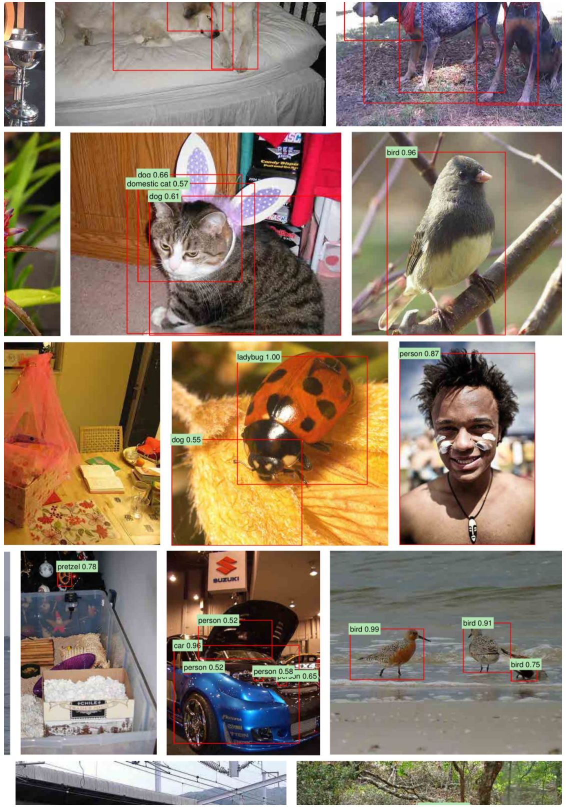
Object Detection

Deep Learning

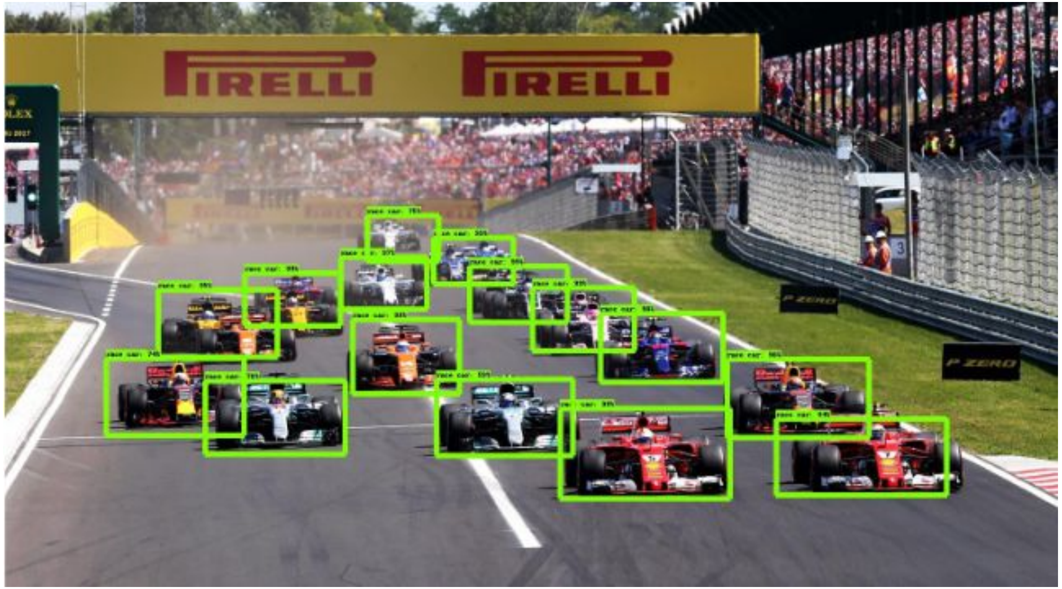
Aziz Temirkhanov
Lambda, HSE

Object Detection



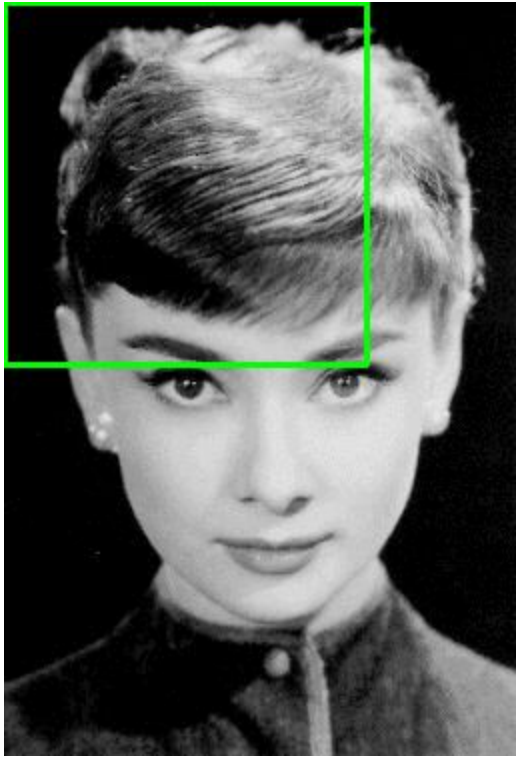
Object Detection

- **Goal**: find objects in the image and localize them with bounding boxes (BB)
- Variable number of objects in each image
- Intra-class variation of objects
- Imbalanced / rare classes
- Efficiency (FPS)



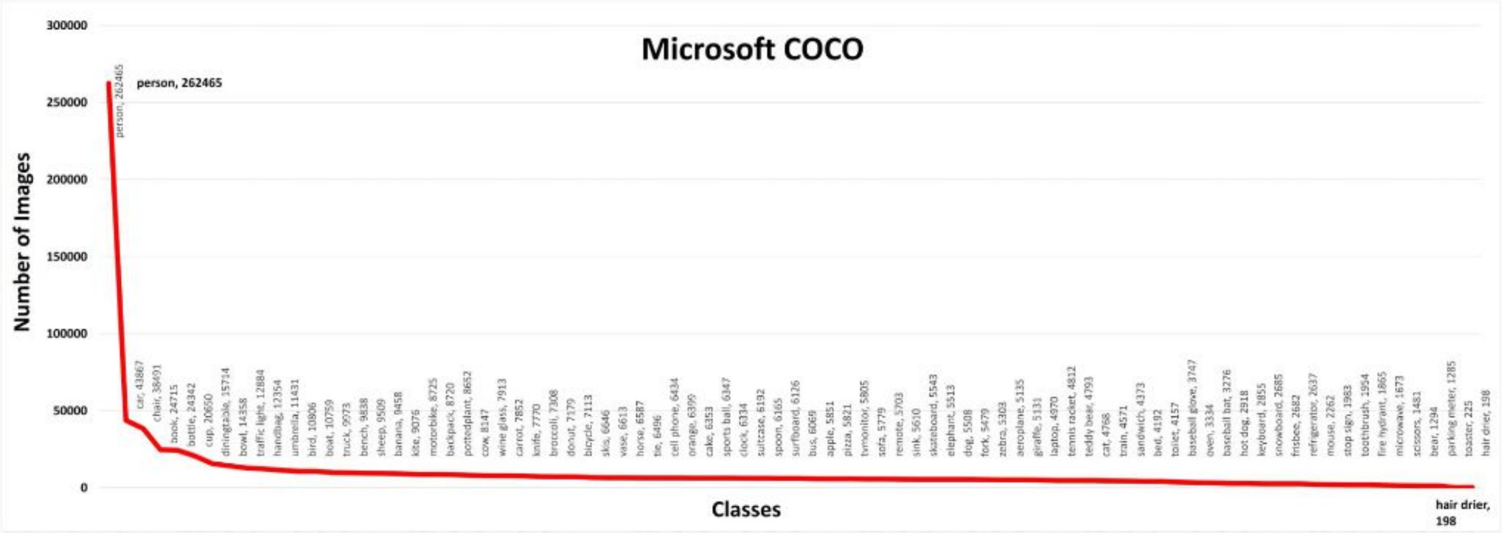
Sliding Window

- Classify object and find coordinate of it
- Let’s crop some region within the image and classify it
- The region with the highest probability of containing the object is the region we are looking for!
- In order to find best possible BB, iterate not only through the image, but also through window’ hyperparams (e.g. window size)



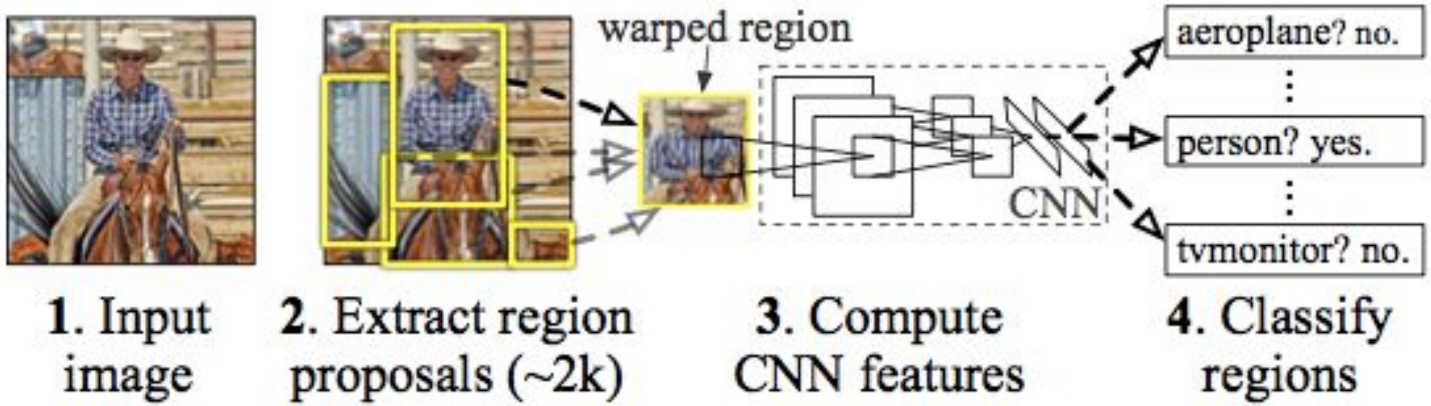
Datasets

- Pascal VOC (Visual Object Classes), 20 classes
- MS COCO (Microsoft Common Objects in Context), 91 classes
- ImageNet, 200 classes



Two-Stage Detectors

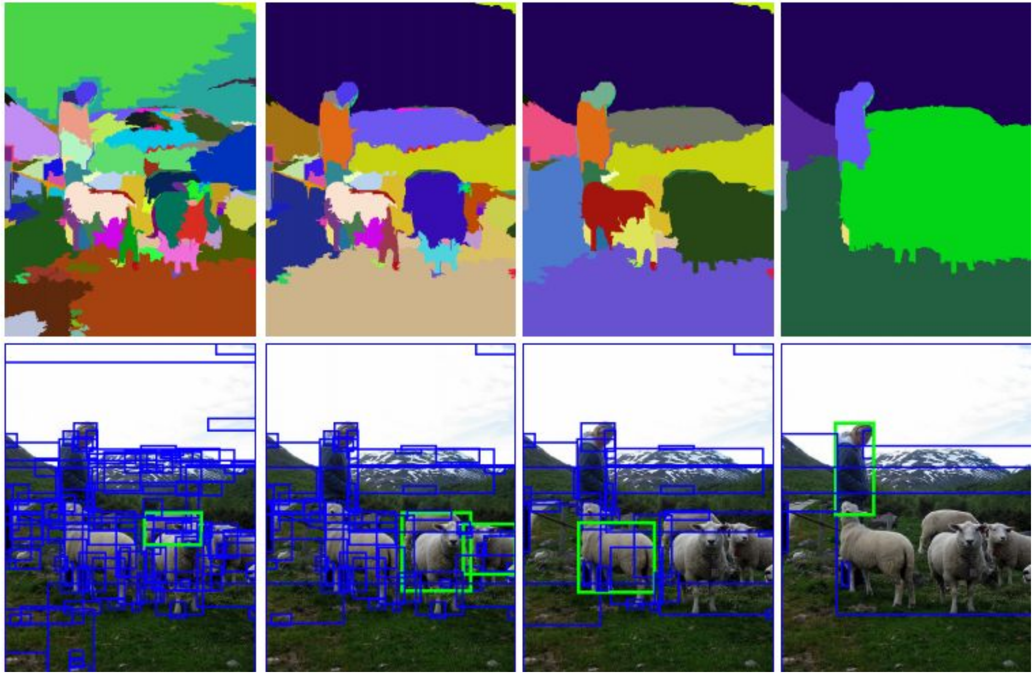
- Find regions with high probability of containing object
- Adjust BB in the second stage



R-CNN

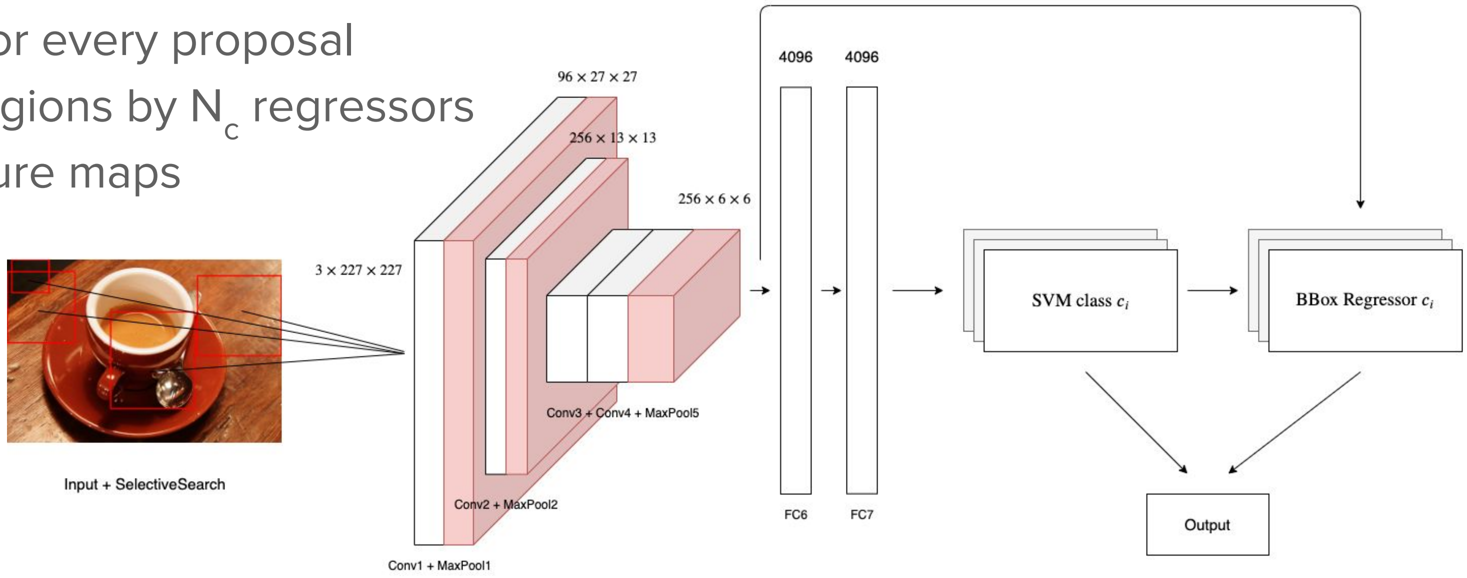
- Propose regions by Selective Search
  - Based on pixel intensity and using a graph-based algorithm, over-segment pixels
  - Group similar regions together
- Classify proposals and extract features using CNN
- Adjust coordinates
- Repeat for all proposals

Girshick et. al

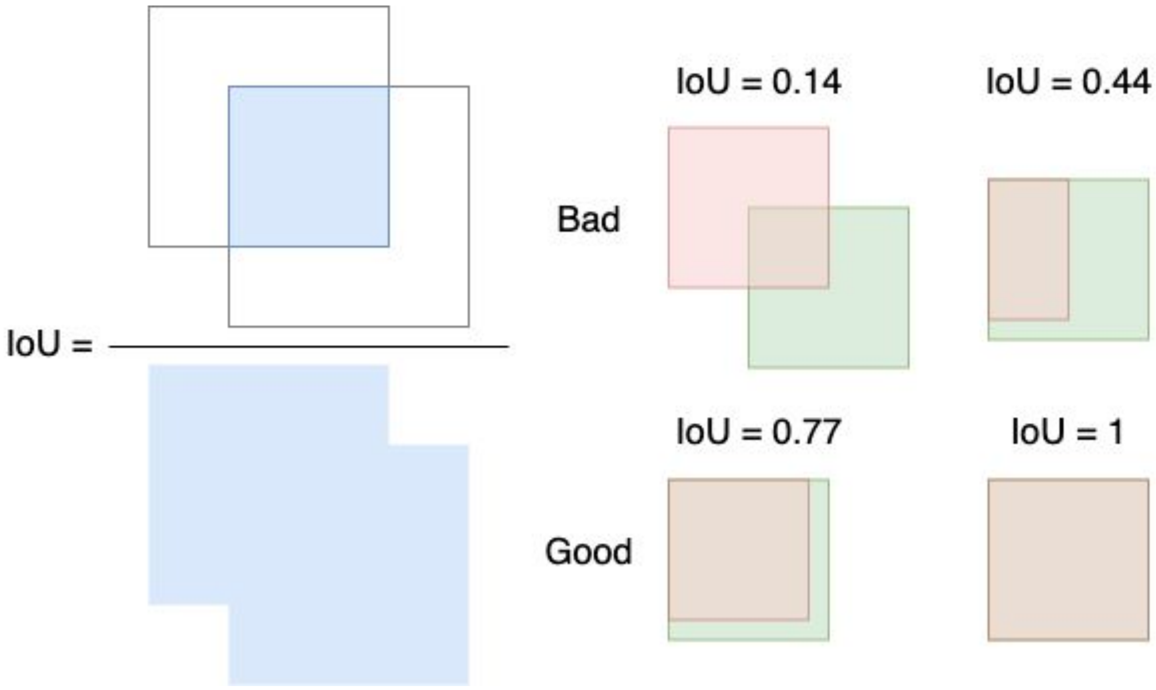


R-CNN

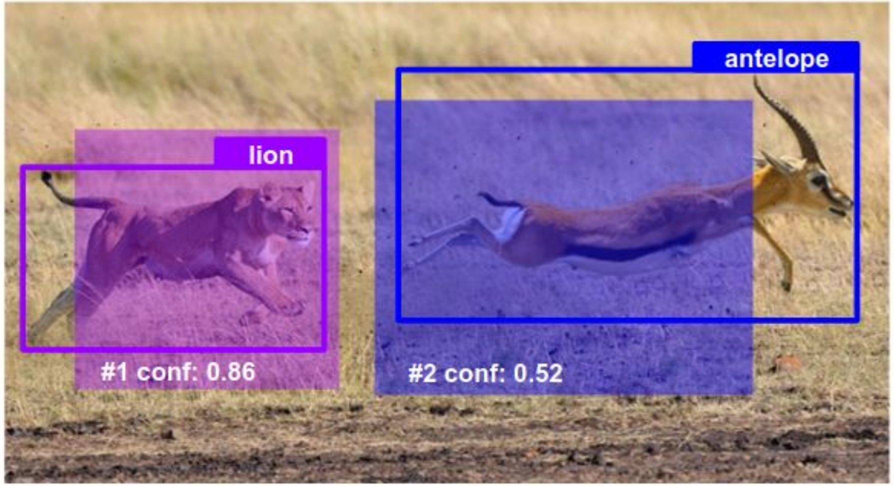
- Extract a vector from last layer of CNN (AlexNet in original work)
- $N_c + 1$ of total SVM classifiers solving one-vs-rest task
- Repeat for every proposal
- Adjust regions by N_c regressors
- Use feature maps

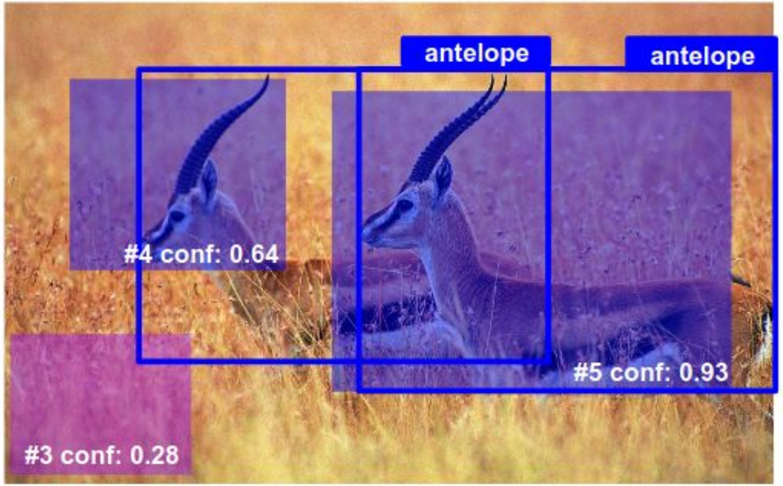


IoU



mAP





Class: **antelope**

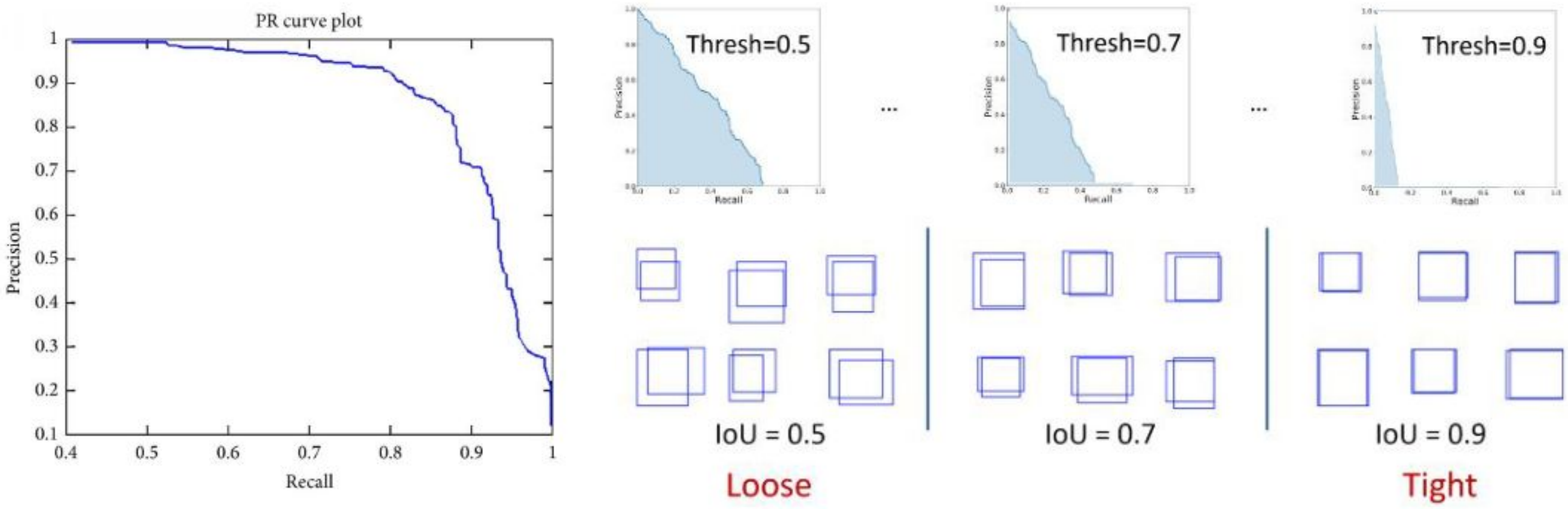
BB	conf	IoU (thr = 0.5)	TP/FP	precision	recall
#5	0.93	0.74	TP	1	0.33
#4	0.64	0.15	FP	0.5	0.33
#2	0.52	0.54	TP	0.66	0.66

Class: **lion**

BB	conf	IoU (thr = 0.5)	TP/FP	precision	recall
#1	0.86	0.68	TP	1	1
#3	0.28	0.0	FP	0.5	1

mAP

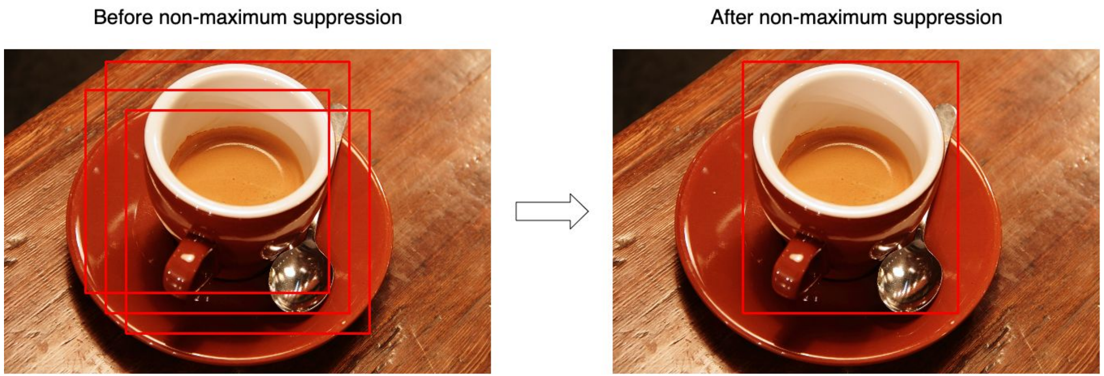
- **AP** (average precision) = area under PR-curve
- **mAP** (mean AP) = AP averaged over all classes
- Different variants depending on IoU threshold: mAP_{50}, mAP_{75}, $mAP_{[50:95]}$



Non Maximum Suppression

1. Take a set of BBoxes and sort them by certainty score
2. Select BBox with highest score and add it to final list of BBoxes, remove it from original list
3. Take next BBox in original list and compare it with BBoxes in selected list
4. If their IoU is higher than threshold remove second BBox

```
function nms(hypotheses, threshold):
    sorted = sort(hypotheses.values, key=hypotheses.scores)
    result = []
    for first in sorted:
        result.join(first)
        without_first = sorted / first
        for second in without_first:
            if IoU(first, second) > threshold:
                sorted.remove(second)
    return result
```



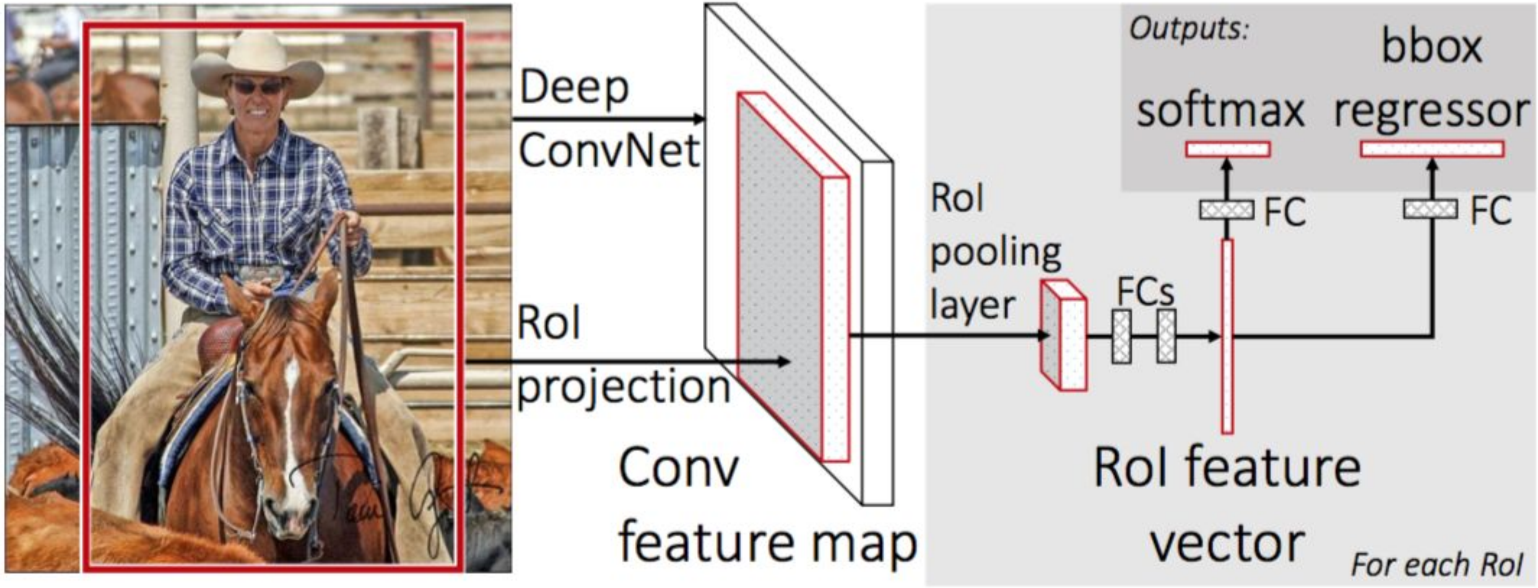
Hard Negative Mining

- Positive hypothesis — ones that contains an object
- Negative hypothesis — ones that contains a background or a part of an object
- Penalty classifier for false positive hp
- How to deal with Negative hypothesis?
  - Easy Negative — background
  - Hard Negative — wrong class or partial right class

Hard Negative Mining

- Compare true BB with proposed one, using IoU
- Low IoU — easy negative
- High IoU — hard negative or positive
- Explicitly find those hard negative examples and add them to training data

Fast R-CNN

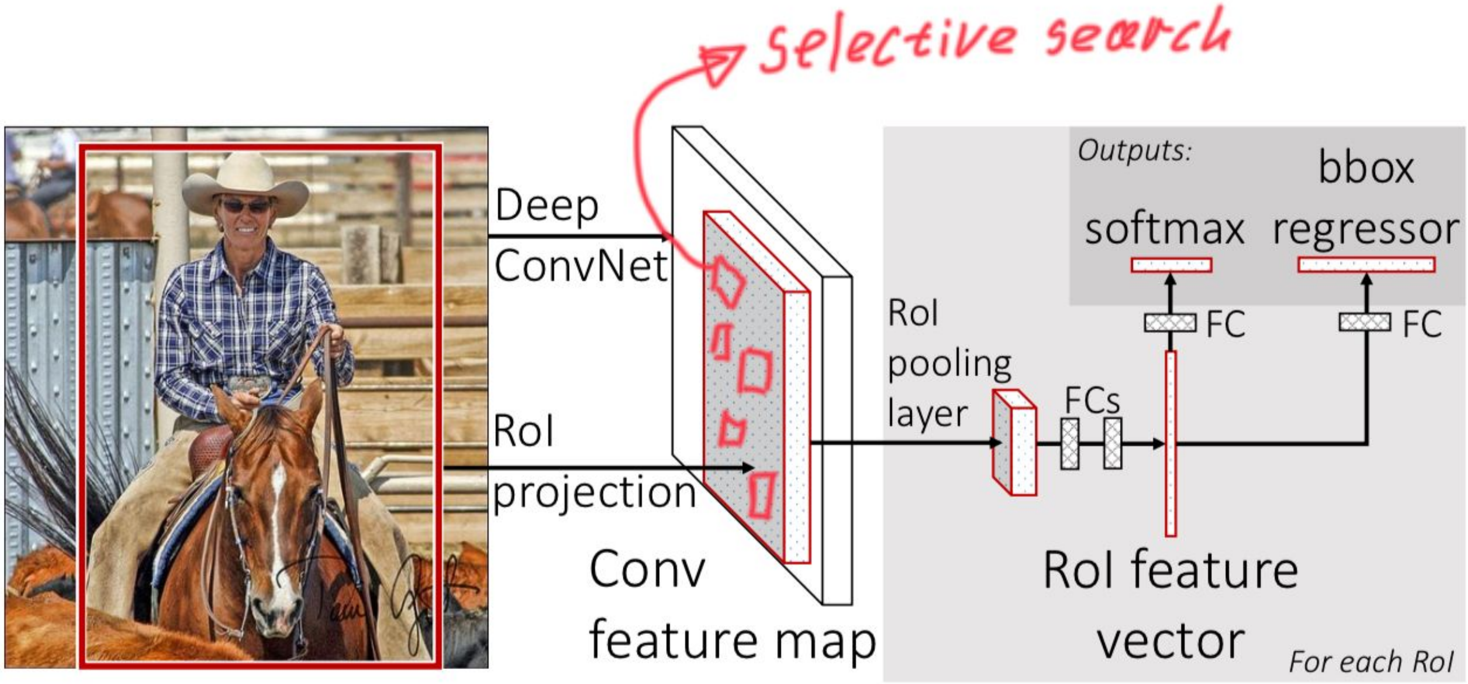


Deep
ConvNet
RoI
projection
Conv
feature map
RoI
pooling
layer
FCs
RoI feature
vector
For each RoI
Outputs:
softmax
bbox
regressor
FC
FC

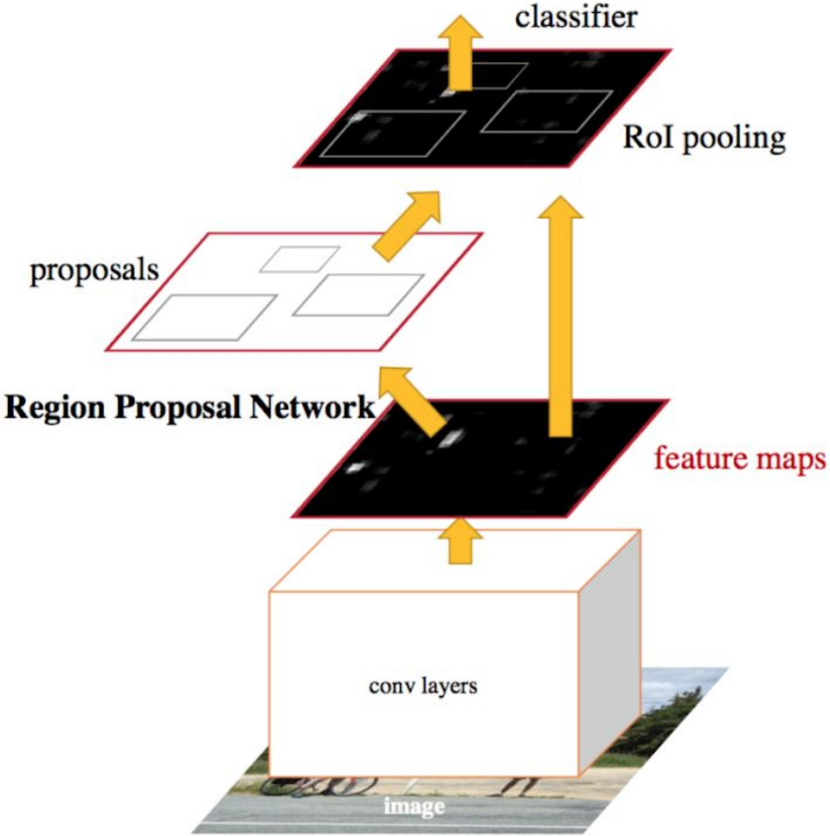
Fast R-CNN

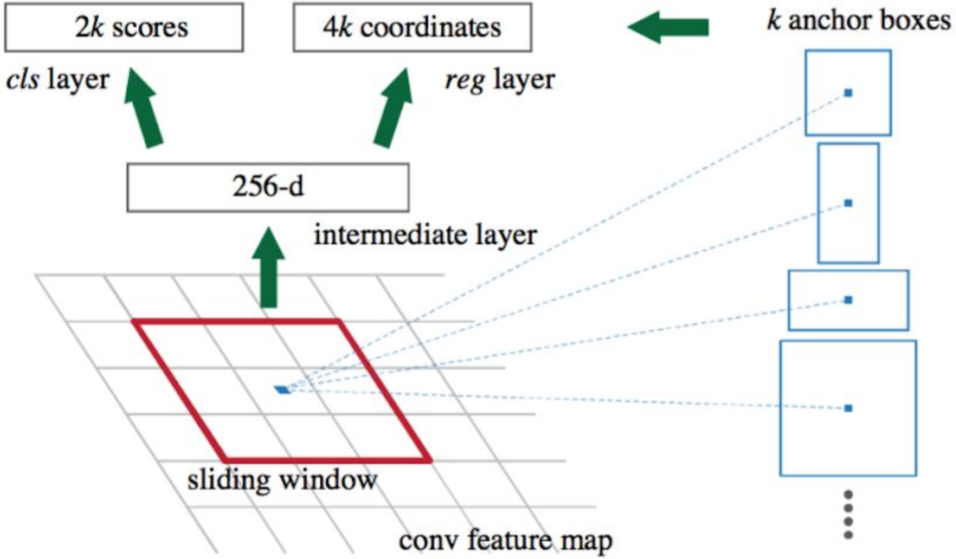
1. Feed image to a CNN block and obtain Feature Map
2. Run a Selective Search on the image and obtain proposals
3. Project proposal region onto feature map
4. Employ RoI pooling layer: adjust size of the region to be the same, and then maxpool the same-size filters

Fast R-CNN


selective search
Deep
ConvNet
RoI
projection
Conv
feature map
RoI
pooling
layer
FCs
Outputs:
softmax
bbox
regressor
FC
FC
RoI feature
vector
For each RoI

Faster R-CNN


classifier
RoI pooling
proposals
Region Proposal Network
feature maps
conv layers
image


2k scores
4k coordinates
k anchor boxes
cls layer
reg layer
256-d
intermediate layer
sliding window
conv feature map

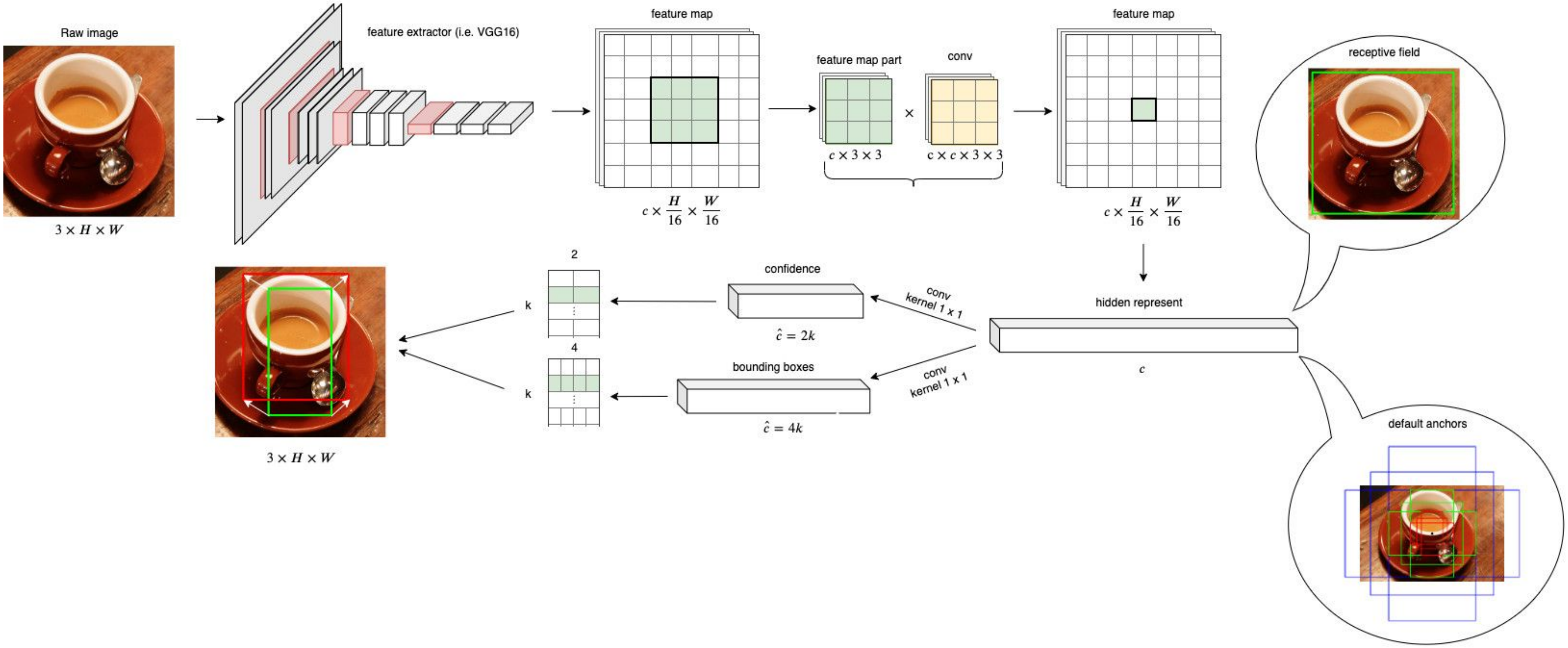
Faster R-CNN

1. Pre-train a CNN network on image classification tasks.
2. Fine-tune the RPN (region proposal network) end-to-end for the region proposal task, which is initialized by the pre-train image classifier. Positive samples have IoU (intersection-over-union) > 0.7, while negative samples have IoU < 0.3.
    a. Slide a small n x n spatial window over the conv feature map of the entire image.
    b. At the center of each sliding window, we predict multiple regions of various scales and ratios simultaneously. An anchor is a combination of (sliding window center, scale, ratio). For example, 3 scales + 3 ratios => k=9 anchors at each sliding position.
3. Train a Fast R-CNN object detection model using the proposals generated by the current RPN
4. Then use the Fast R-CNN network to initialize RPN training. While keeping the shared convolutional layers, only fine-tune the RPN-specific layers. At this stage, RPN and the detection network have shared convolutional layers!
5. Finally fine-tune the unique layers of Fast R-CNN
6. Step 4-5 can be repeated to train RPN and Fast R-CNN alternatively if needed.

Region Proposal Network

- Using VGG16’ last conv layer — conv5_3:
    - Effective stride: 16
    - Receptive Field size: 196
    - number of channels (feature maps): 512
- Take this feature maps and propose k hypothesis (k=9 at the beggining) for different sizes and aspect ratio. For default size it 14x14x9=1764 hypothesis
- Take $c\frac{H}{16}\frac{W}{16}$ feature map and apply 3x3 convolution layer with stride=1
- Apply two 1x1 convolutions simultaneously to the feature map
    - cls layer translates 512 filters to 2k filters (binary classification for objects)
    - reg layer translates 512 filters to 4k filters: coordinates for each hypothesis
- Employ anchors: a defaults BBoxes with 3 different sizes (128x128, 256x256, 512x512) and 3 different aspect ratios (1:1, 2:1, 1:2), so 9 anchors in total.
- Now adjust this anchors with reg layer

Faster R-CNN


Raw image
$3 \times H \times W$
feature extractor (i.e. VGG16)
feature map
$c \times \frac{H}{16} \times \frac{W}{16}$
feature map part
conv
×
$c \times 3 \times 3$
$c \times c \times 3 \times 3$
feature map
$c \times \frac{H}{16} \times \frac{W}{16}$
receptive field
hidden represent
c
conv
kernel 1 x 1
confidence
$\hat{c} = 2k$
bounding boxes
$\hat{c} = 4k$
2
4
k
k
$3 \times H \times W$
default anchors

Faster R-CNN

1. Initialize RPN with CNN layer and generate current proposals
2. Train Fast R-CNN model using current proposals. Initialize weights with CNN
3. Train RPN network again using Fast R-CNN weights. At this stage, RPN and Fast R-CNN network shares the same layers, so freeze those layers and train only RPN-specific ones
4. Fine-tune Fast R-CNN network
5. Steps 3 and 4 can be repeated if needed

One-Stage Detectors

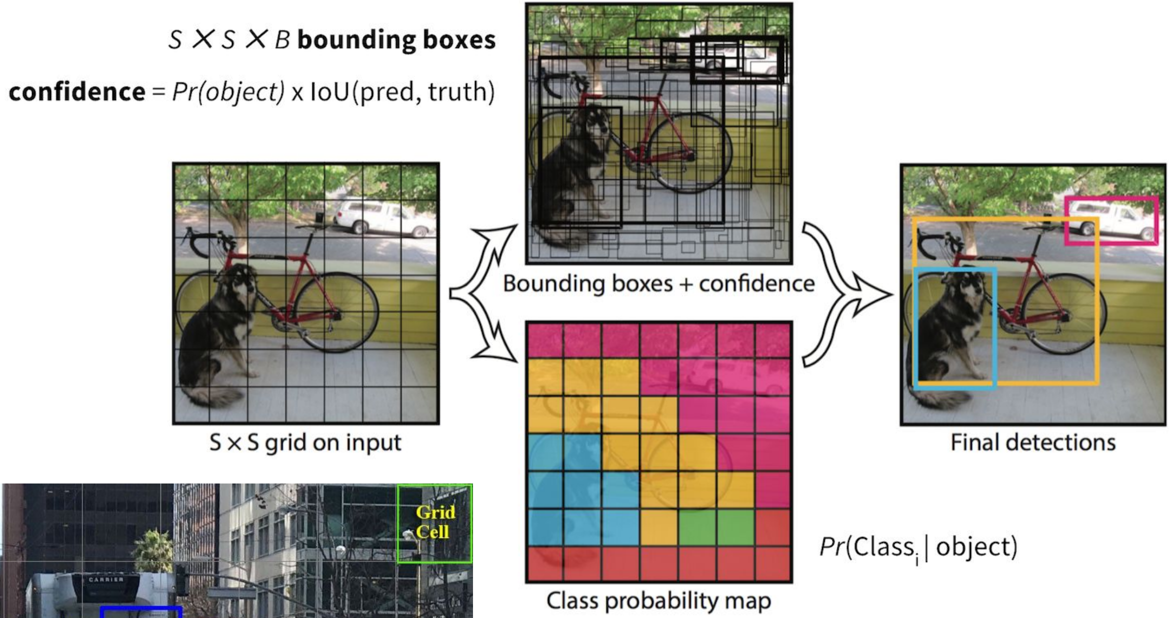
Bonus part

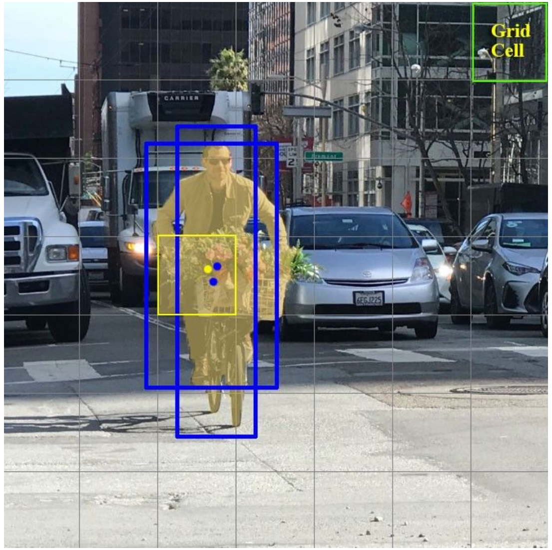
Two-stage and one-stage detectors

- Family of R-CNN models are all two-stage detectors, which can be slow
- One-Stage detectors runs once over dense sampling of possible locations

YOLO

1. Pre-train the CNN model
2. Split image into SxS cells
3. Assign B bounding boxes to each cell. Initial size is set via K-means and IoU throughout true bboxes
4. Predict 4 coordinates: x-center, y-center, h, w
5. Predict probability of containing object
6. Predict K conditional probabilities





YOLO

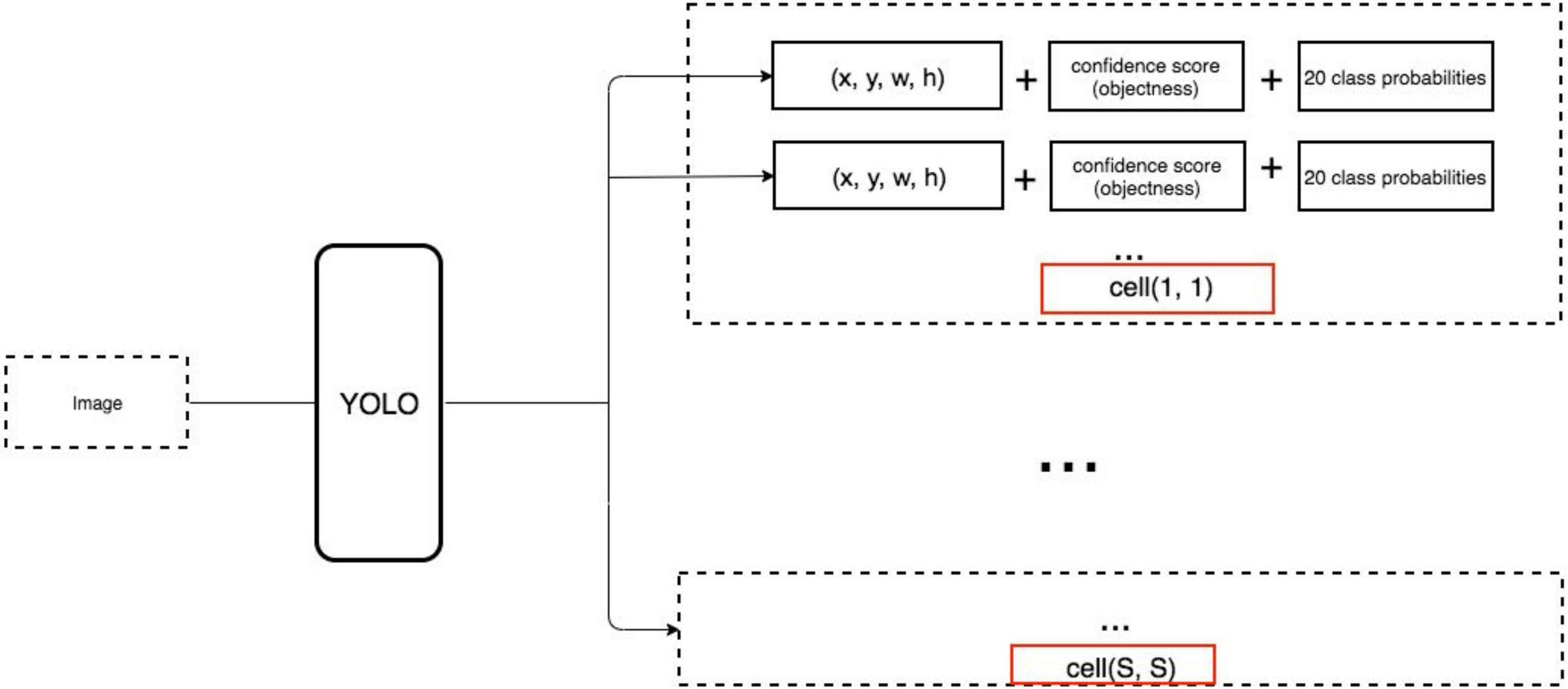
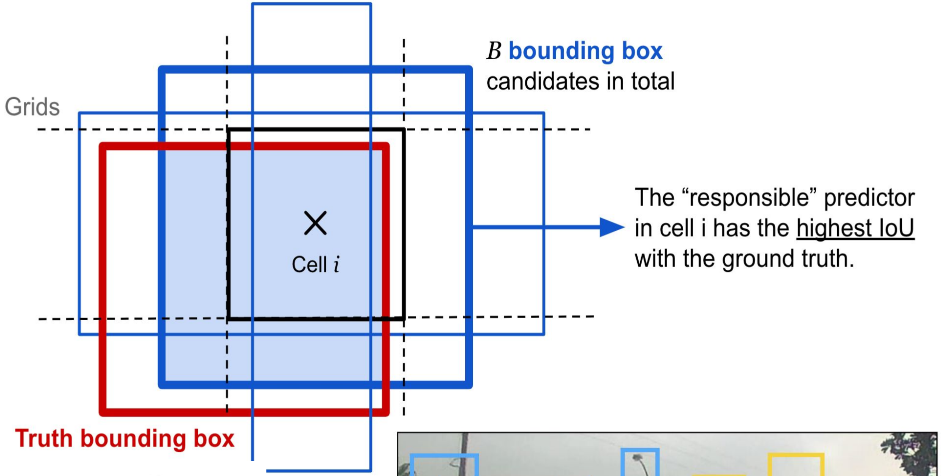


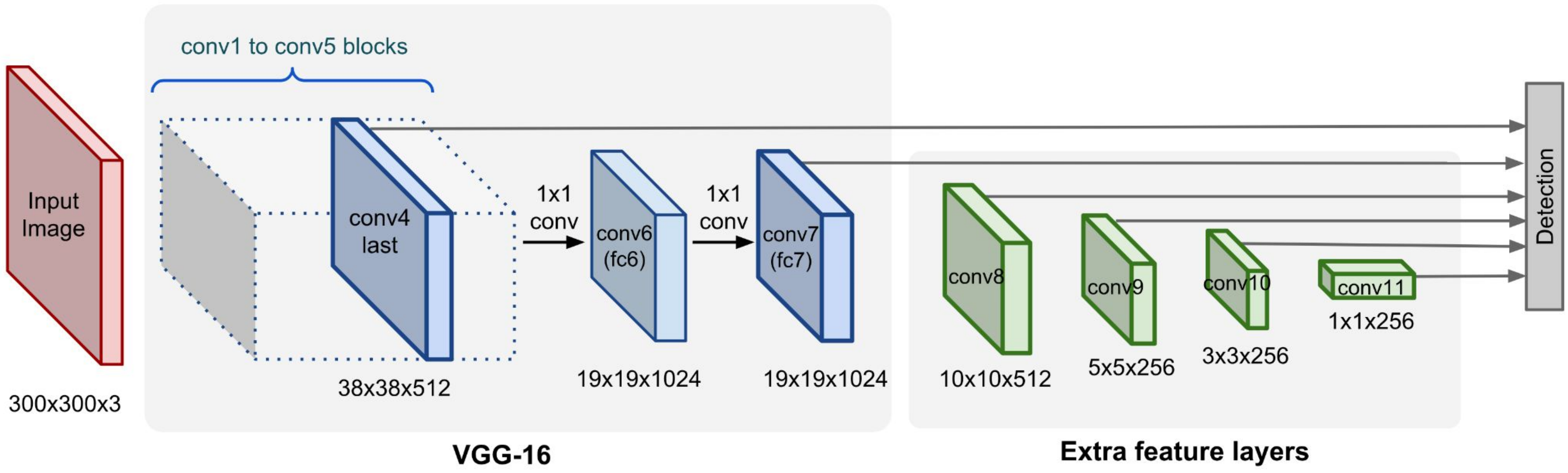
Image
YOLO
(x, y, w, h)
+
confidence score
(objectness)
+
20 class probabilities
(x, y, w, h)
+
confidence score
(objectness)
+
20 class probabilities
...
cell(1, 1)
...
...
cell(S, S)

YOLO

- Set a threshold beforehand in terms of IoU
- If proposed BBoxes has IoU > thresh leave this BB and remove other ones



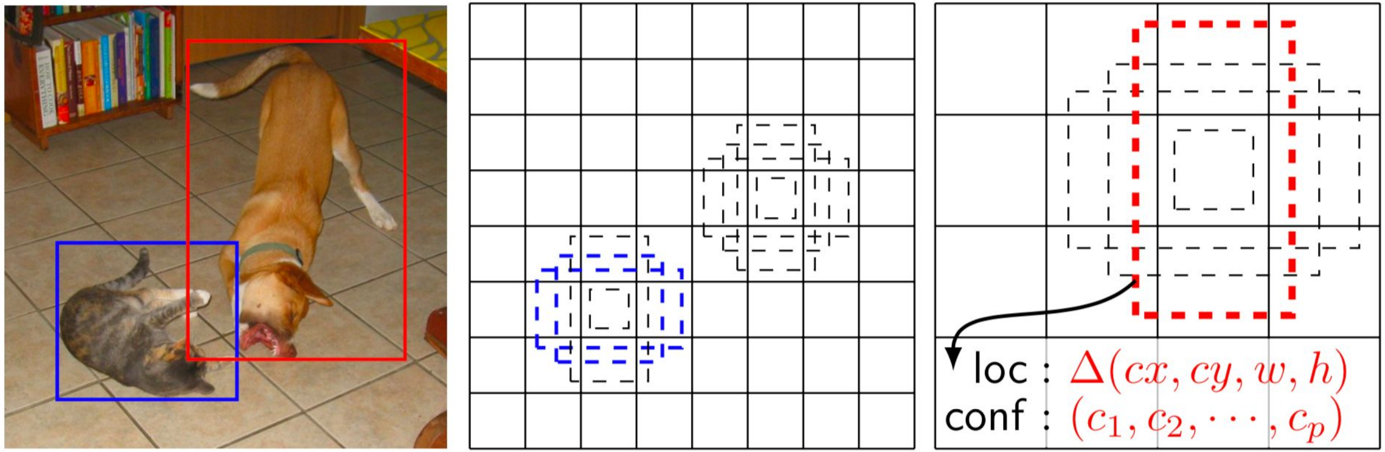
SSD



conv1 to conv5 blocks
Input Image
300x300x3
conv4 last
38x38x512
1x1 conv
conv6 (fc6)
19x19x1024
1x1 conv
conv7 (fc7)
19x19x1024
conv8
10x10x512
conv9
5x5x256
conv10
3x3x256
conv11
1x1x256
Detection
VGG-16
Extra feature layers

SSD

- Set of default BB of fixed size and ration
- Object of different sized is detected at different levels (depth)



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map