Deep Learning

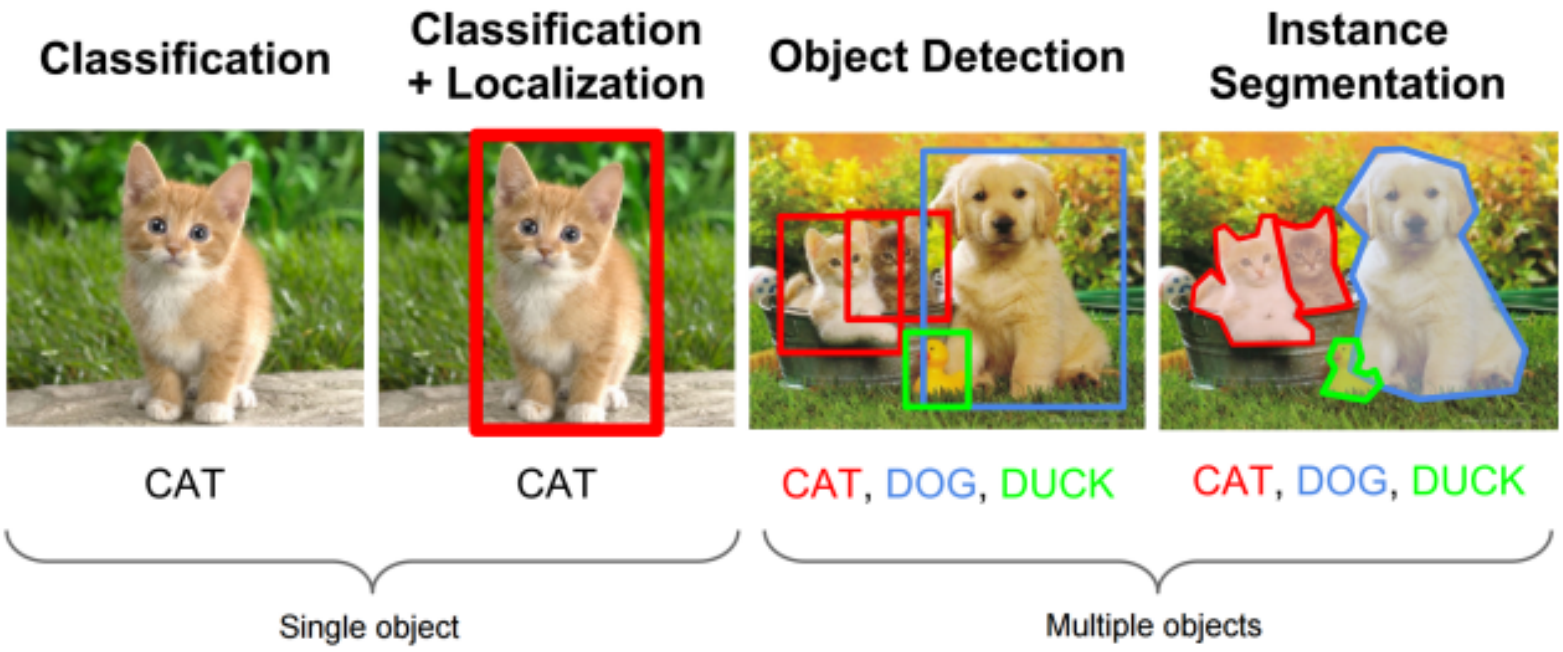
10. Detection and segmentation with neural networks

Viacheslav Dudar

Taras Shevchenko National University of Kyiv

2018

Overview


Classification
Classification + Localization
Object Detection
Instance Segmentation
CAT
CAT
CAT, DOG, DUCK
CAT, DOG, DUCK
Single object
Multiple objects

Localize objects with regression

Input: image

Only one object,
simpler than detection

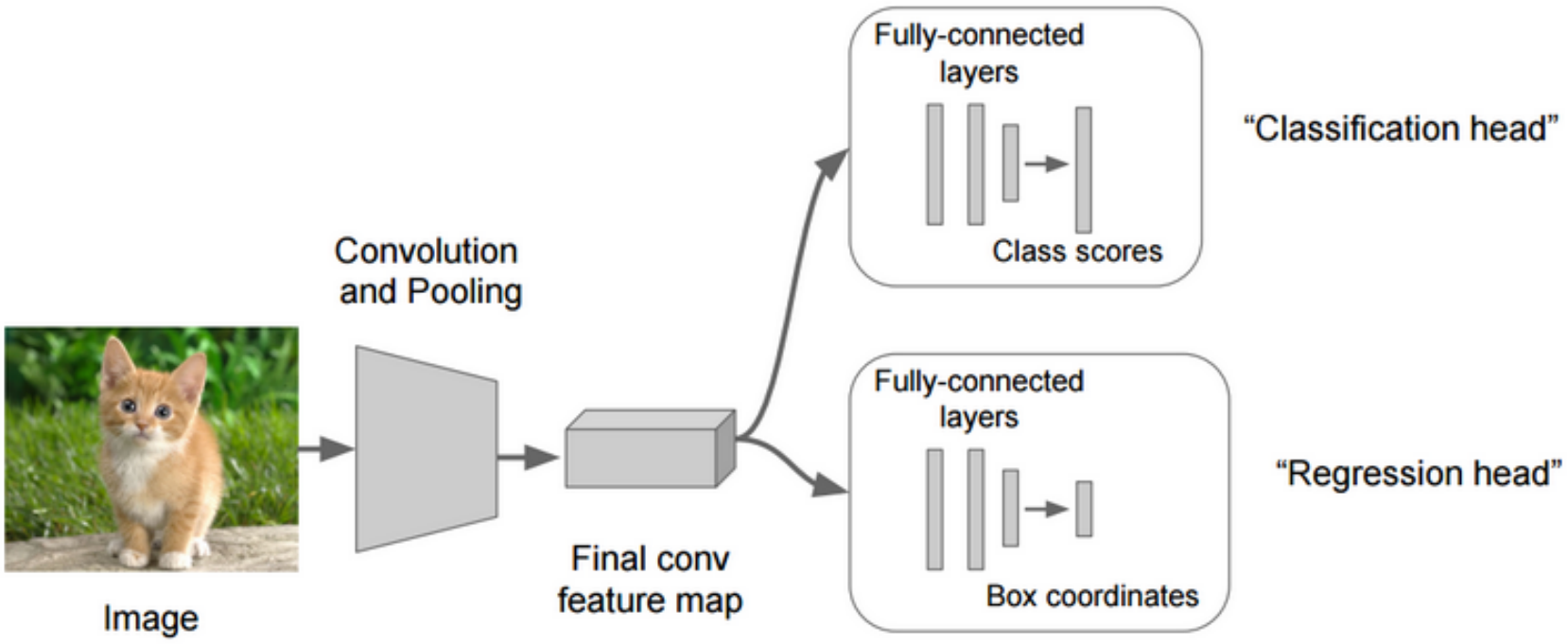
Neural Net

Output:
Box coordinates
(4 numbers)

Correct output:
box coordinates
(4 numbers)

Loss:
L2 distance

Localize objects with regression



Precision and recall

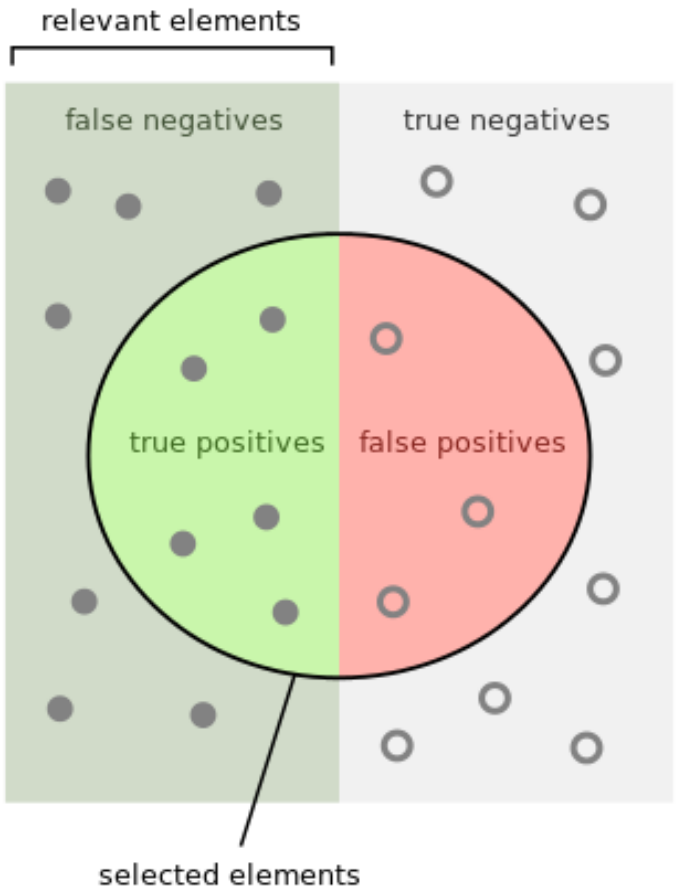
$$TP = \text{True positive}$$
$$TN = \text{True negative}$$
$$FP = \text{False positive}$$
$$FN = \text{False negative}$$

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

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

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

Precision and recall



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{selected elements}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$

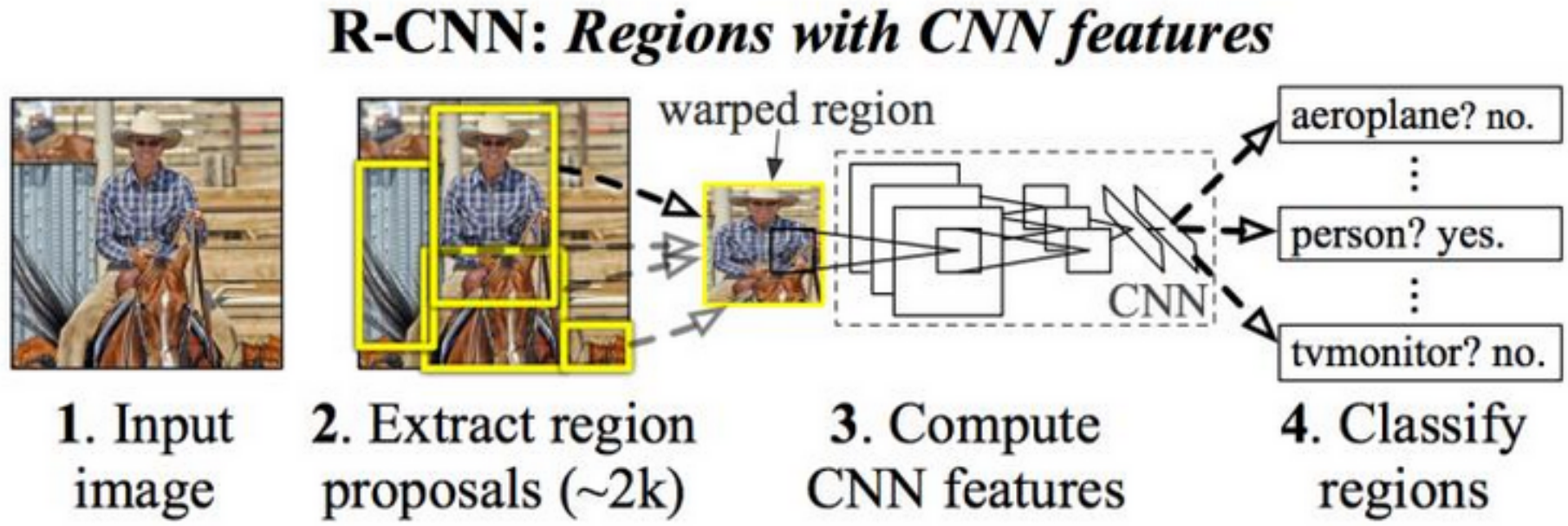
IoU (intersection over union)

Ground truth

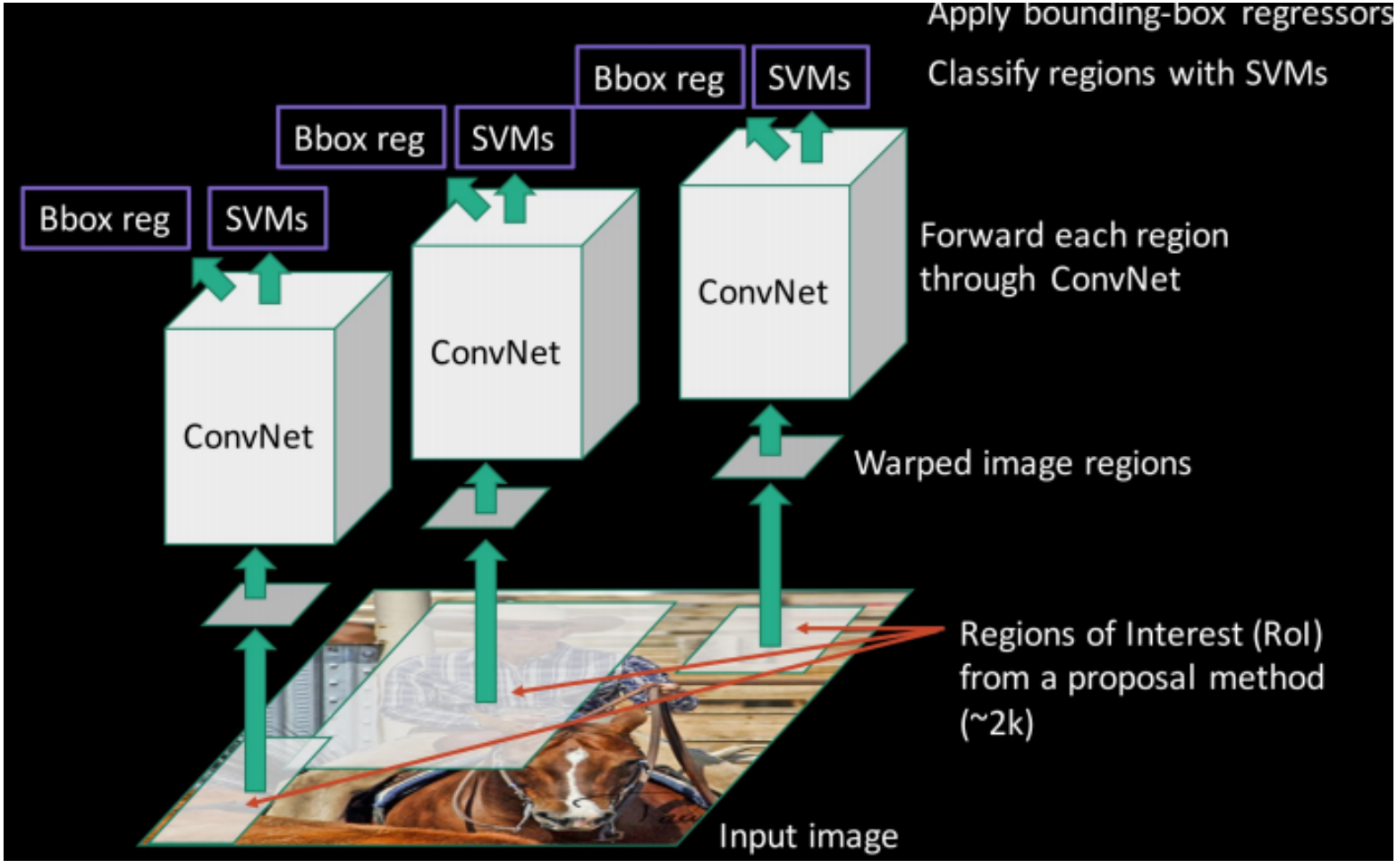
Prediction

$$IoU = \frac{\text{area of overlap}}{\text{area of union}}$$

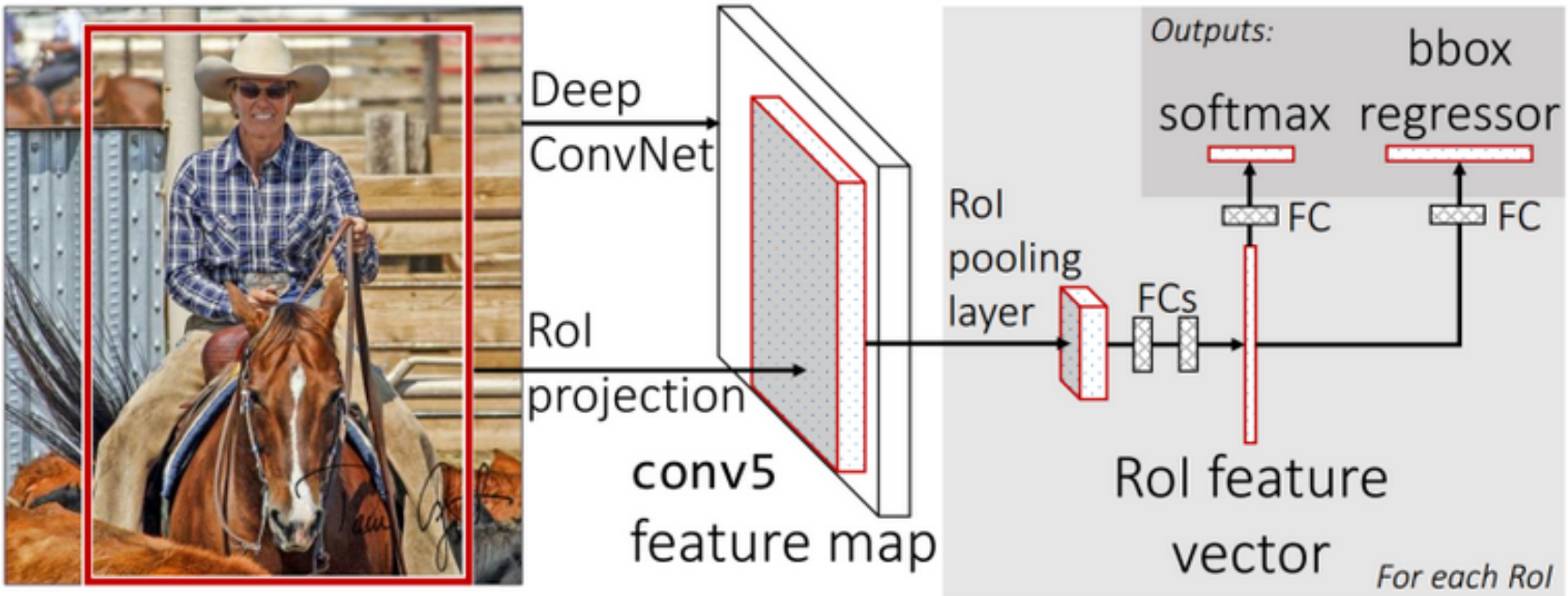
RCNN


R-CNN: Regions with CNN features
warped region
CNN
aeroplane? no.
person? yes.
tvmonitor? no.
1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

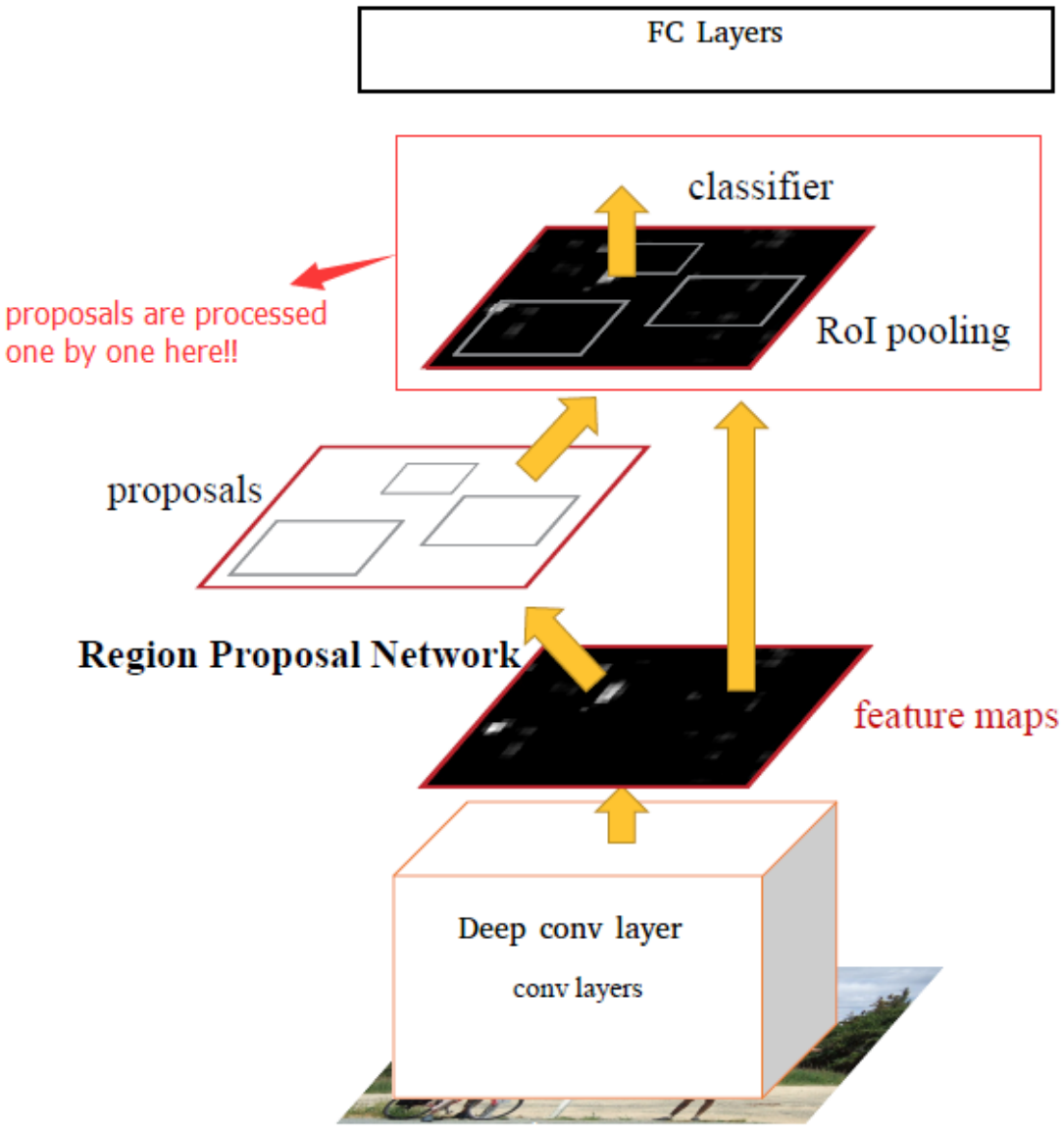
RCNN


Apply bounding-box regressors
Classify regions with SVMs
Bbox reg
SVMs
Bbox reg
SVMs
Bbox reg
SVMs
ConvNet
ConvNet
ConvNet
Forward each region through ConvNet
Warped image regions
Regions of Interest (RoI) from a proposal method (~2k)
Input image

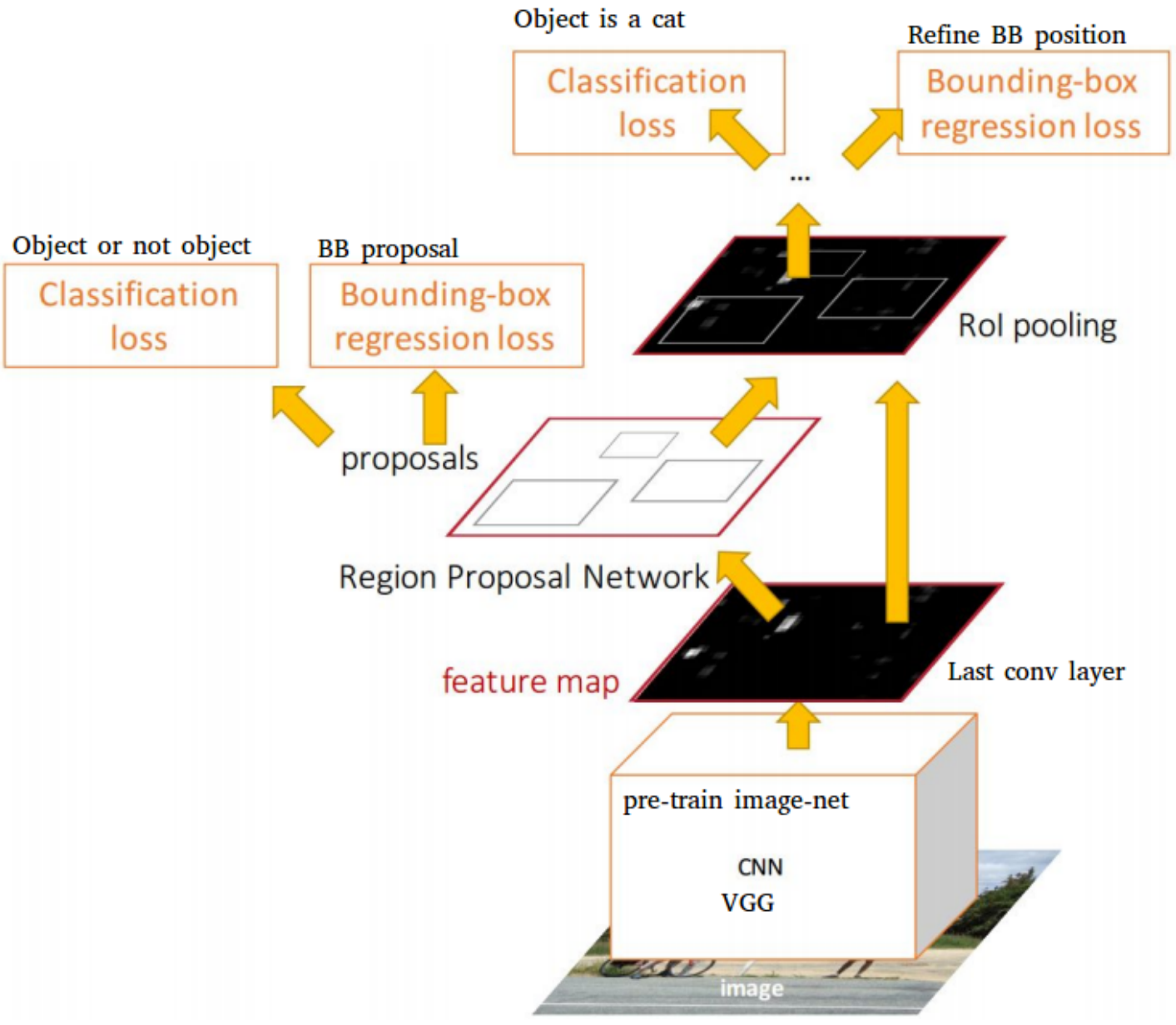
Fast RCNN


Deep
ConvNet
RoI
projection
conv5
feature map
RoI
pooling
layer
FCs
Outputs:
softmax
bbox
regressor
FC
FC
RoI feature
vector
For each RoI

Faster RCNN


FC Layers
classifier
RoI pooling
proposals are processed
one by one here!!
proposals
Region Proposal Network
feature maps
Deep conv layer
conv layers

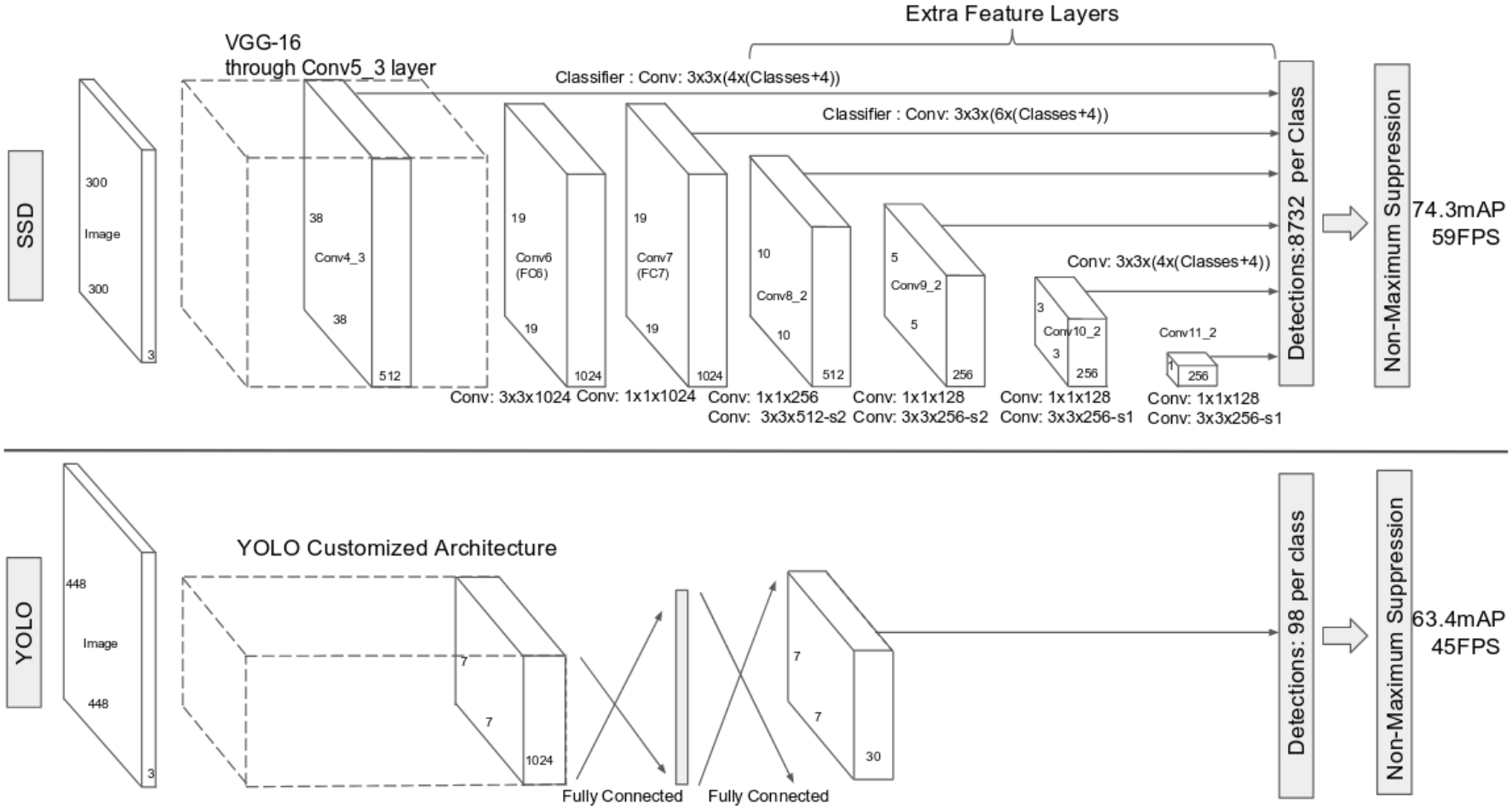
Faster RCNN


Object is a cat
Classification loss
Refine BB position
Bounding-box regression loss
...
Object or not object
Classification loss
BB proposal
Bounding-box regression loss
RoI pooling
proposals
Region Proposal Network
feature map
Last conv layer
pre-train image-net
CNN
VGG
image

Results

	R-CNN	**Fast R-CNN**	**Faster R-CNN**
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**66.9**

Single shot detectors

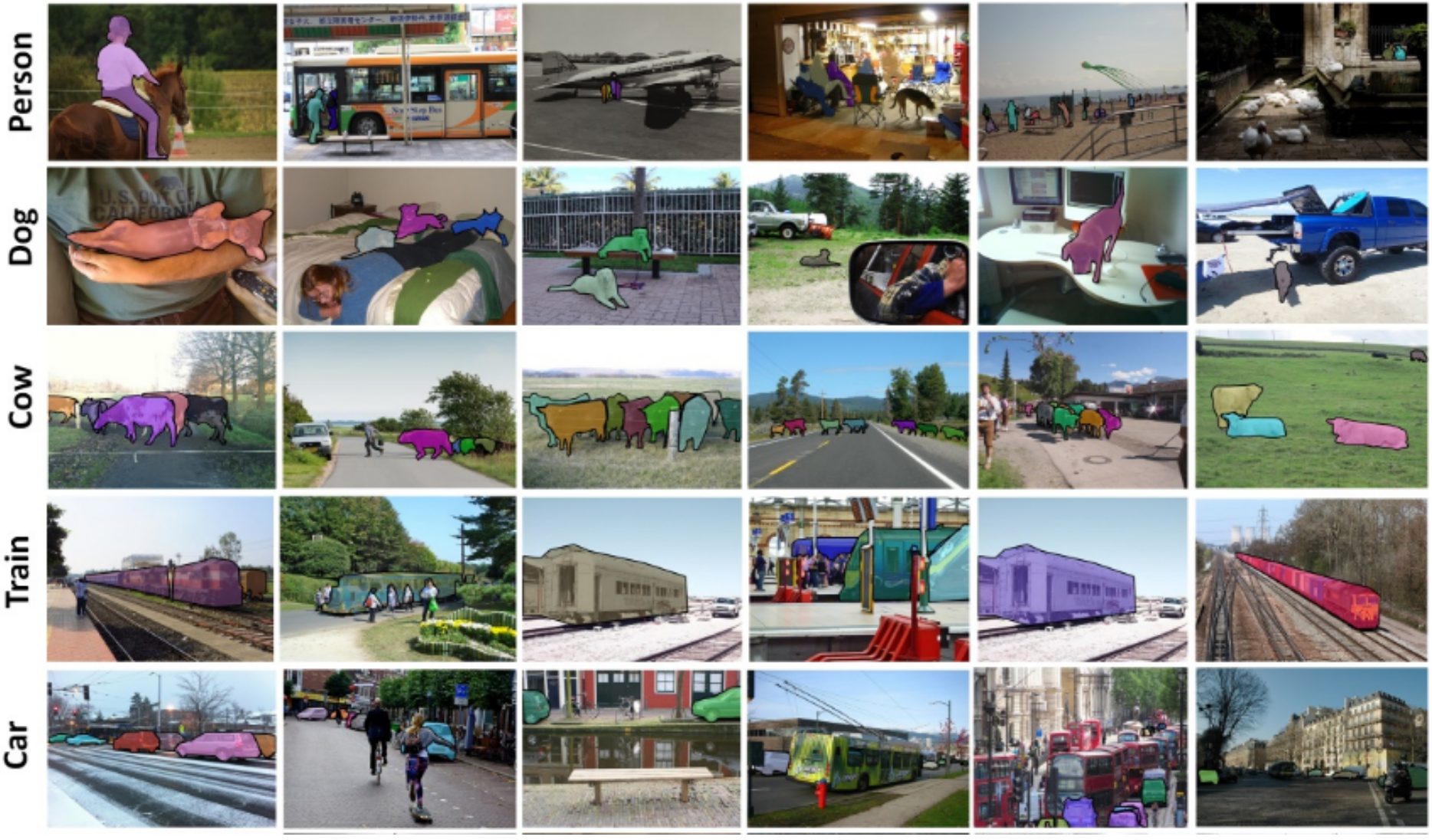


Datasets for detection and segmentation

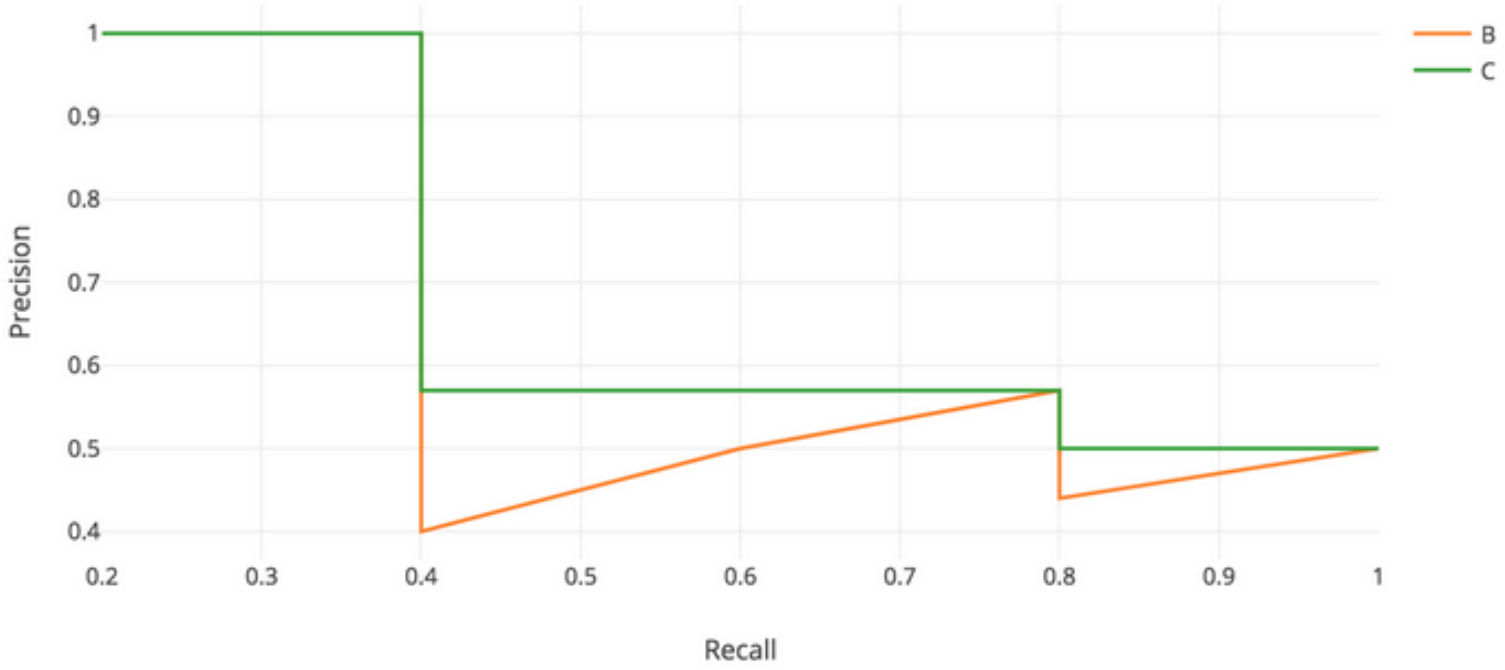
Coco dataset:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

Coco



Mean Average Precision



$$AP = \frac{1}{11} \sum_{r \in \{0.0,\ldots,1.0\}} AP_r$$

$$= \frac{1}{11} \sum_{r \in \{0.0,\ldots,1.0\}} p_{interp}(r)$$

where

$$p_{interp}(r) = \max_{\tilde{r} \geq r} p(\tilde{r})$$

COCO mAP

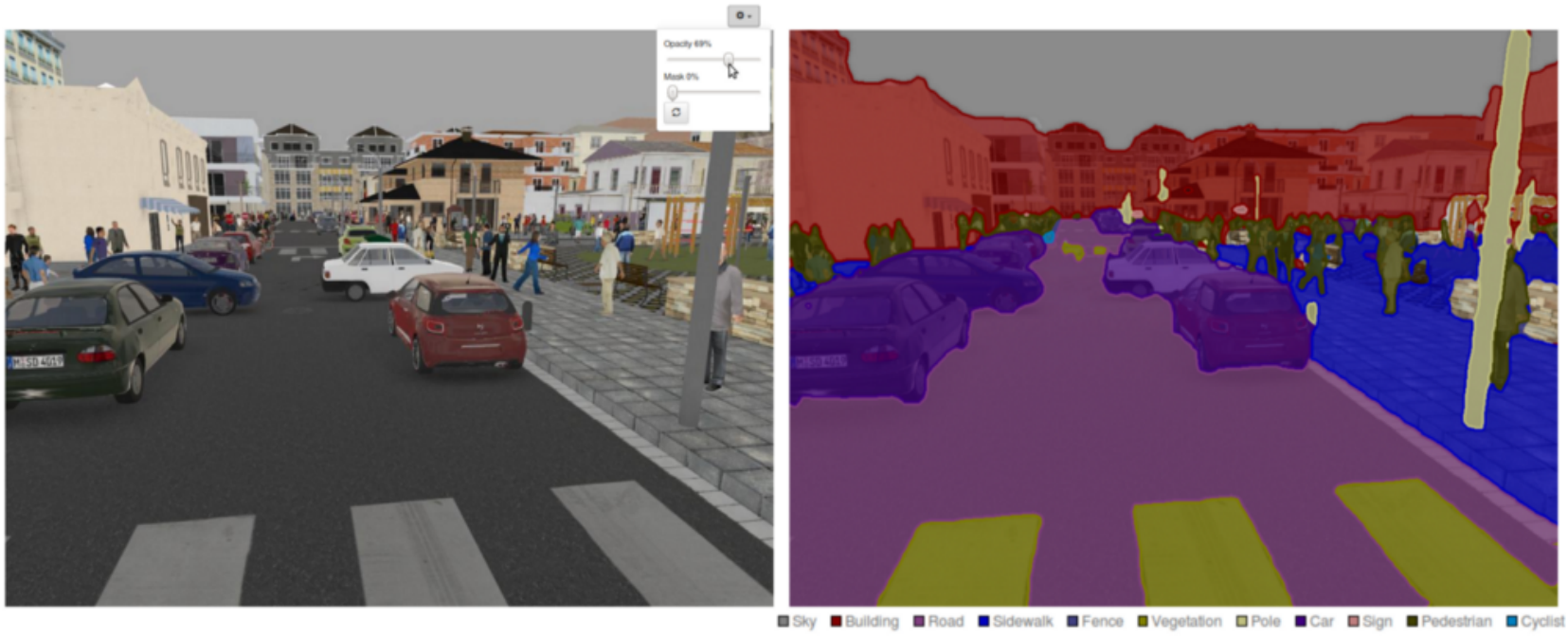
Latest research papers tend to give results for the COCO dataset only. For COCO, AP is the average over multiple IoU (the minimum IoU to consider a positive match). **AP@[.5:.95]** corresponds to the average AP for IoU from 0.5 to 0.95 with a step size of 0.05. For the COCO competition, AP is the average over 10 IoU levels on 80 categories (AP@[.50:.05:.95]: start from 0.5 to 0.95 with a step size of 0.05).

Here is the AP result for the YOLOv3 detector.

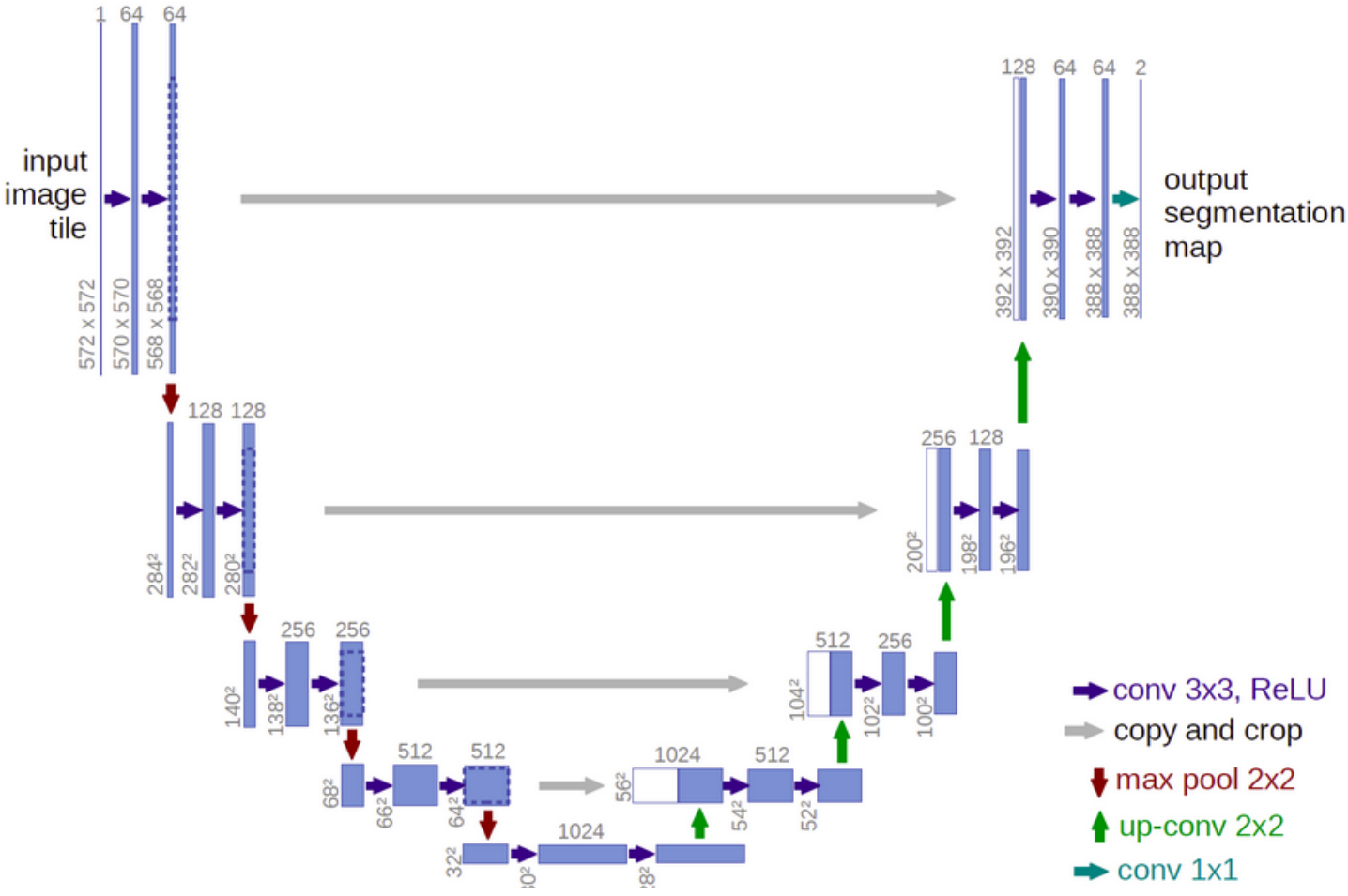
	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [3]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [6]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [4]	Inception-ResNet-v2 [19]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [18]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [13]	DarkNet-19 [13]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [9, 2]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [2]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [7]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [7]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

COCO for YOLOv3

Semantic segmentation



Unet


input image tile
output segmentation map
conv 3x3, ReLU
copy and crop
max pool 2x2
up-conv 2x2
conv 1x1

LinkNet

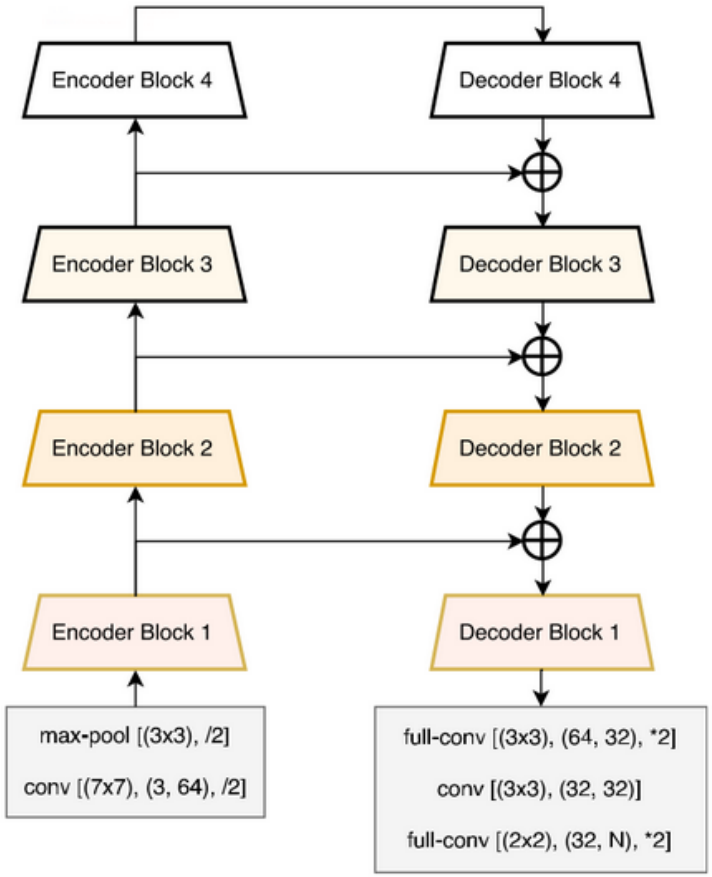
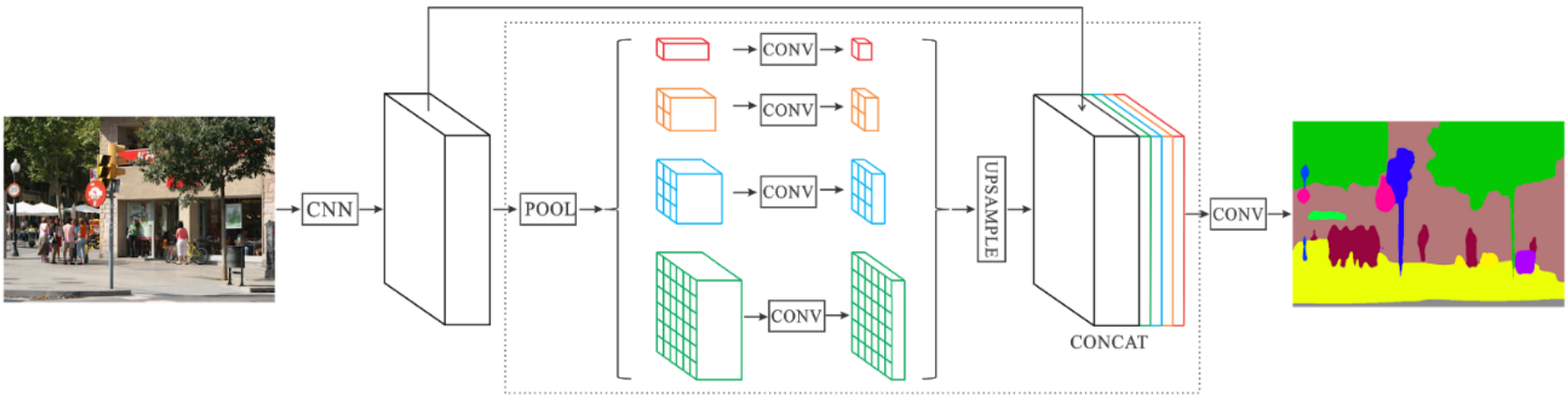


Fig. 1: LinkNet Architecture

PSPNET



(a) Input Image (b) Feature Map (c) Pyramid Pooling Module (d) Final Prediction