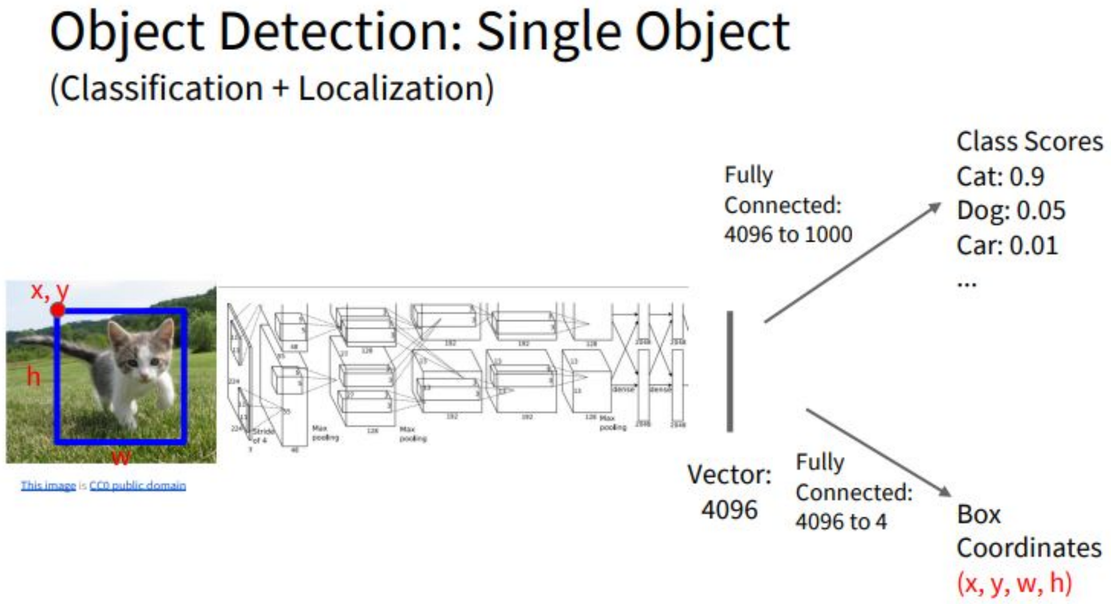
Detection

План лекции

- Localization
- R-CNN, Fast R-CNN
- One-staged detectors, YOLO
- Architectures
- Metrics

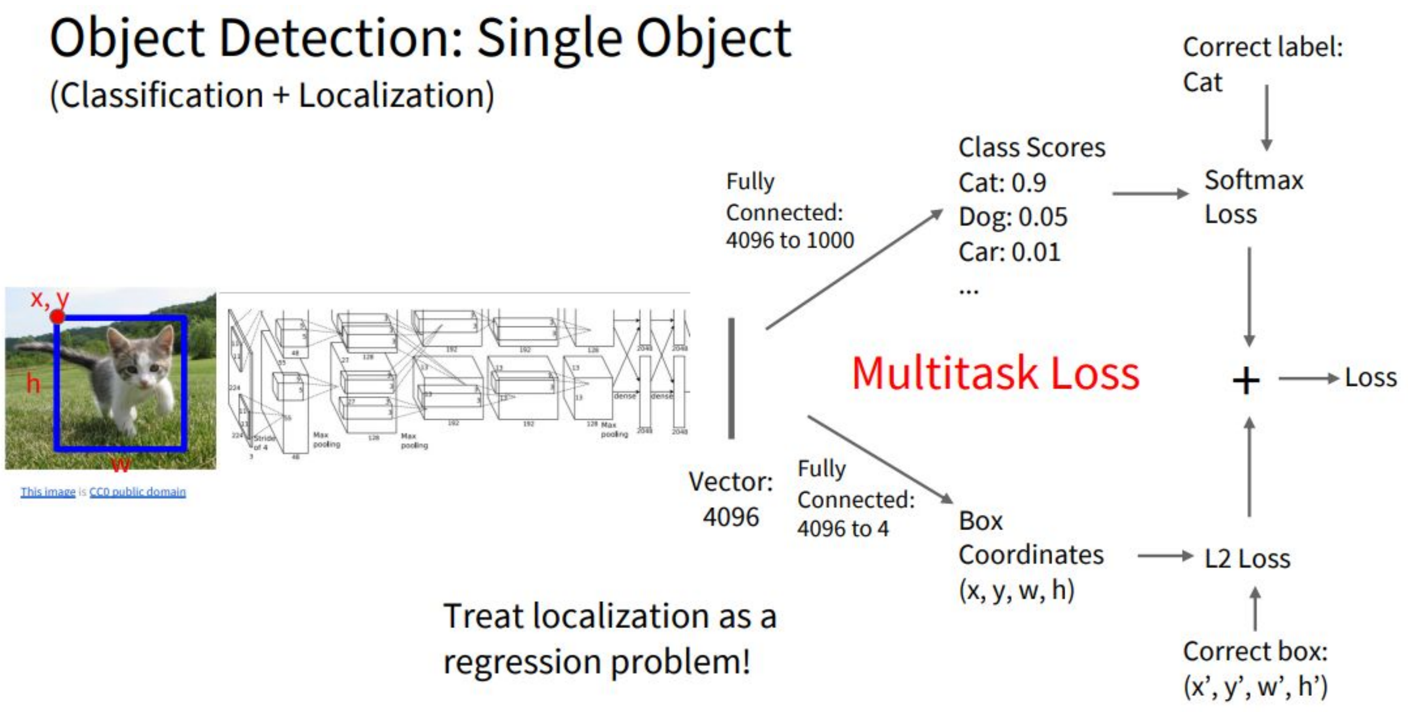
Object detection: Localization

Если бы мы знали, что на фото один объект. Какой тогда будет лосс?



link

Object detection: Localization



link

Object detection

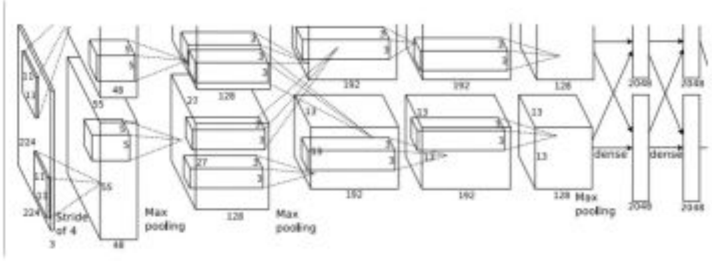
Но что делать, если на фото неизвестное нам число объектов?

link

Какое решение можно придумать, используя сеть для классификации?

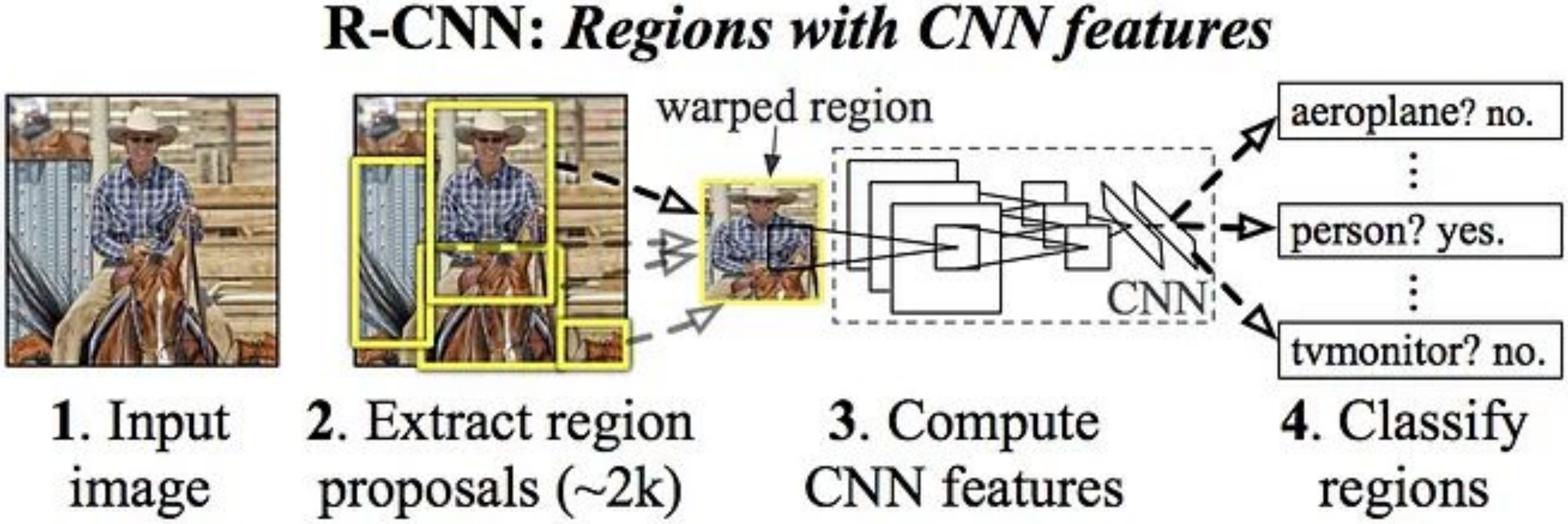
Какое решение можно придумать, используя сеть для классификации?

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



Dog? YES
Cat? NO
Background? NO

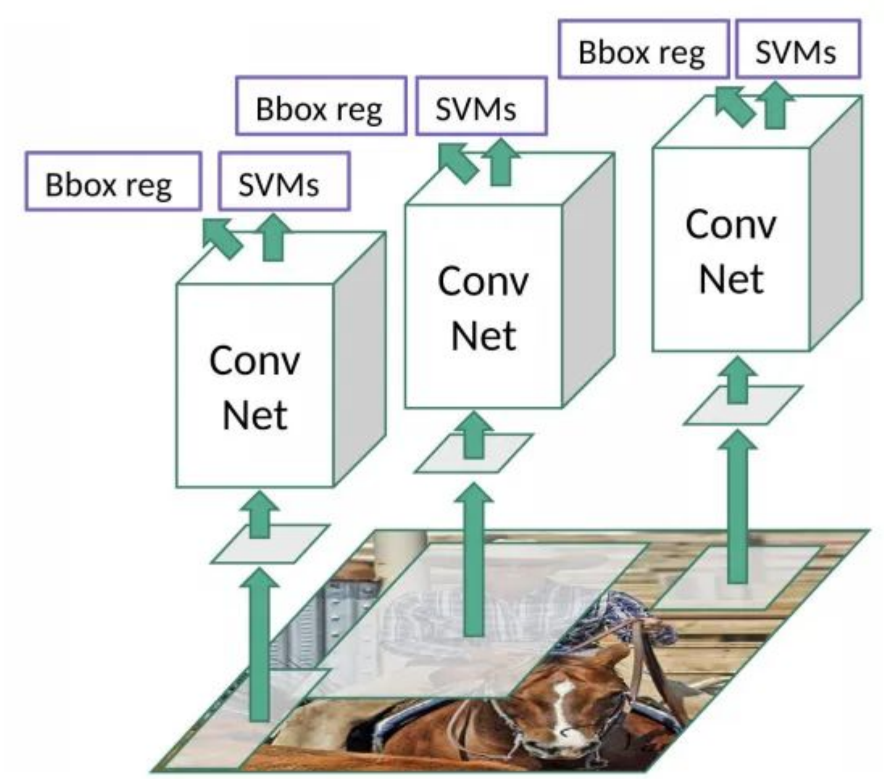
Object detection. R-CNN, 2015



Object detection. R-CNN, 2015

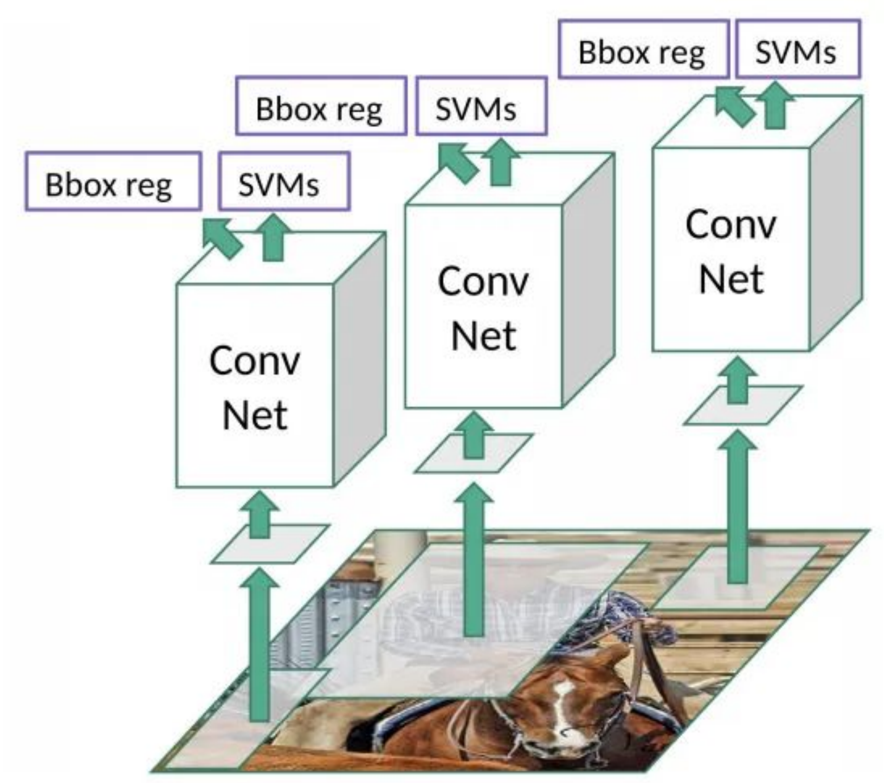
- Relatively fast to run; e.g. Selective Search gives 2000 region proposals in a few seconds on CPU

Object detection. R-CNN



Какие проблемы могут возникнуть у этой сети?

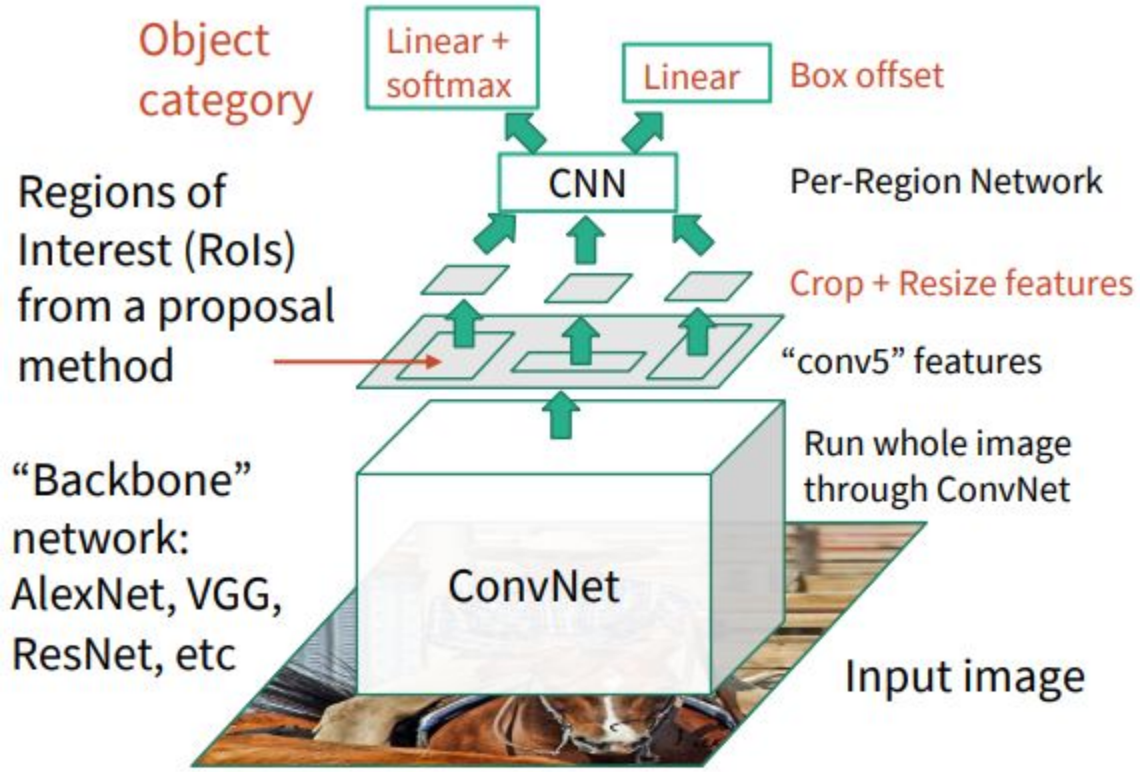
Object detection. R-CNN



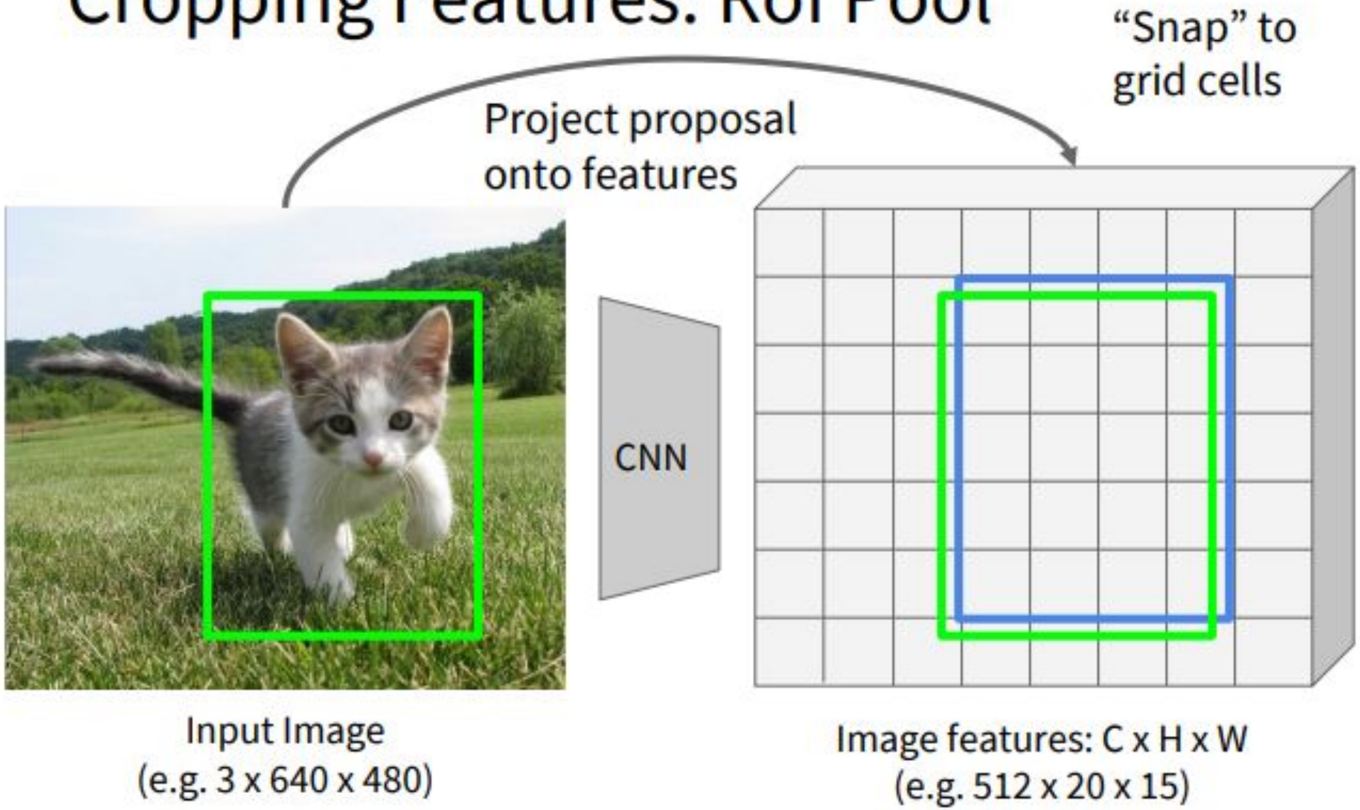
- Необходимо классифицировать 2000 предложенных регионов на каждом изображении
- Алгоритм выбора регионов никак не обучается. Это может привести к созданию плохих предложений регионов-кандидатов

- Как можно улучшить?

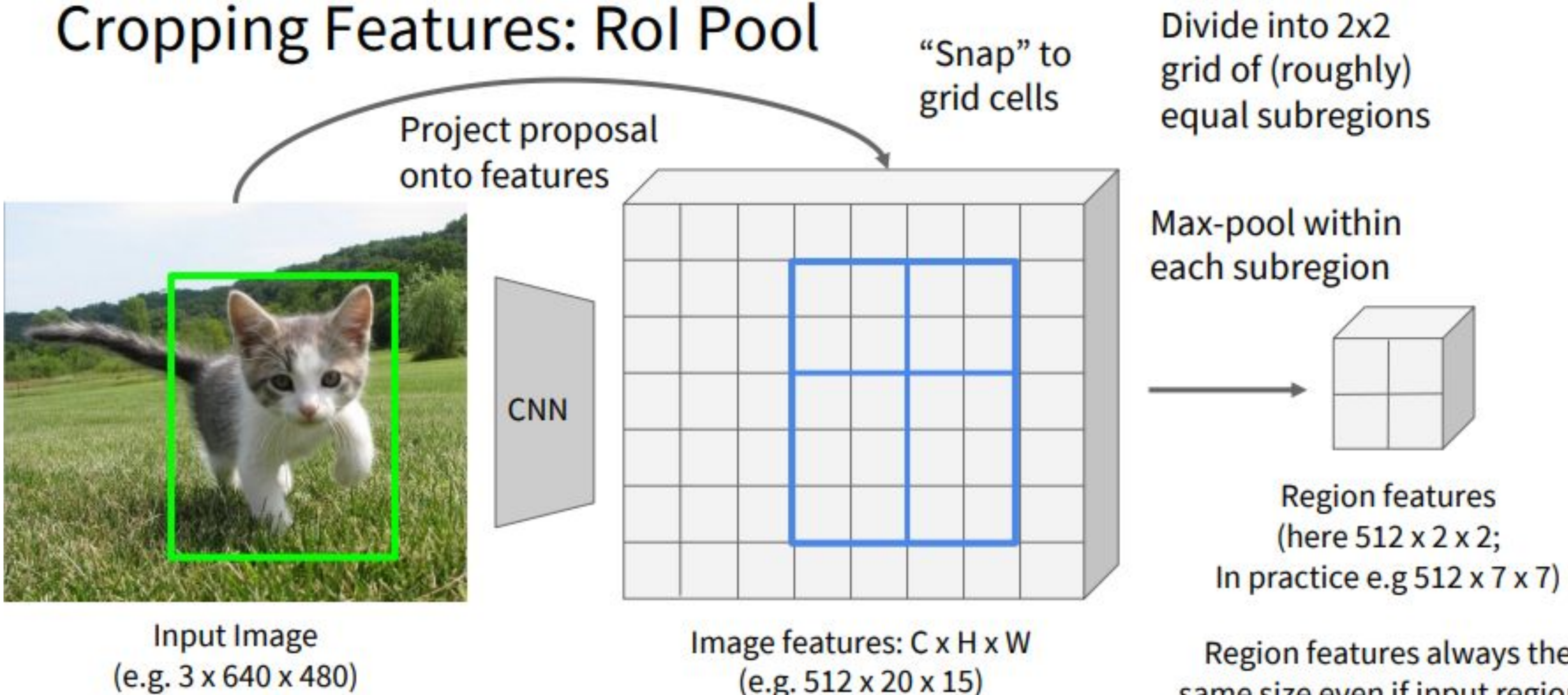
Object detection. Fast R-CNN



Cropping Features: RoI Pool

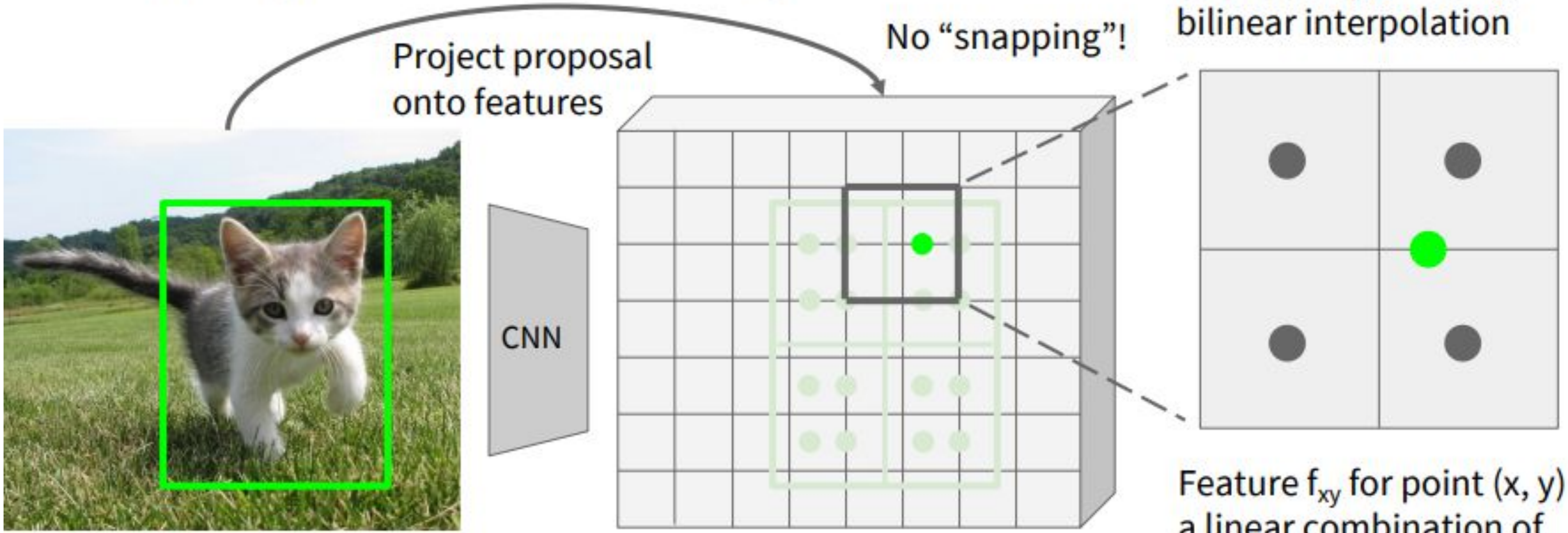


Cropping Features: RoI Pool



Region features always the same size even if input regions have different sizes!

Cropping Features: RoI Align



Input Image
(e.g. 3 x 640 x 480)

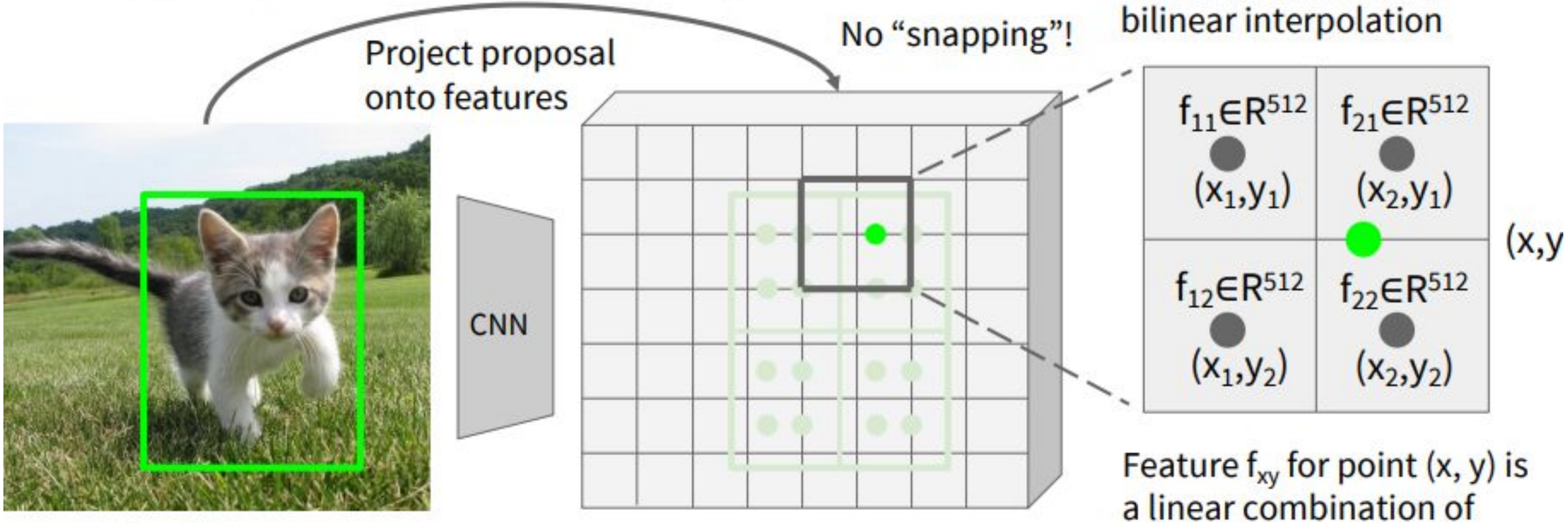
Image features: C x H x W
(e.g. 512 x 20 x 15)

Sample at regular points in each subregion using bilinear interpolation

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

Cropping Features: RoI Align

Sample at regular points in each subregion using bilinear interpolation



Input Image
(e.g. 3 x 640 x 480)

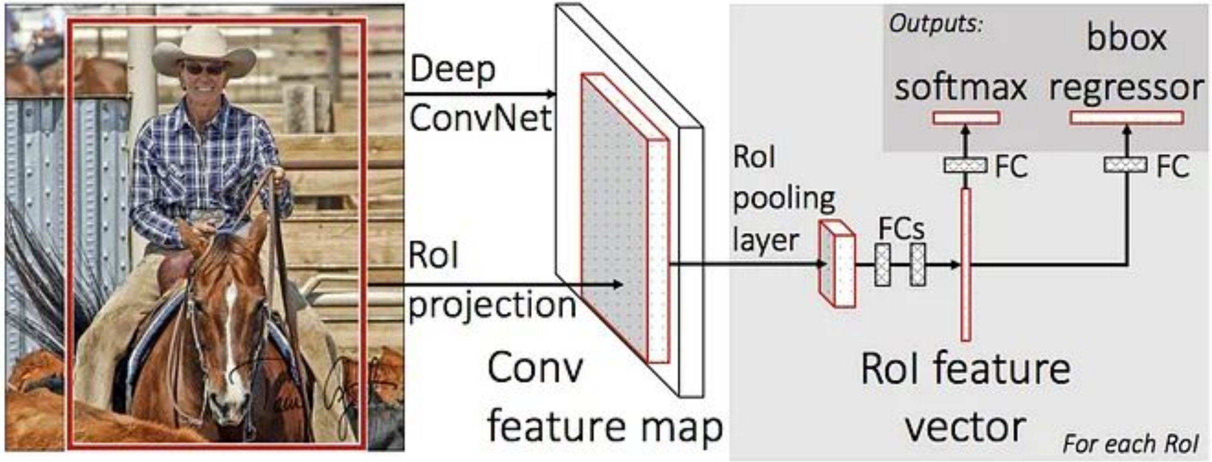
Image features: C x H x W
(e.g. 512 x 20 x 15)

Feature f_{xy} for point (x, y) is a linear combination of features at its four neighboring grid cells:

$$f_{xy} = \sum_{i,j=1}^{2} f_{i,j} \max(0, 1 - |x - x_i|) \max(0, 1 - |y - y_j$$

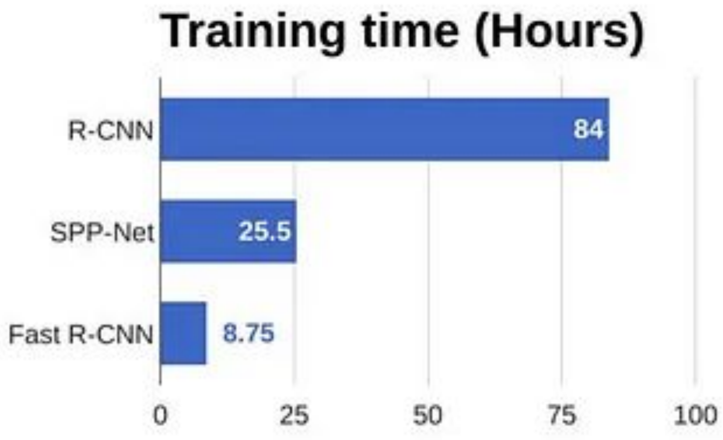
et al, "Mask R-CNN", ICCV 2017

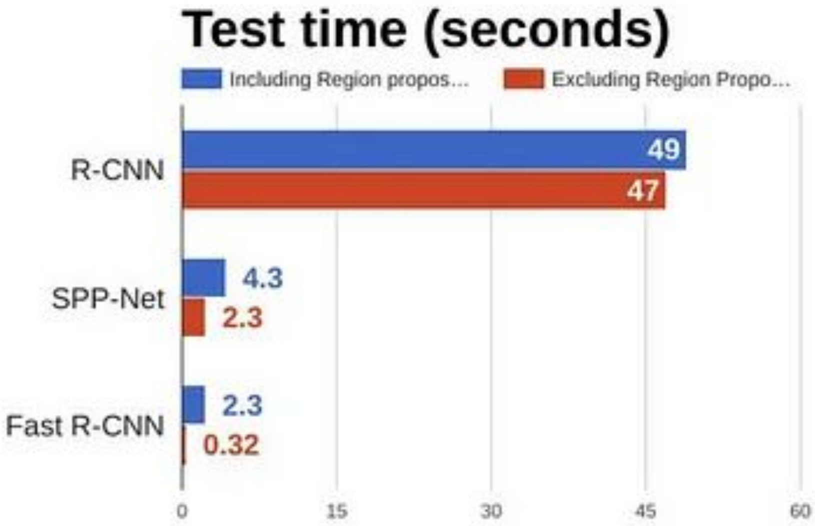
Object detection. Fast R-CNN



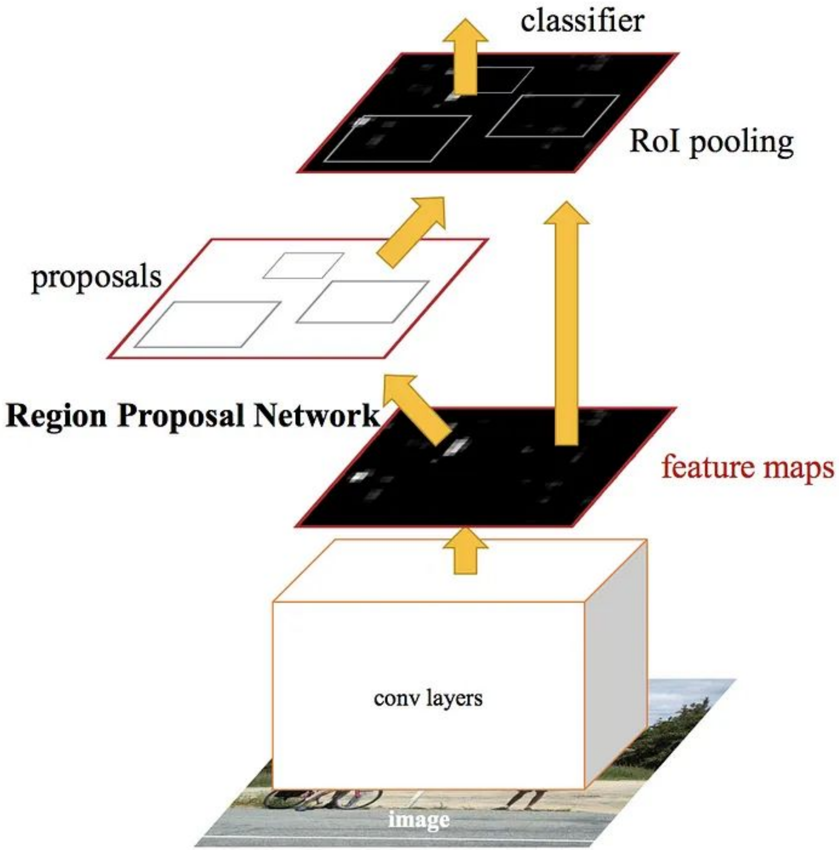
- Не нужно прогонять 2000 изображений через сеть. Карта признаков вычисляется один раз
- RoI pooling layer позволяет получать признаки одной размерности для применения полносвязного слоя
- Однако все еще используется алгоритм поиска регионов интереса

Object detection. R-CNN vs Fast R-CNN





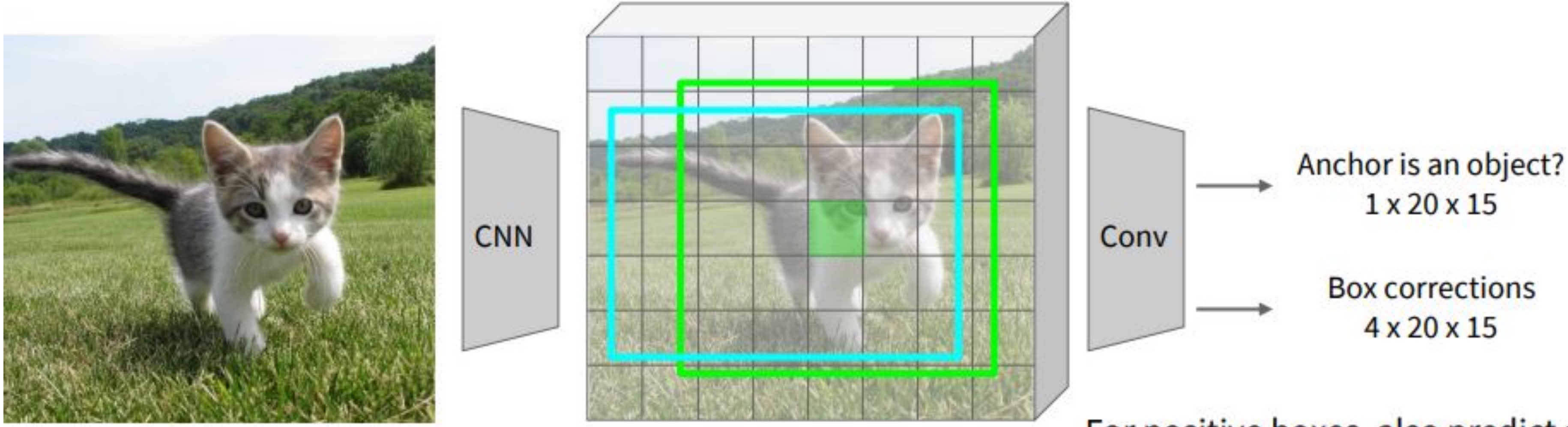
Object detection. Faster R-CNN



- Вместо использования selective search на карте признаков, для region proposals используется отдельная сеть. Region proposals проходят через ROI pooling, который затем используется для классификации изображения в предложенной области и прогнозирования значений смещения для ограничивающих рамок.

Region Proposal Network

Imagine an anchor box of fixed size at each point in the feature map

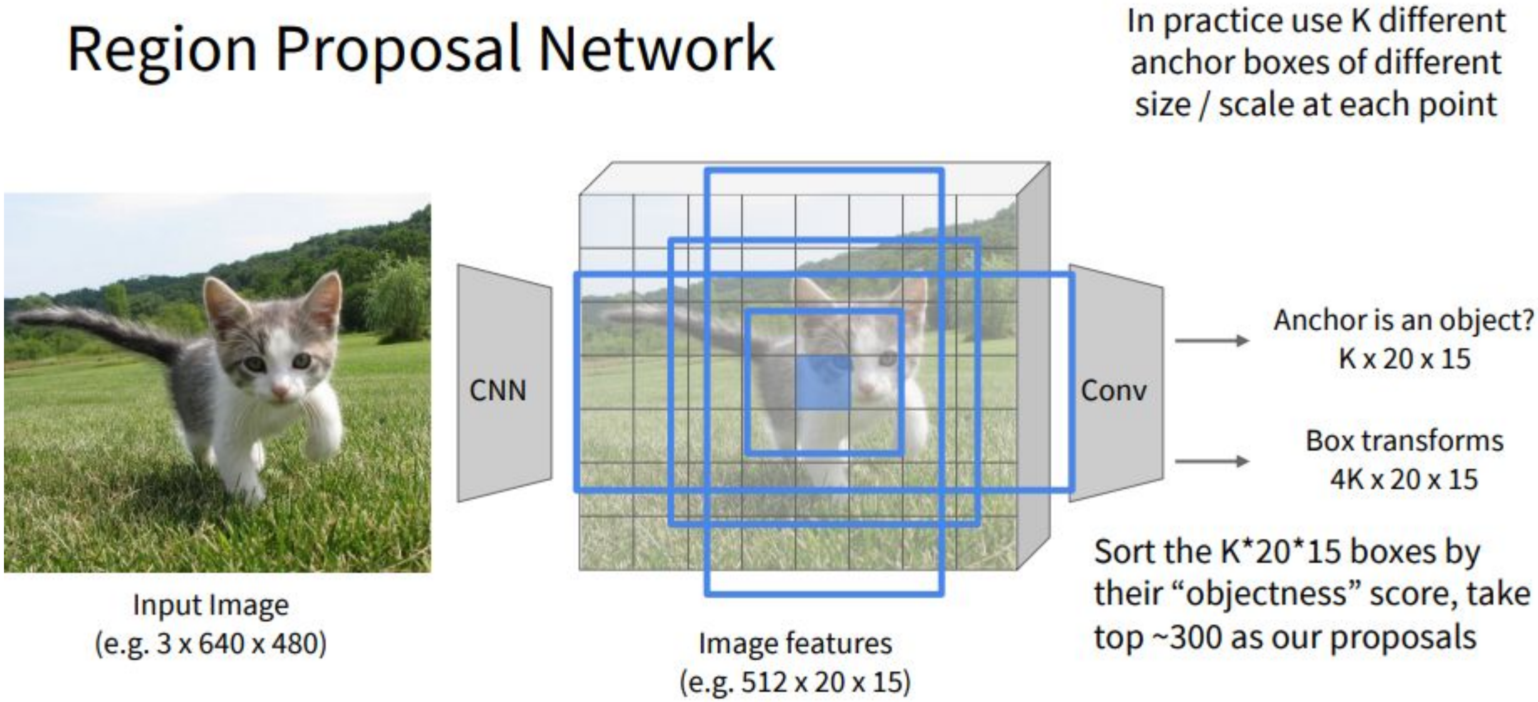


Input Image
(e.g. 3 x 640 x 480)

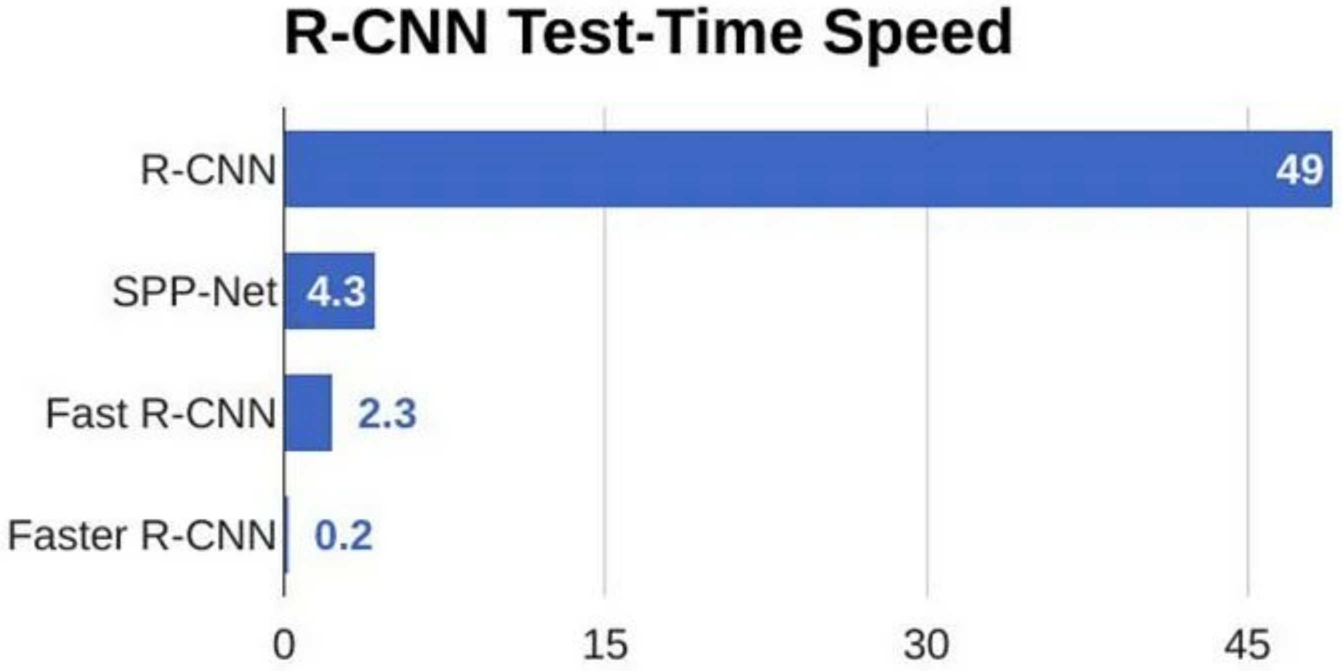
Image features
(e.g. 512 x 20 x 15)

For positive boxes, also predict a corrections from the anchor to the ground-truth box (regress 4 numbers per pixel)

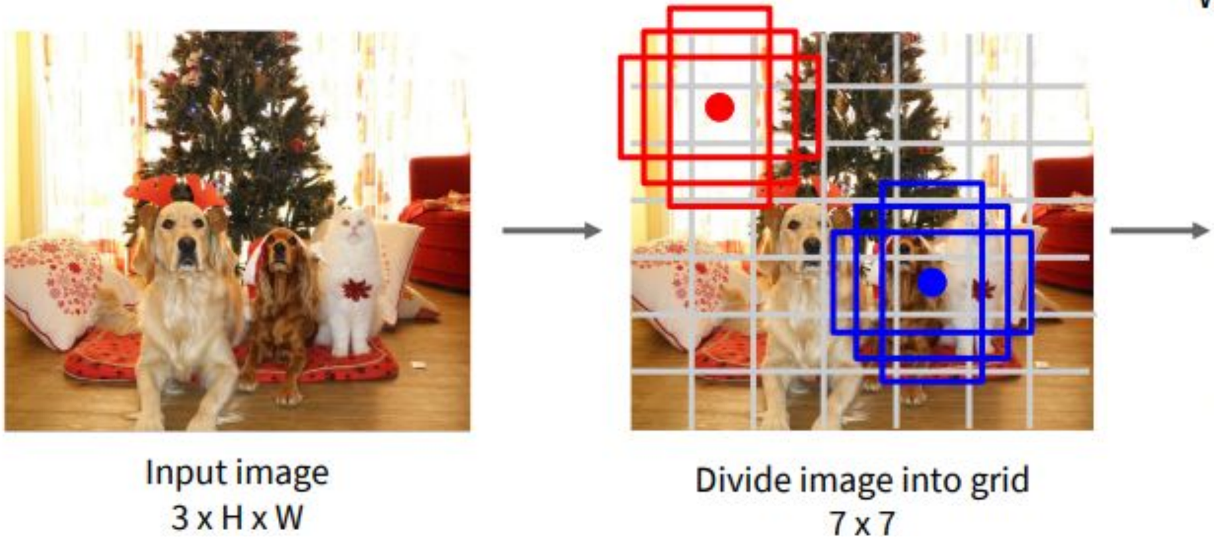
Object detection. Region Proposal Network



Object detection. Fast R-CNN vs Faster R-CNN



Object detection. Single-Stage Object Detectors: YOLO / SSD / RetinaNet



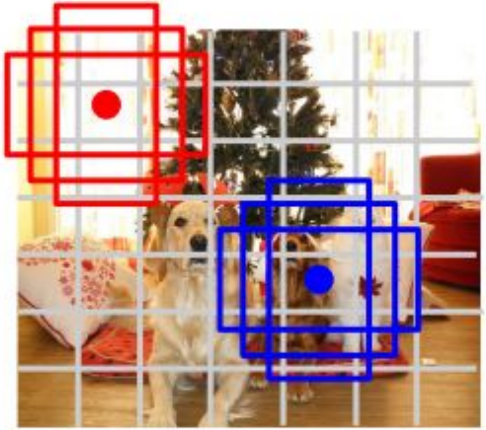
Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers: (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)
- Looks a lot like RPN, but category-specific!

Какой будет размер выхода?

Object detection. Single-Stage Object Detectors: YOLO / SSD / RetinaNet

Input image
3 x H x W



Divide image into grid
7 x 7

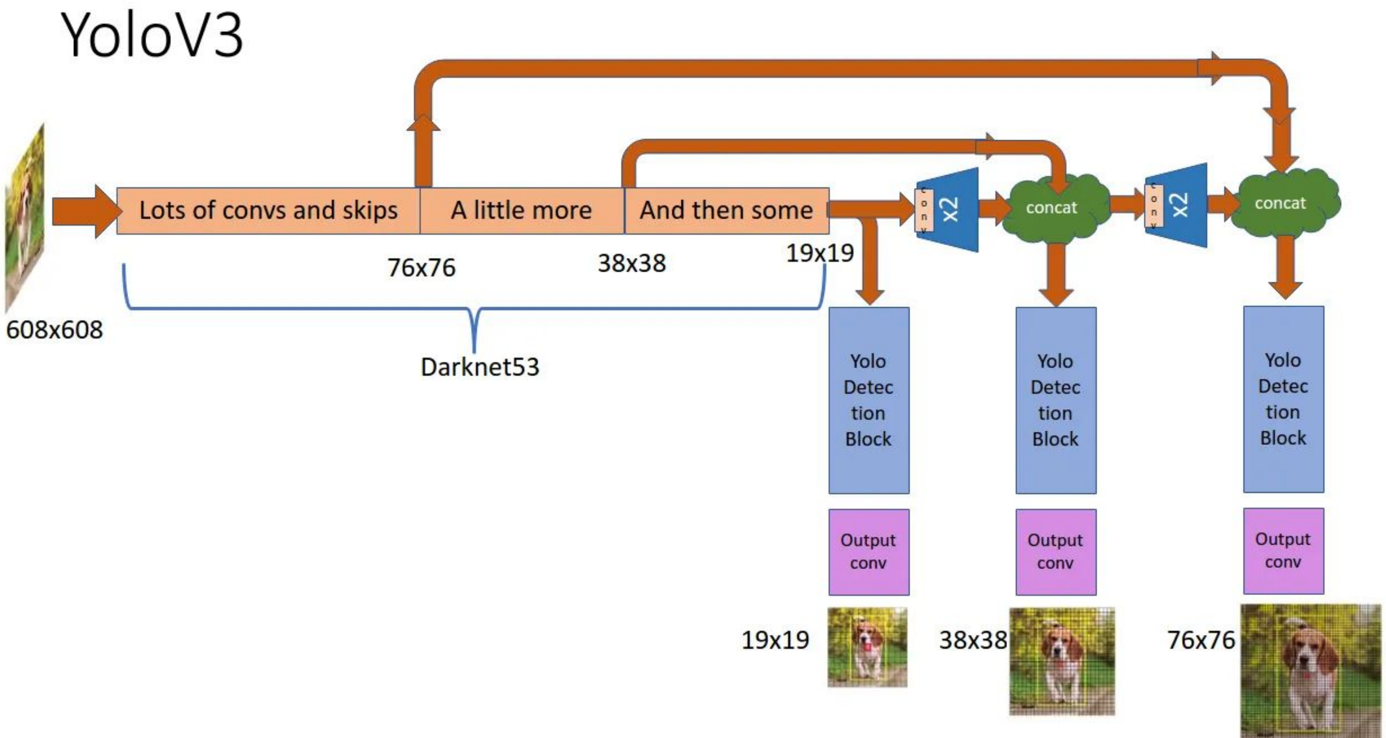
Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers: (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)
- Looks a lot like RPN, but category-specific!

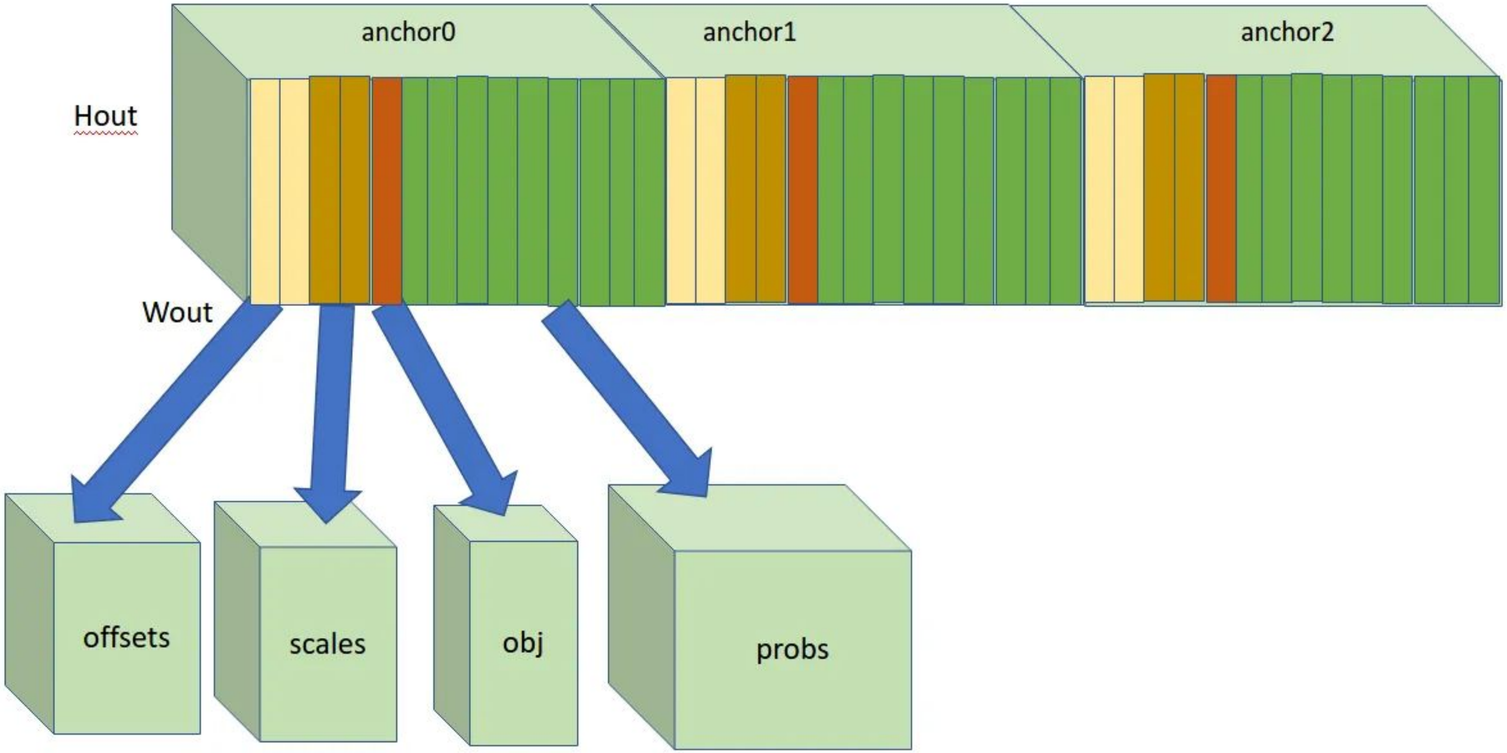
Какой будет размер выхода? $7 \times 7 \times (5 * B + C)$

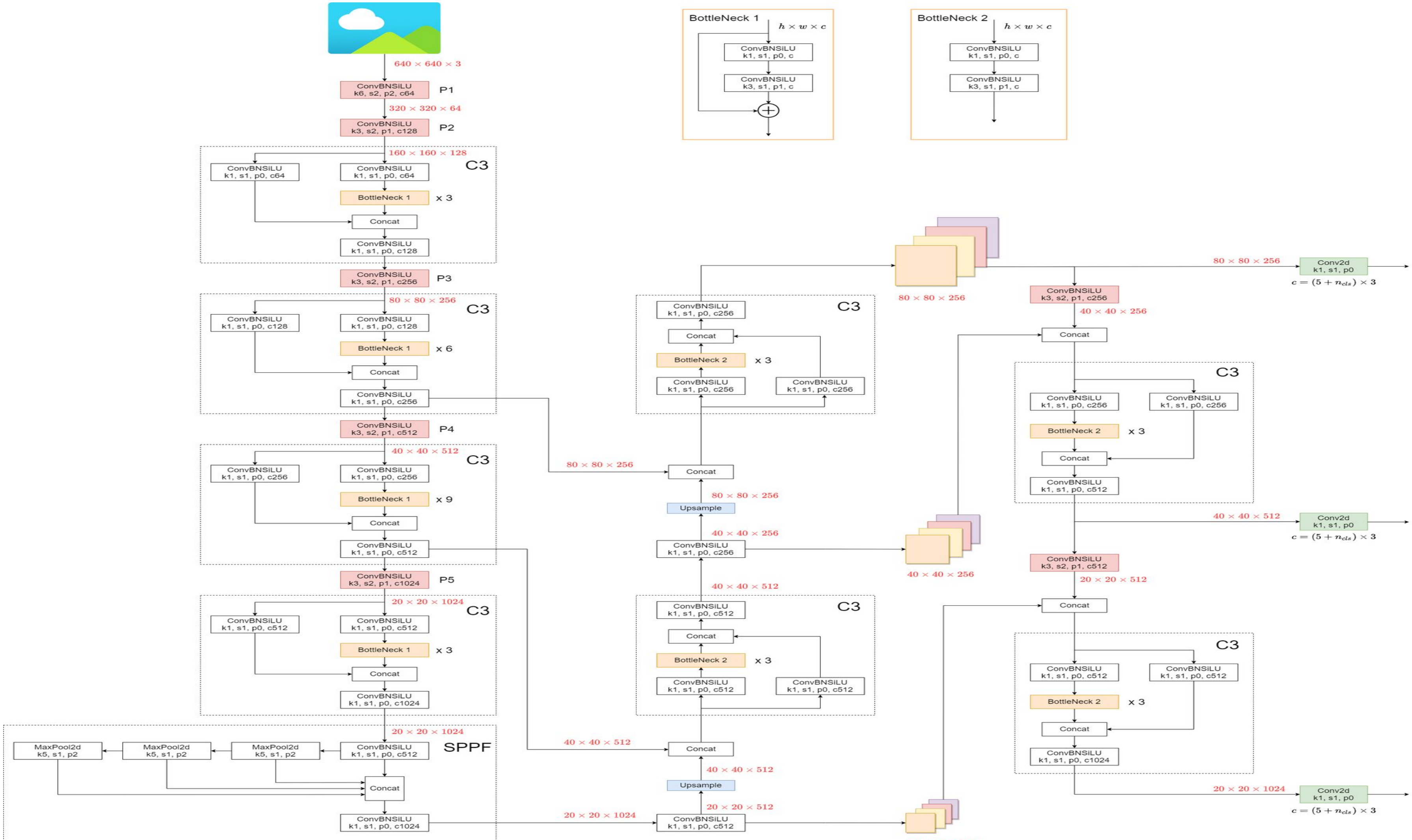
Object detection. YOLO

You look only once. 2016. > 40FPS

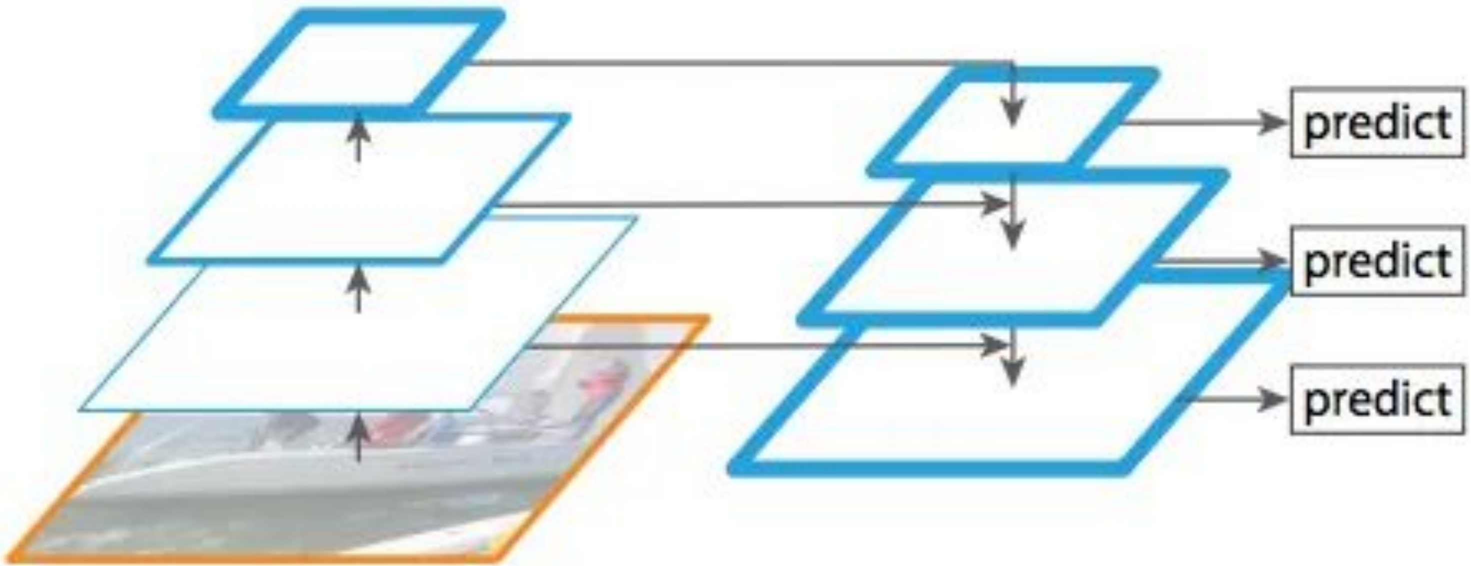


Object detection. YOLO

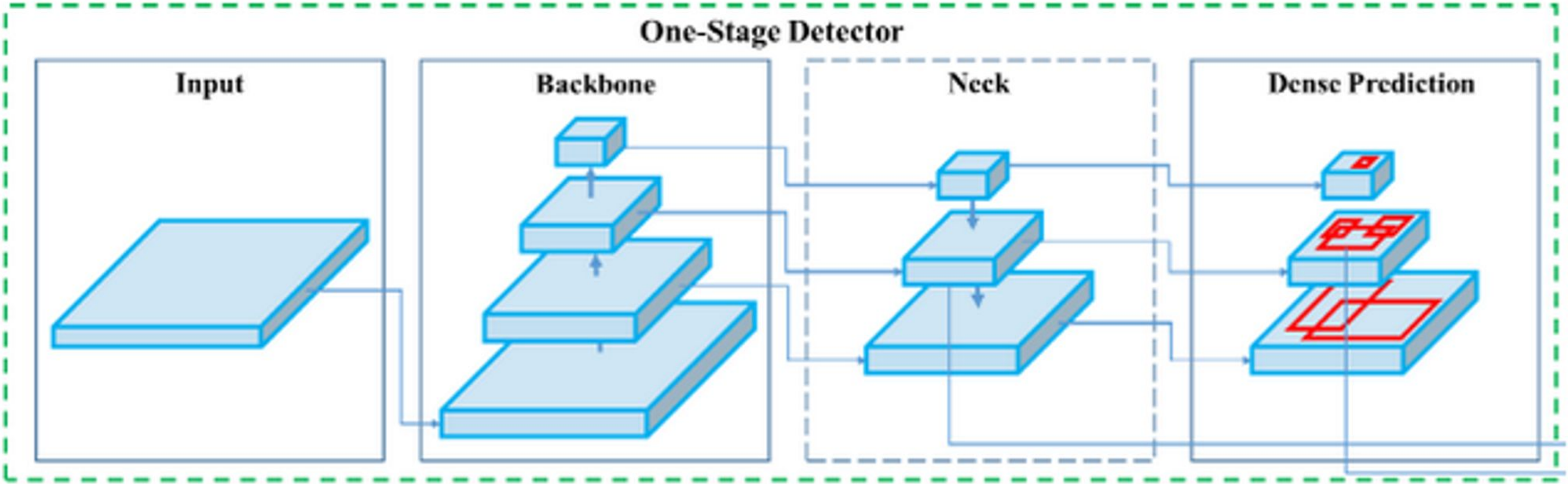



BottleNeck 1
$h \times w \times c$
ConvBNSiLU k1, s1, p0, c
ConvBNSiLU k3, s1, p1, c
BottleNeck 2
$h \times w \times c$
ConvBNSiLU k1, s1, p0, c
ConvBNSiLU k3, s1, p1, c
$640 \times 640 \times 3$
ConvBNSiLU k6, s2, p2, c64
P1
$320 \times 320 \times 64$
ConvBNSiLU k3, s2, p1, c128
P2
$160 \times 160 \times 128$
C3
ConvBNSiLU k1, s1, p0, c64
ConvBNSiLU k1, s1, p0, c64
BottleNeck 1
x 3
Concat
ConvBNSiLU k1, s1, p0, c128
ConvBNSiLU k3, s2, p1, c256
P3
$80 \times 80 \times 256$
C3
ConvBNSiLU k1, s1, p0, c128
ConvBNSiLU k1, s1, p0, c128
BottleNeck 1
x 6
Concat
ConvBNSiLU k1, s1, p0, c256
ConvBNSiLU k3, s2, p1, c512
P4
$40 \times 40 \times 512$
C3
ConvBNSiLU k1, s1, p0, c256
ConvBNSiLU k1, s1, p0, c256
BottleNeck 1
x 9
Concat
ConvBNSiLU k1, s1, p0, c512
ConvBNSiLU k3, s2, p1, c1024
P5
$20 \times 20 \times 1024$
C3
ConvBNSiLU k1, s1, p0, c512
ConvBNSiLU k1, s1, p0, c512
BottleNeck 1
x 3
Concat
ConvBNSiLU k1, s1, p0, c1024
$20 \times 20 \times 1024$
SPPF
MaxPool2d k5, s1, p2
MaxPool2d k5, s1, p2
MaxPool2d k5, s1, p2
ConvBNSiLU k1, s1, p0, c512
Concat
ConvBNSiLU k1, s1, p0, c1024
$20 \times 20 \times 1024$
ConvBNSiLU k1, s1, p0, c512
$20 \times 20 \times 512$
Upsample
$40 \times 40 \times 512$
$40 \times 40 \times 512$
Concat
C3
ConvBNSiLU k1, s1, p0, c512
ConvBNSiLU k1, s1, p0, c512
BottleNeck 2
x 3
Concat
ConvBNSiLU k1, s1, p0, c512
$40 \times 40 \times 512$
ConvBNSiLU k1, s1, p0, c256
$40 \times 40 \times 256$
Upsample
$80 \times 80 \times 256$
$80 \times 80 \times 256$
Concat
C3
ConvBNSiLU k1, s1, p0, c256
ConvBNSiLU k1, s1, p0, c256
BottleNeck 2
x 3
Concat
ConvBNSiLU k1, s1, p0, c256
$80 \times 80 \times 256$
$40 \times 40 \times 256$
$80 \times 80 \times 256$
Conv2d k1, s1, p0
$c = (5 + n_{cls}) \times 3$
ConvBNSiLU k3, s2, p1, c256
$40 \times 40 \times 256$
Concat
C3
ConvBNSiLU k1, s1, p0, c256
ConvBNSiLU k1, s1, p0, c256
BottleNeck 2
x 3
Concat
ConvBNSiLU k1, s1, p0, c512
$40 \times 40 \times 512$
Conv2d k1, s1, p0
$c = (5 + n_{cls}) \times 3$
ConvBNSiLU k3, s2, p1, c512
$20 \times 20 \times 512$
Concat
C3
ConvBNSiLU k1, s1, p0, c512
ConvBNSiLU k1, s1, p0, c512
BottleNeck 2
x 3
Concat
ConvBNSiLU k1, s1, p0, c1024
$20 \times 20 \times 1024$
Conv2d k1, s1, p0
$c = (5 + n_{cls}) \times 3$

Feature Pyramid Network

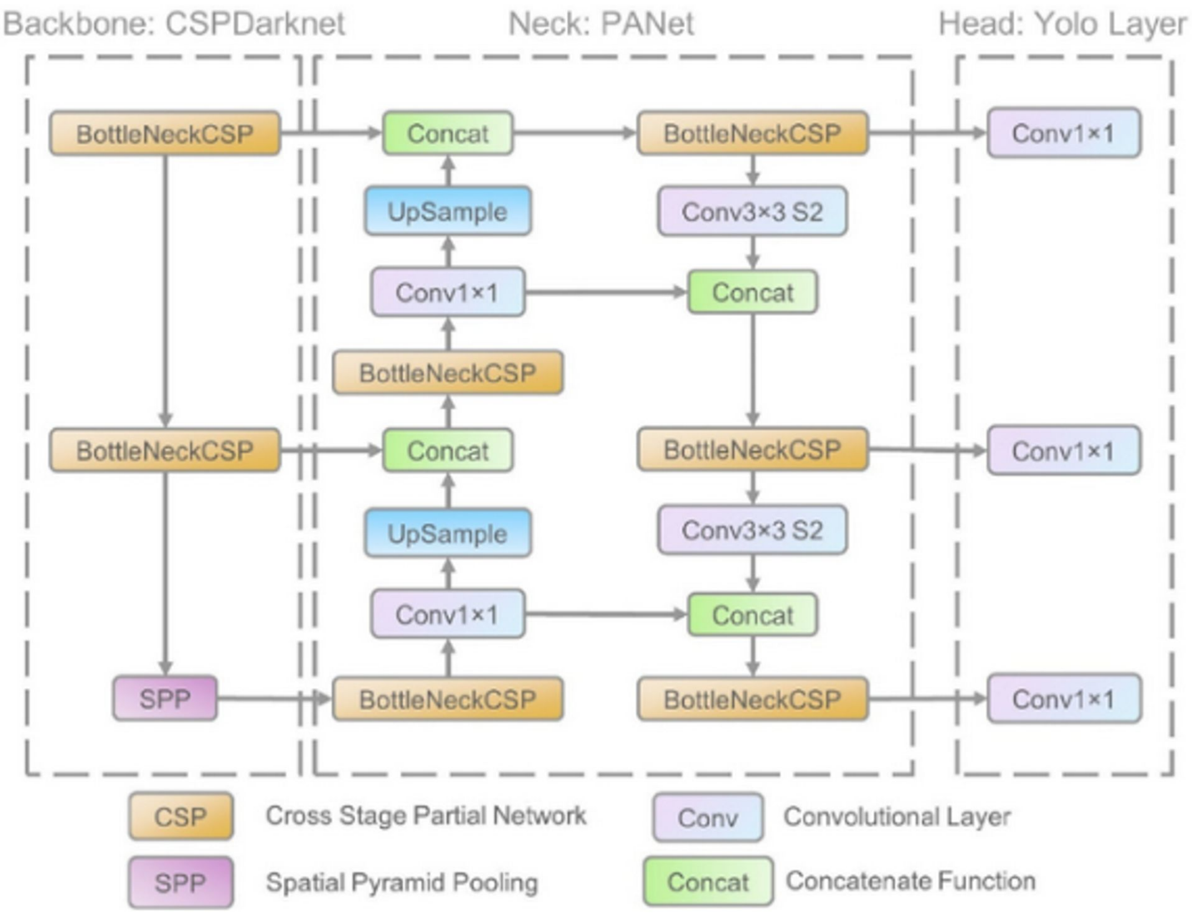


YOLOv5



Single-Stage Detector Architecture [1]

YOLOv5



Network architecture for YOLO v5 [2]

NMS

Algorithm 1 Non-Max Suppression

procedure $\text{NMS}(B, c)$

$B_{nms} \leftarrow \emptyset$ Initialize empty set

for $b_i \in B$ **do** => Iterate over all the boxes

$discard \leftarrow \text{False}$ Take boolean variable and set it as false. This variable indicates whether b(i) should be kept or discarded

for $b_j \in B$ **do** Start another loop to compare with b(i)

if $\text{same}(b_i, b_j) > \boldsymbol{\lambda}_{\mathbf{nms}}$ **then** If both boxes having same IOU

if $\text{score}(c, b_j) > \text{score}(c, b_i)$ **then**

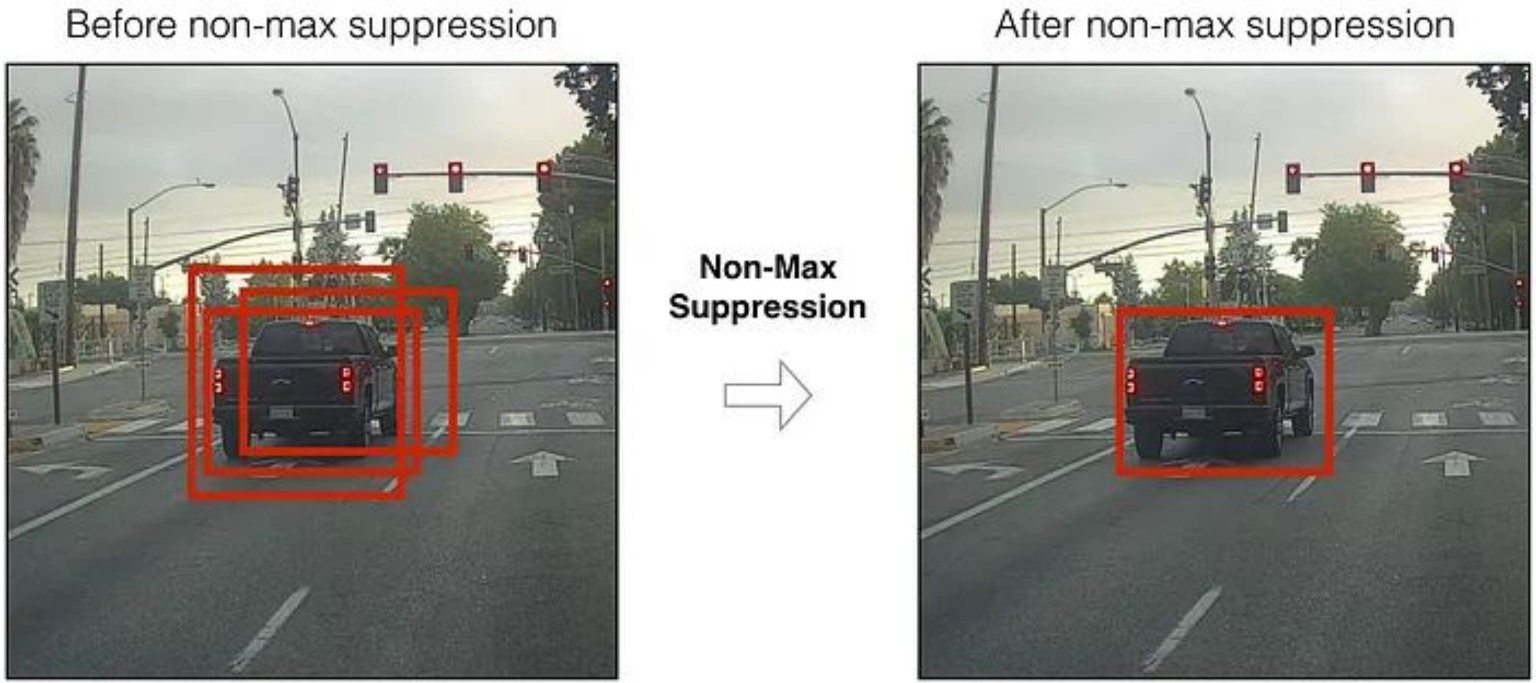
$discard \leftarrow \text{True}$ Compare the scores. If score of b(i) is less than that of b(j), b(i) should be discarded, so set the flag to True.

if not $discard$ **then** Once b(i) is compared with all other boxes and still the discarded flag is False, then b(i) should be considered. So add it to the final list.

$B_{nms} \leftarrow B_{nms} \cup b_i$

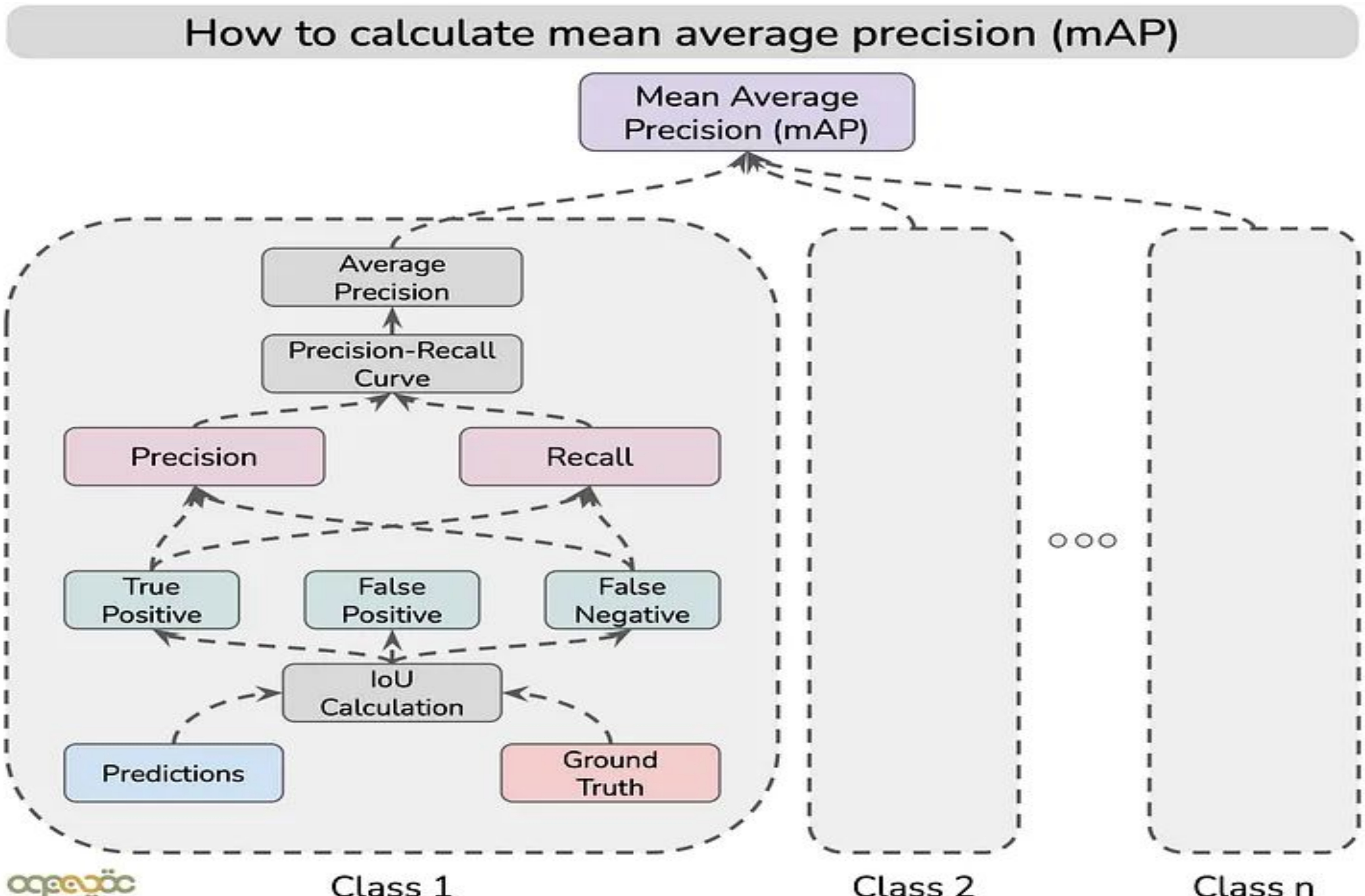
return B_{nms} Do the same procedure for remaining boxes and return the final list

NMS


Before non-max suppression
Non-Max
Suppression
After non-max suppression

Detection metrics

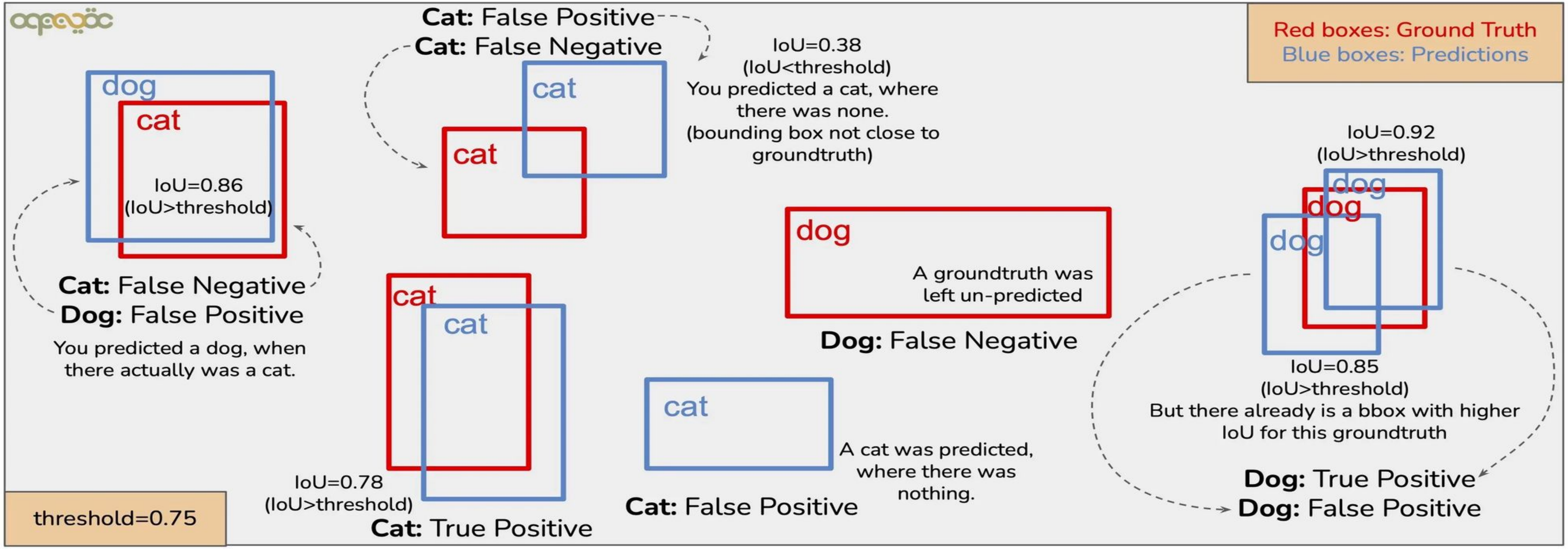
[link]

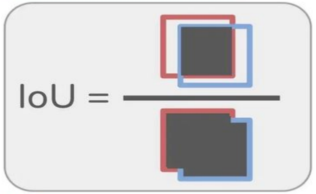


Detection metrics

link

Object Detection and Localization - IoU, True Positive, False Positive, False Negative





Threshold	Class	# GroundTruth	# predictions	TP	FP	FN	Precision	Recall
0.75	Cat	3	3	1	2	2	1/3	1/3
	Dog	2	3	1	2	1	1/3	1/2
0.35	Cat	3	3	2	1	1	2/3	2/3
	Dog	2	3	1	2	1	1/3	1/2

@_aqeelanwar
aqeelanwarmalik

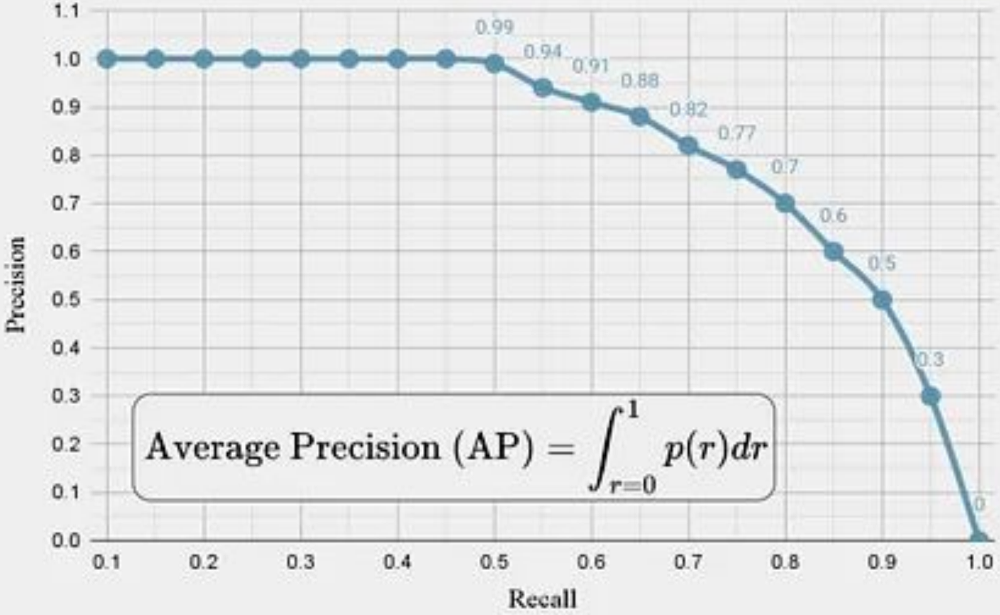
Detection metrics

link

Precision Recall Curve (PR Curve)

Conf. Thresh.	Recall	Precision	Effect
0.95	0.1	1	More FN
0.9	0.15	1	
0.85	0.2	1	
0.8	0.25	1	
0.75	0.3	1	
0.7	0.35	1	
0.65	0.4	1	
0.6	0.45	1	
0.55	0.5	0.99	
0.5	0.55	0.94	
0.45	0.6	0.91	
0.4	0.65	0.88	
0.35	0.7	0.82	
0.3	0.75	0.77	
0.25	0.8	0.7	
0.2	0.85	0.6	
0.15	0.9	0.5	
0.1	0.95	0.3	
0.05	1	0	More FP

The **smaller the probability confidence threshold**, the higher the number of detections made by the model, and the lower the chances that the ground-truth labels were missed and hence **higher the recall** *(Generally, but not always)*. On the other hand, **the higher the confidence threshold**, the more confident the model is in what it predicts and hence **higher the precision** *(Generally, but not always)*.



$$\text{Average Precision (AP)} = \int_{r=0}^{1} p(r)dr$$

Detection metrics

link

Calculating Average Precision from PR Curve

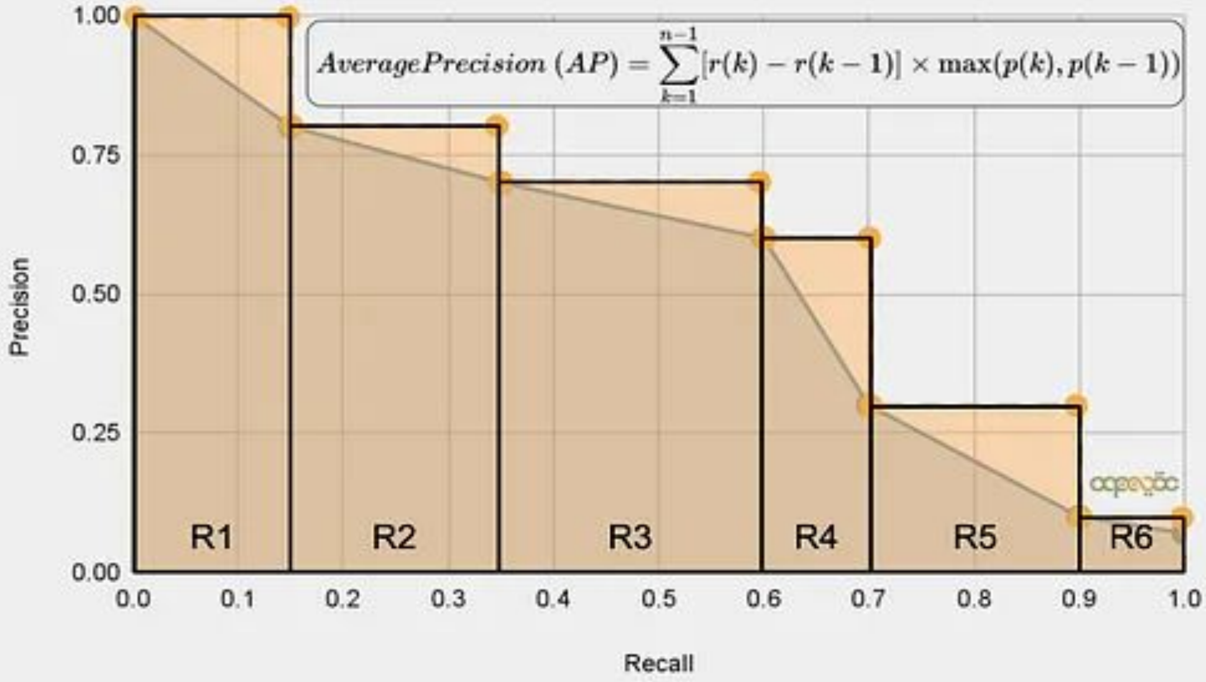
@_aqeelanwar aqeelanwarmalik

Approach 1: Sample-and-hold

- For each precision-recall pair (j=0, ..., n-1), the area under the PR curve can be found by approximating the curve using rectangles.
- The width of such rectangles can be found by taking the difference of two consecutive recall values (r(k), r(k-1)), and the height can be found by taking the maximum value of the precision for the selected recall values i.e.

$$w = r(k)-r(k-1)$$
$$h = \max(p(k), p(k-1))$$

- AP can be calculated by the sum of the areas of these rectangles.



Detection metrics

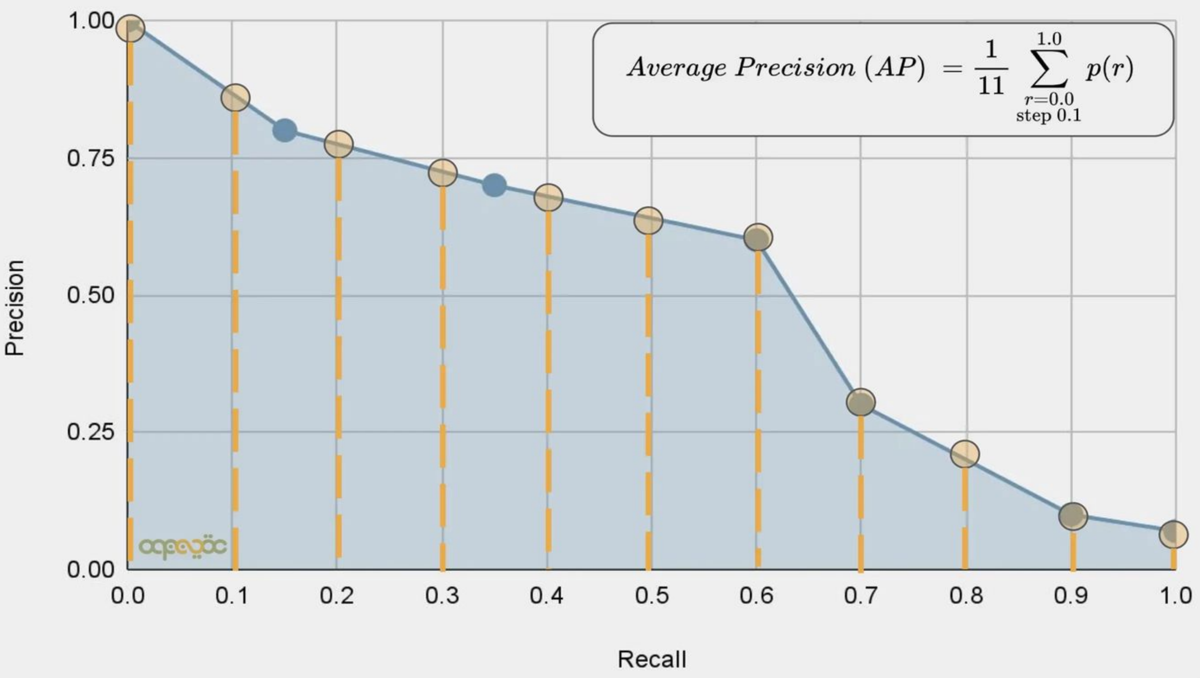
link

Calculating Average Precision from PR Curve

Approach 2: Interpolation and 11-point average

- The precision values for the 11 recall values from 0.0 to 1.0 with an increment of 0.1 is calculated
- These 11 points can be seen as orange samples in the figure on the right
- AP can be calculated by taking the mean of these 11 precision values i.e.

$$(AP) = \frac{1}{11} \sum_{\substack{r=0.0 \\ \text{step } 0.1}}^{1.0} p(r)$$



Detection metrics

[link]

$$\text{Average Precision (AP)} = \int_{r=0}^{1} p(r)dr$$

$$mAP = \frac{1}{k}\sum_{i}^{k} AP_i$$