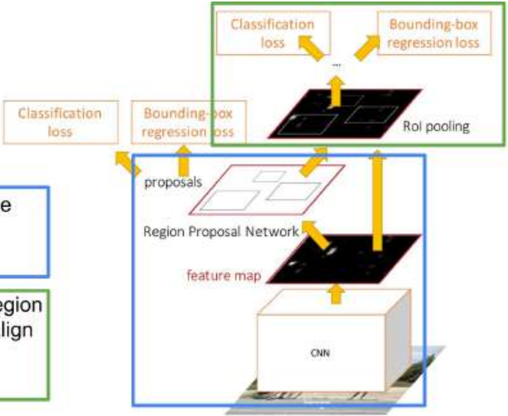
Faster R-CNN:

Make CNN do proposals!

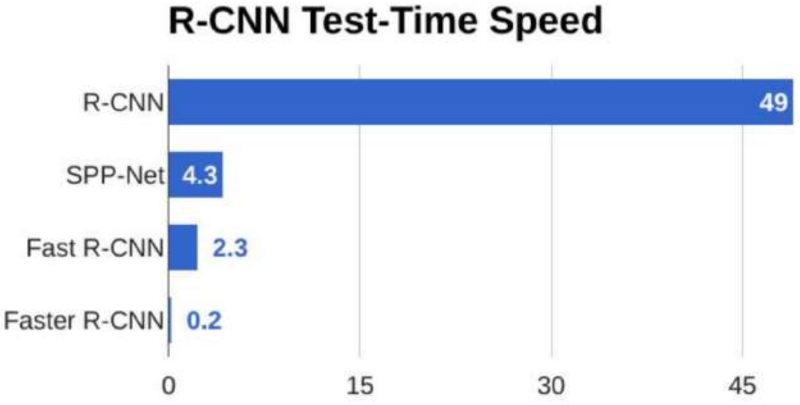
Faster R-CNN is a
Two-stage object detector

First stage: Run once per image
- Backbone network
- Region proposal network

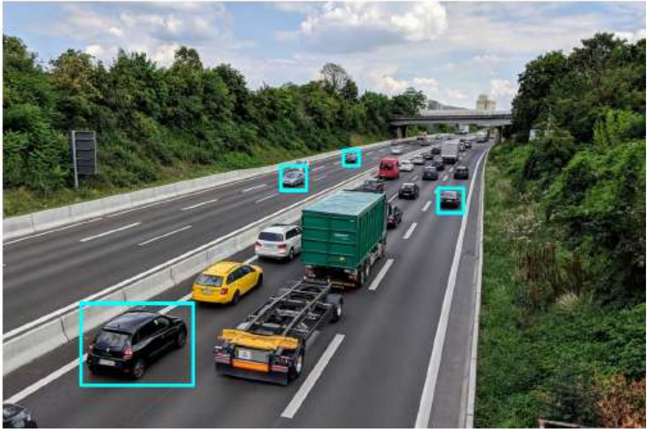
Second stage: Run once per region
- Crop features: RoI pool / align
- Predict object class
- Prediction bbox offset



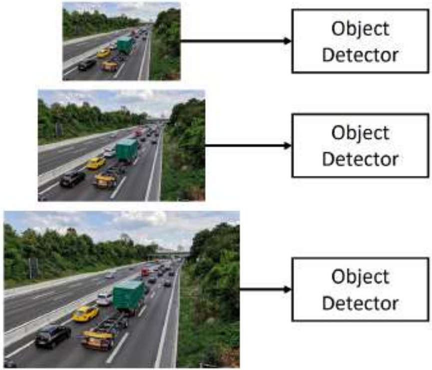
Faster R-CNN (cont.)



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



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

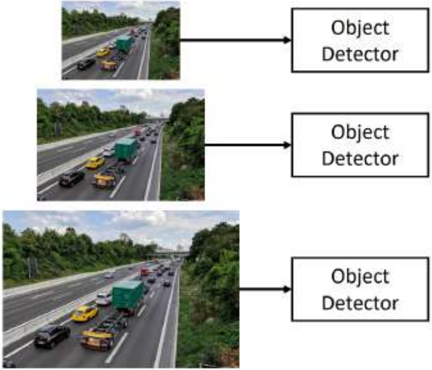


Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

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

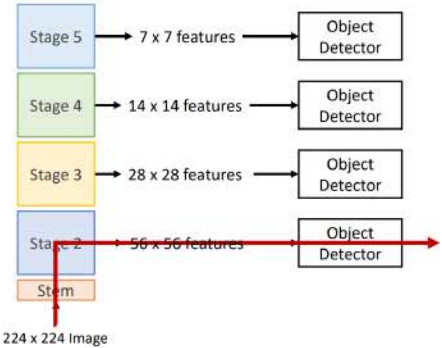
Problem: Expensive! Don't share any computation between scales

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

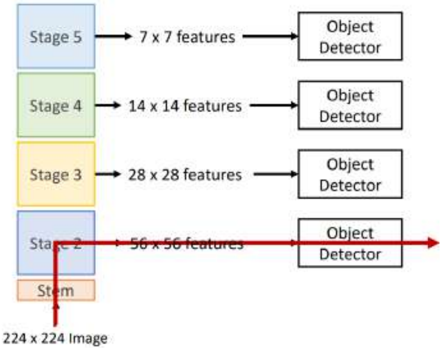


Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

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

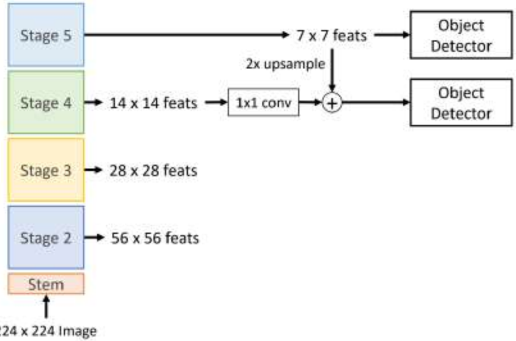
Problem: detector on early features doesn't make use of the entire backbone; doesn't get access to high-level features

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



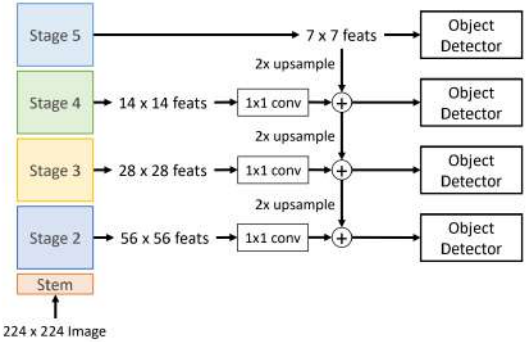
Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017

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

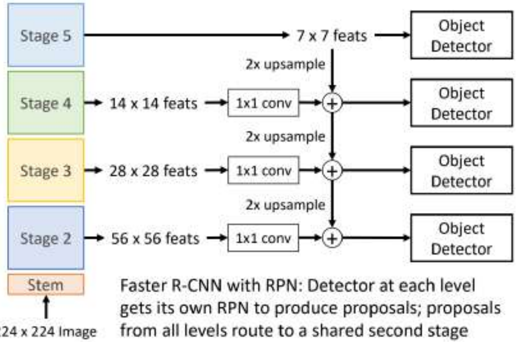
Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017



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

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

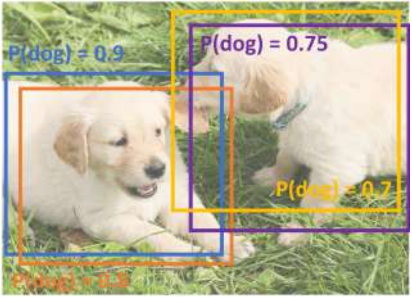
Lin et al, "Feature Pyramid Networks for Object Detection", ICCV 2017



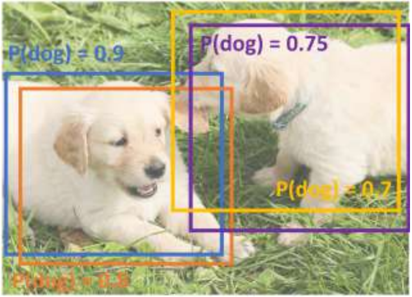
Faster R-CNN with RPN: Detector at each level gets its own RPN to produce proposals; proposals from all levels route to a shared second stage

Overlapping Boxes: Non-Max Suppression (NMS)

- **Problem:** Object detectors often output many overlapping detections

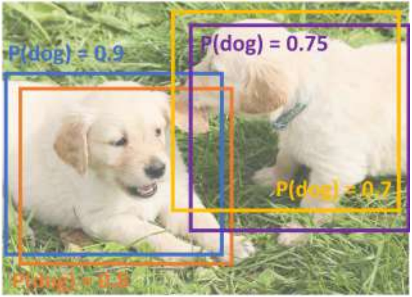


- **Problem:** Object detectors often output many overlapping detections
- **Solution:** Post-process raw detections using Non-Max Suppression (NMS)



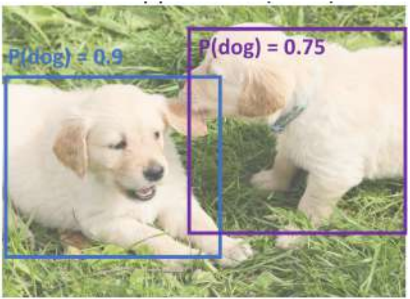
Overlapping Boxes: Non-Max Suppression (NMS)

- **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 IoU $>$ threshold (e.g. 0.7)
    4. If any boxes remain, GOTO 1



Overlapping Boxes: Non-Max Suppression (NMS)

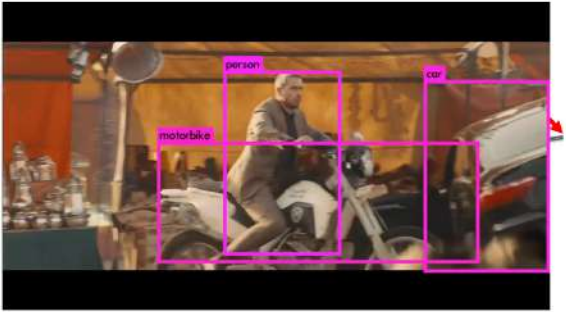
- **Problem:** Object detectors often output many overlapping detections
- **Solution:** Post-process raw detections using Non-Max Suppression (NMS)
  1. Select next highest-scoring box
  2. Eliminate lower-scoring boxes
  3. with IoU $>$ threshold (e.g. 0.7)
  4. If any boxes remain, GOTO 1



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 IoU $>$ 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 =(

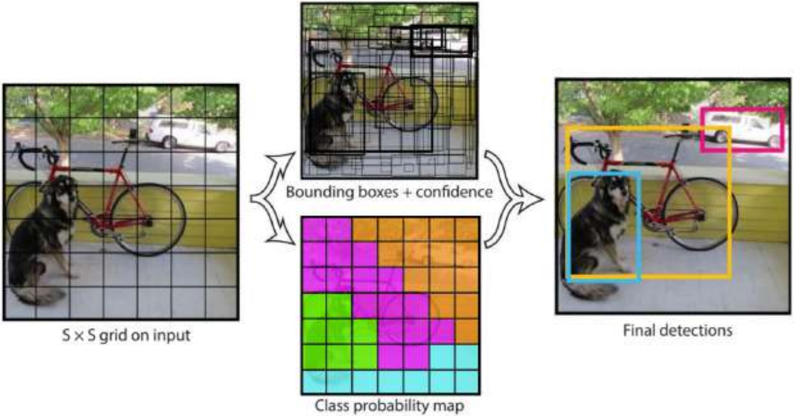
KAUST ACADEMY أكاديمية كاوست



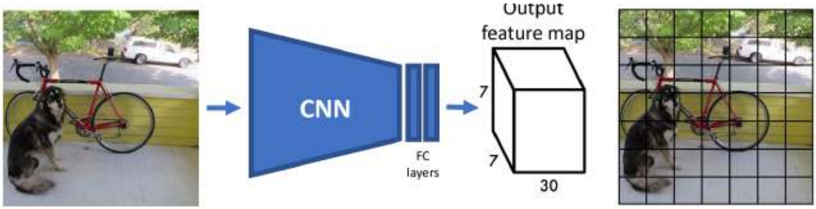
Single Shot:
SSD, YOLO ...

Fast
High false rate

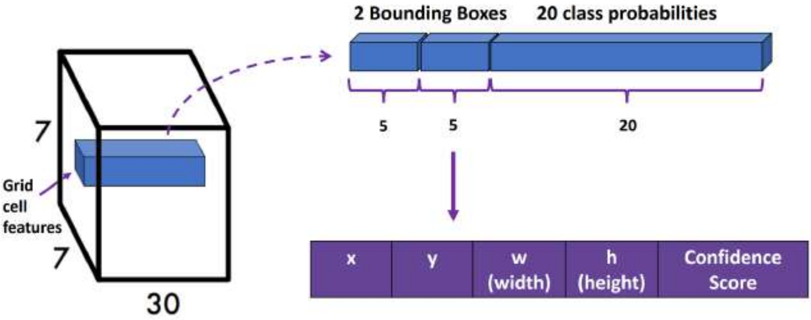
YOLO - Overview



YOLO - Overview



YOLO - Overview

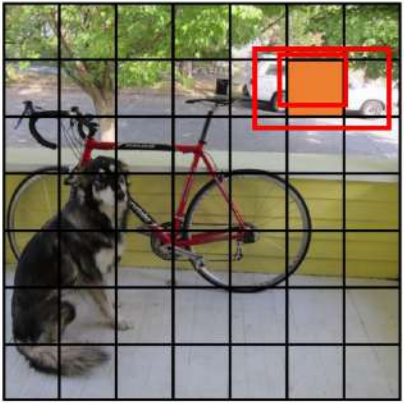


YOLO

Each cell predicts

Each cell predicts

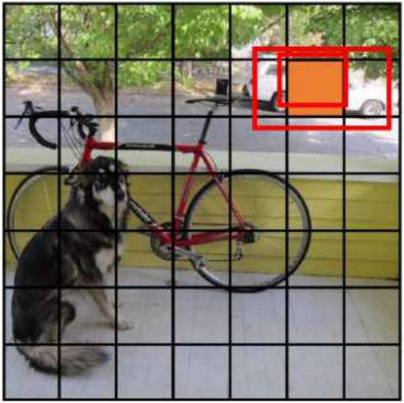
- $B = 2$ bounding boxes
  $(x, y, w, h) +$ confidence score



YOLO

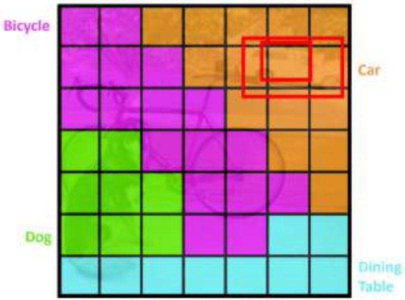
Each cell predicts

- $B = 2$ 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



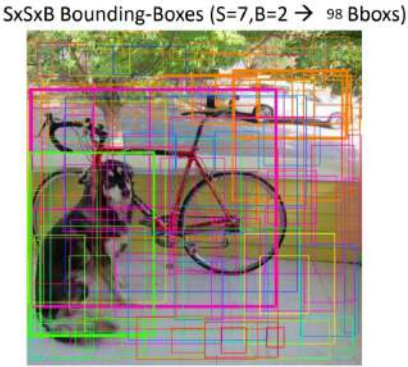
Class probability map

YOLO

Each cell predicts

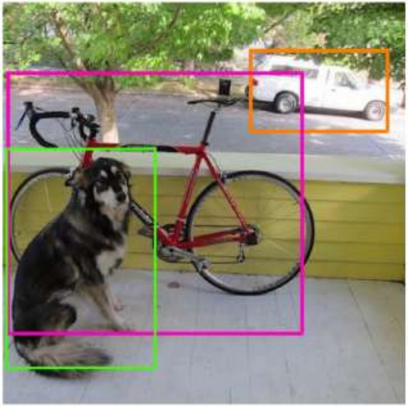
- $B = 2$ bounding boxes $(x, y, w, h) +$ confidence score
- $C = 20$ class probabilities

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



Each cell predicts

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

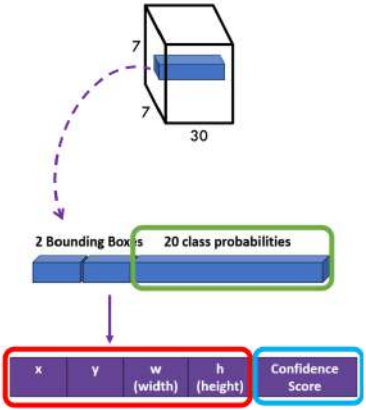


- Apply Non-Maximum Suppression

YOLO - Loss Function

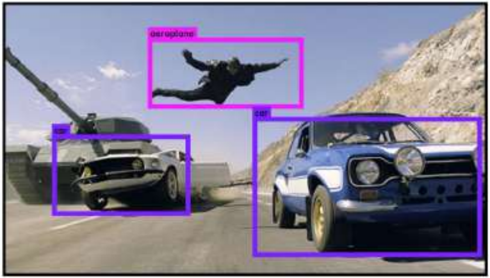
YOLO – Loss function

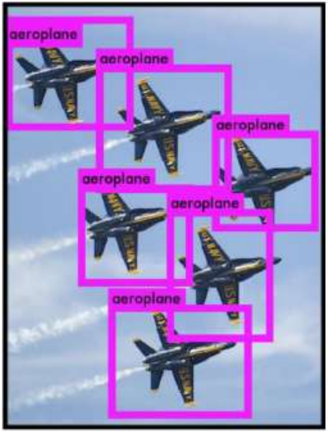
$$\mathcal{L} = \mathcal{L}_{\text{Localization Loss}} + \mathcal{L}_{\text{Confidence Loss}} + \mathcal{L}_{\text{Classification Loss}}$$



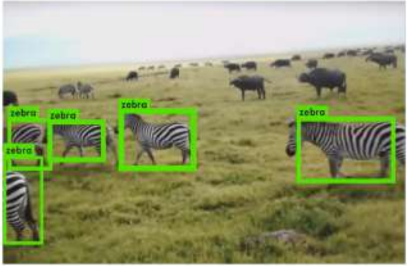
YOLO - Benefits

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



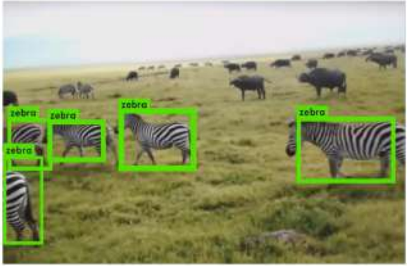


YOLO - Limitations



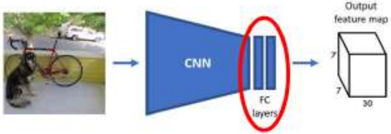
YOLO - Limitations

- Difficult to detect small objects
- Coarse predictions



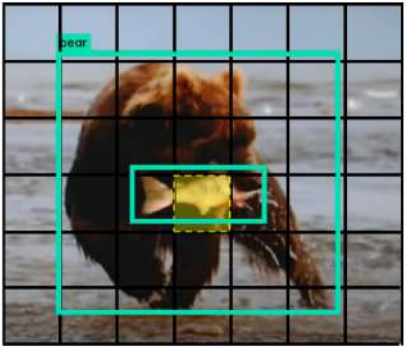
YOLO - Limitations

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



YOLO - Limitations

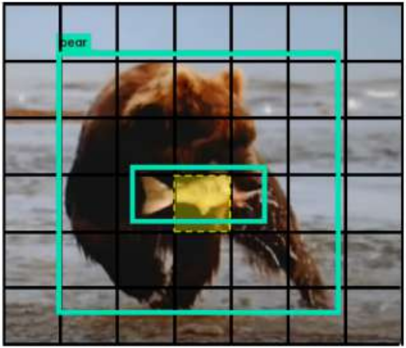
- Difficult to detect small objects
- Coarse predictions
- Fixed input size
- A grid cell can predict only one class



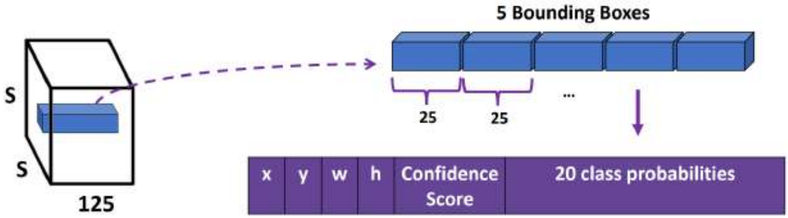
YOLO - Limitations

- 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

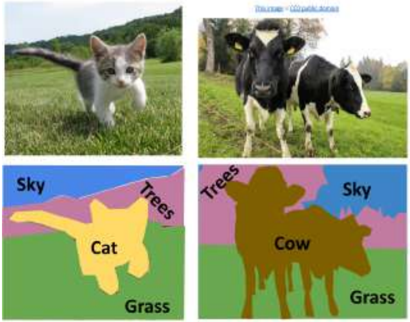


There's always room for improvement!

- 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)

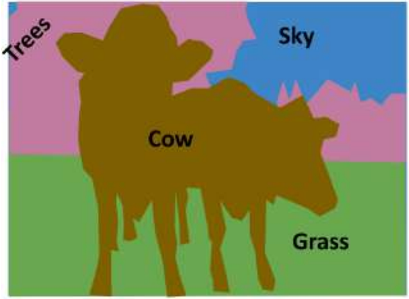
Things and Stuff

- **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)



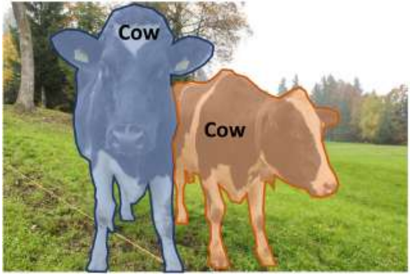
- **Object Detection:** Detects individual object instances, but only gives box(Only things!)

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

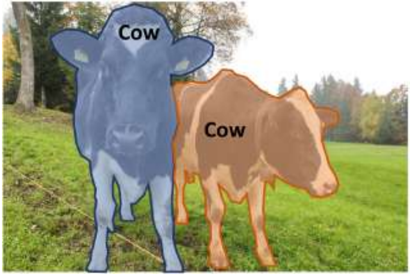


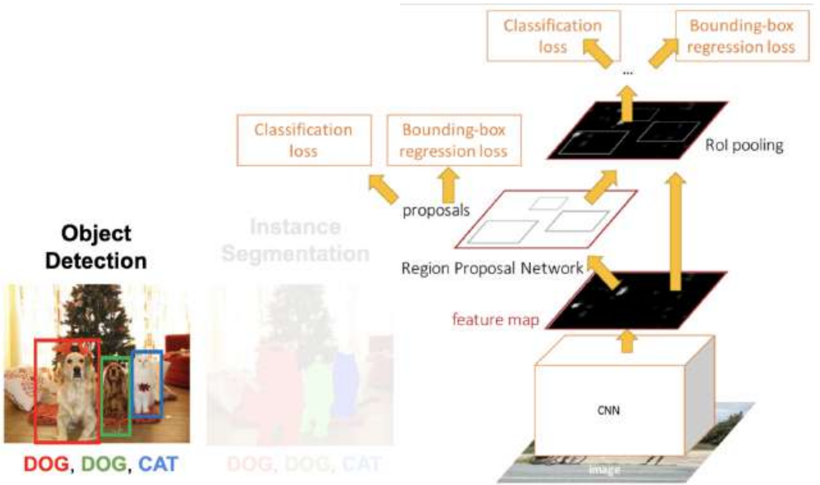
Instance Segmentation

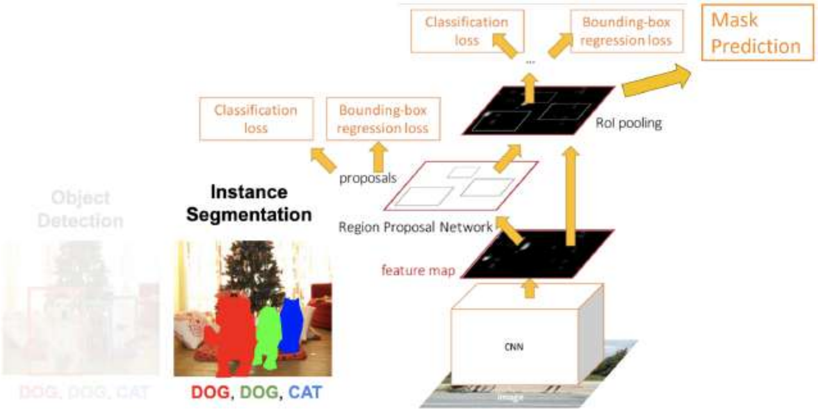
- Detect all objects in the image, and identify the pixels that belong to each object (Only things!)



- 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!

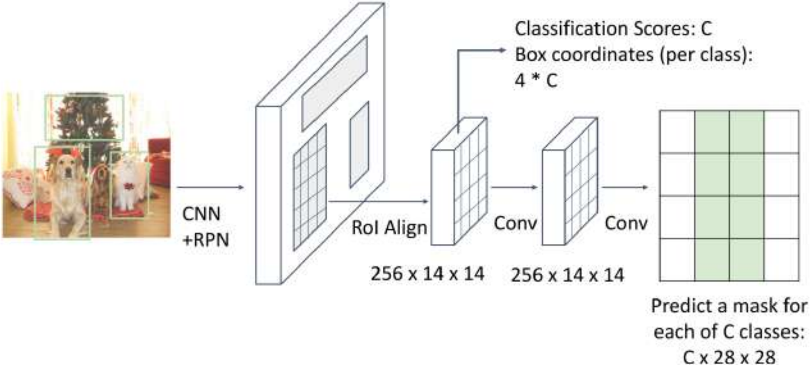



Object Detection
DOG, DOG, CAT
Instance Segmentation
DOG, DOG, CAT
Classification loss
Bounding-box regression loss
Classification loss
Bounding-box regression loss
RoI pooling
proposals
Region Proposal Network
feature map
CNN
image

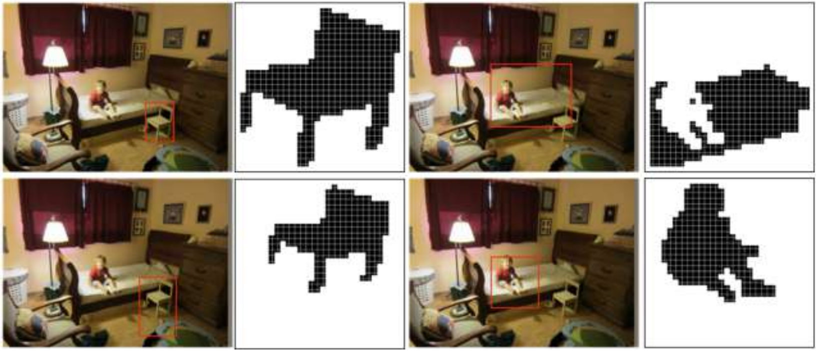


[0] He et al, "Mask R-CNN", ICCV 2017

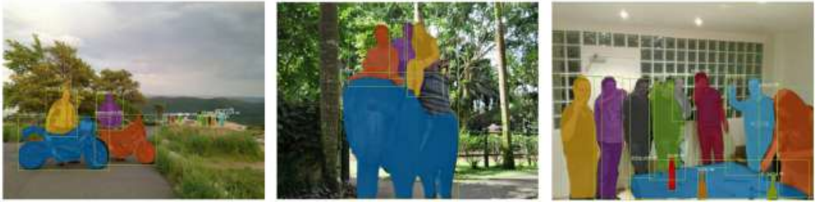
Instance Segmentation: Mask R-CNN



Mask R-CNN: Example Training Targets

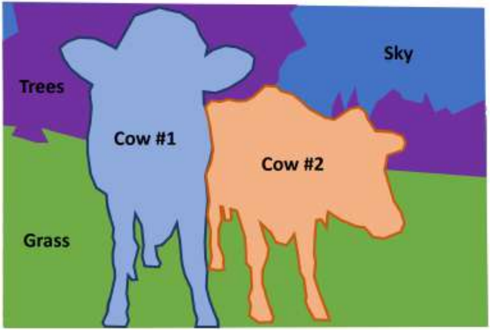


Mask R-CNN: Very Good Results!



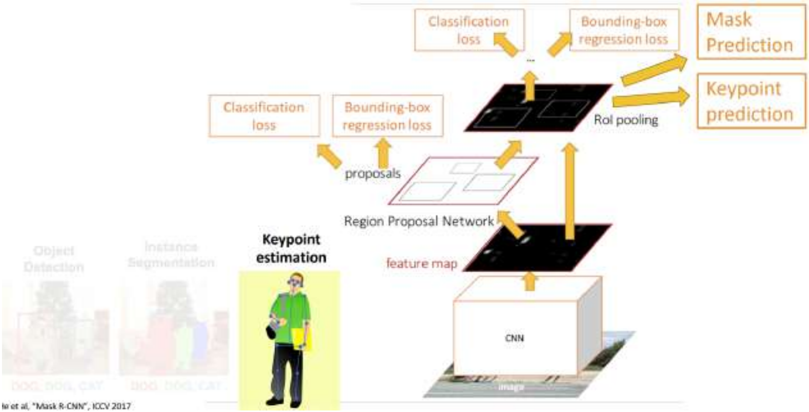
Beyond Instance Segmentation: Panoptic Segmentation

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

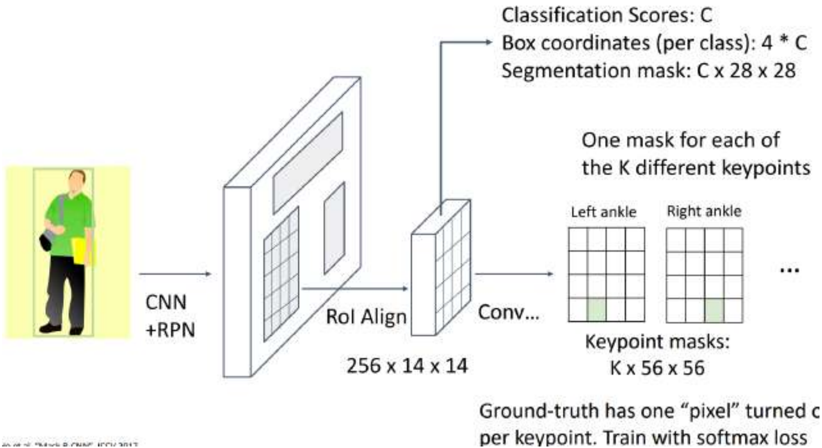


[0]Kirillov et al, "Panoptic Feature Pyramid Networks", CVPR 2019

- 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


Classification
loss
Bounding-box
regression loss
Mask
Prediction
Keypoint
prediction
Classification
loss
Bounding-box
regression loss
RoI pooling
proposals
Region Proposal Network
feature map
CNN
image
Object
Detection
Instance
Segmentation
Keypoint
estimation
He et al, "Mask R-CNN", ICCV 2017

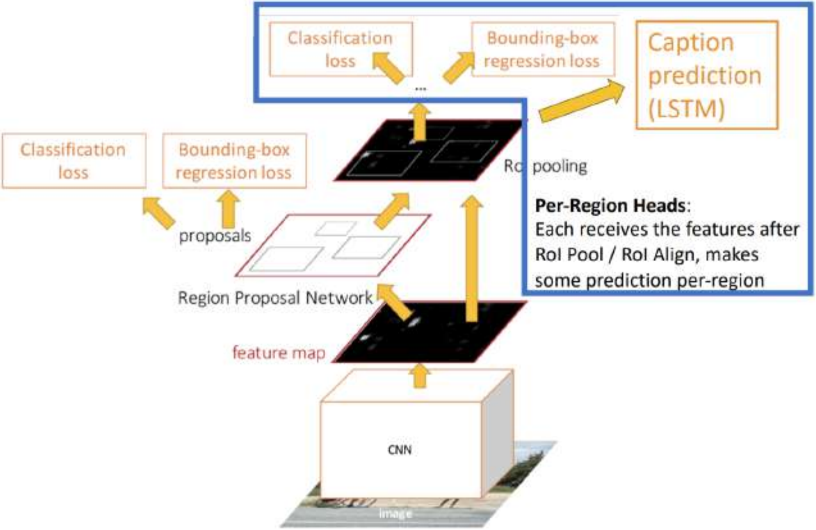
Mask R-CNN: Keypoint Estimation



Joint Instance Segmentation and Pose Estimation

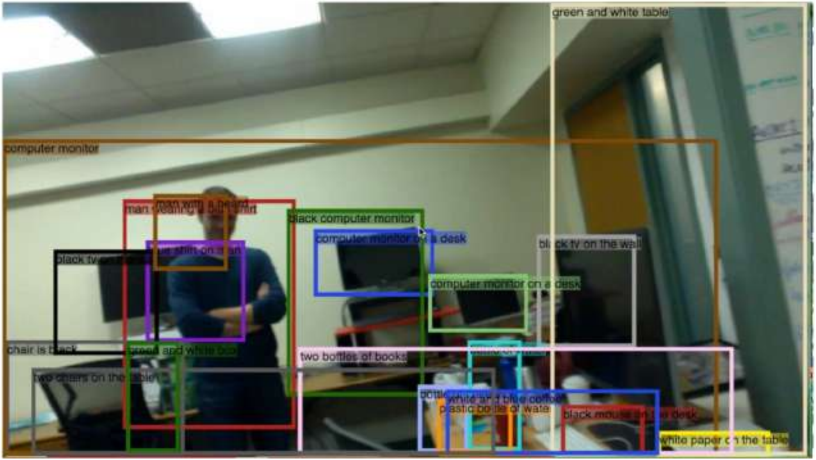
[0] He et al, "Mask R-CNN", ICCV 2017

Captioning: Predict a caption per region!



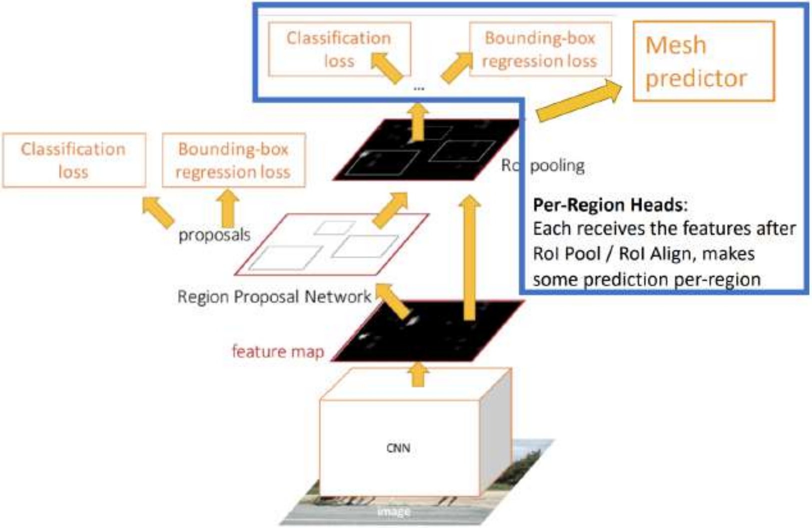
Per-Region Heads:
Each receives the features after RoI Pool / RoI Align, makes some prediction per-region

Captioning: Predict a caption per region!



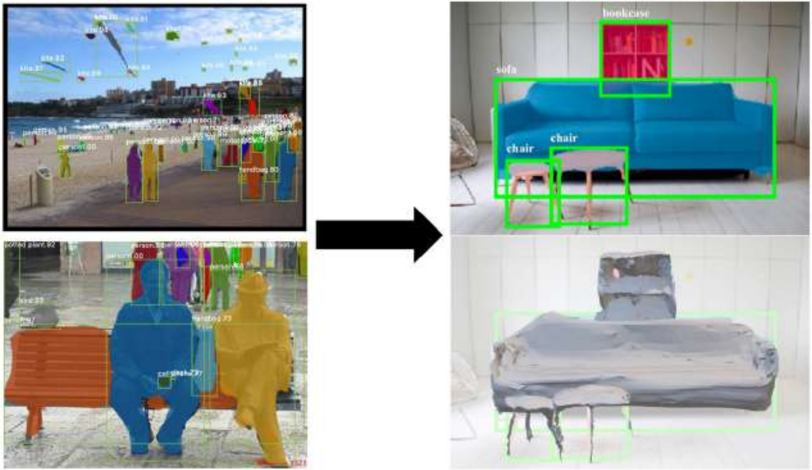
Johnson, Karpathy, and Fei-Fei, "DenseCap: Fully Convolutional Localization Networks for Dense Captioning", CVPR 2016

KAUST ACADEMY



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

Mesh R-CNN:
2D Image -> **3D** shapes



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

Object Tracking

- **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

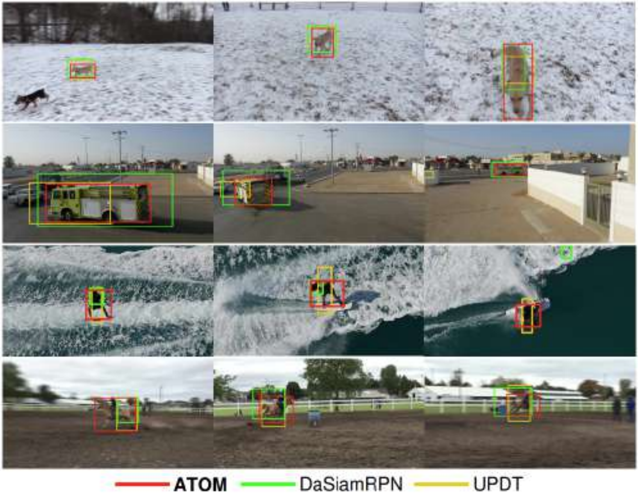


Figure 2: Comparison of 3 approaches for object tracking

Danelljan et al.

References

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