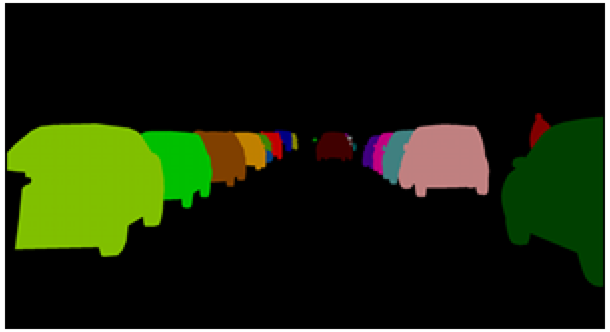
What do we really need

Semantic segmentation

Instance segmentation



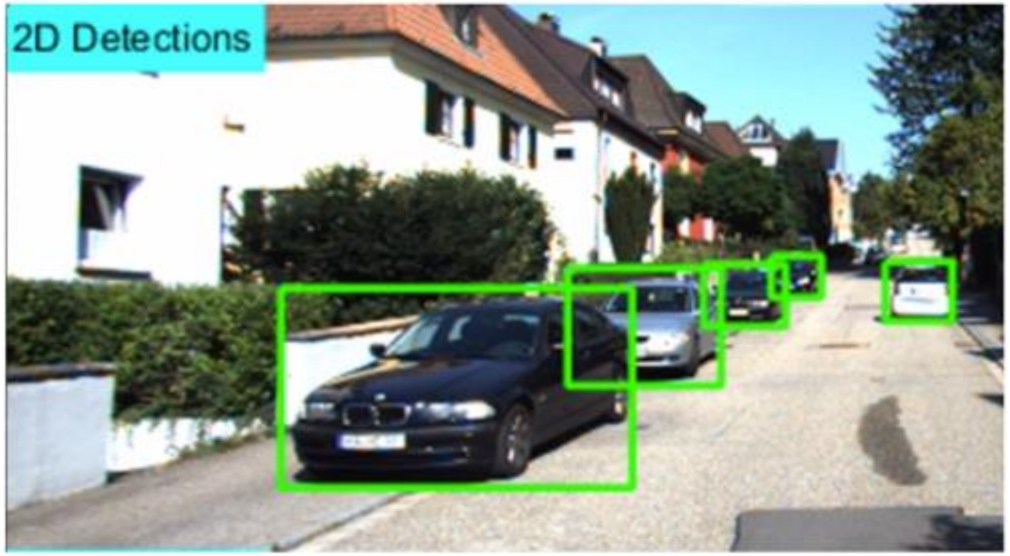
Panoptic segmentation

Image perception beyond classification

- Format 1: semantic segmentation

Image perception beyond classification

- Format 2: object detection



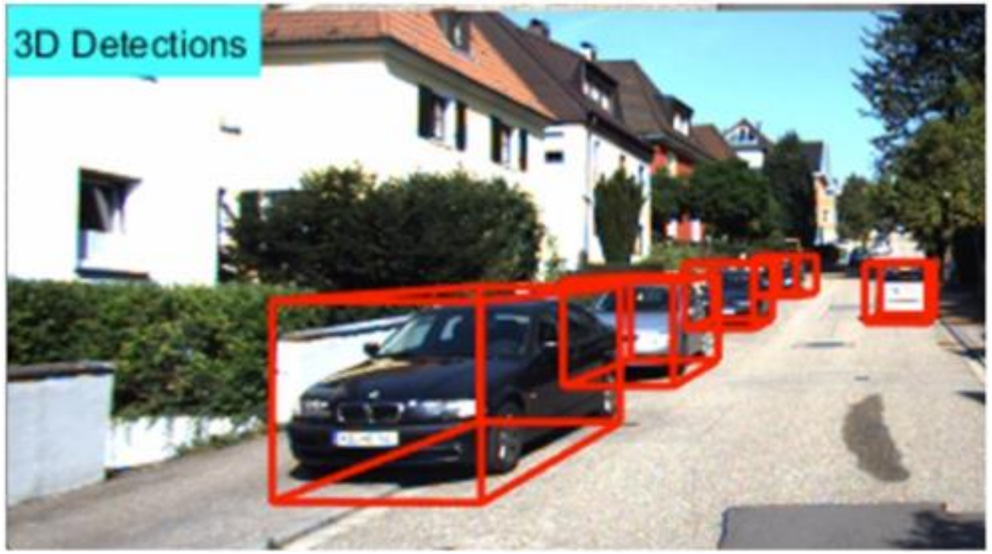
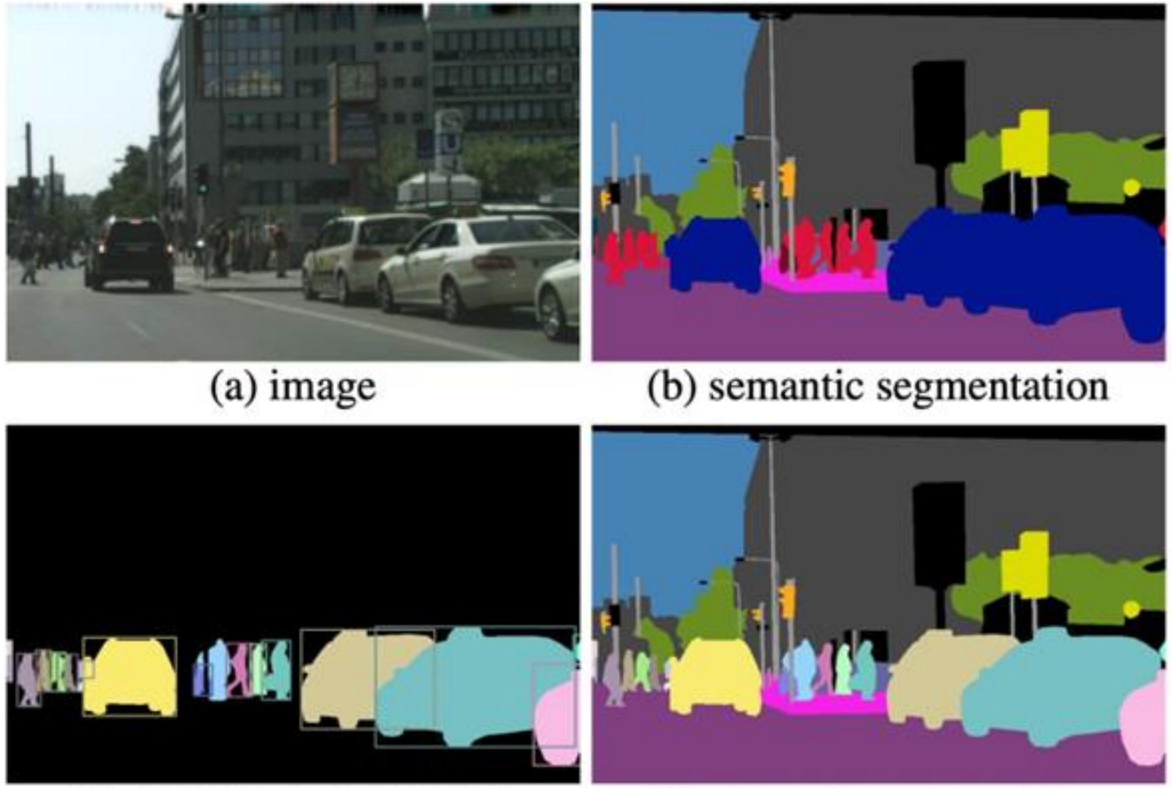


Image perception beyond classification

- Format 3: instance segmentation

Panoptic segmentation

- Different approaches for “things” and “stuff”



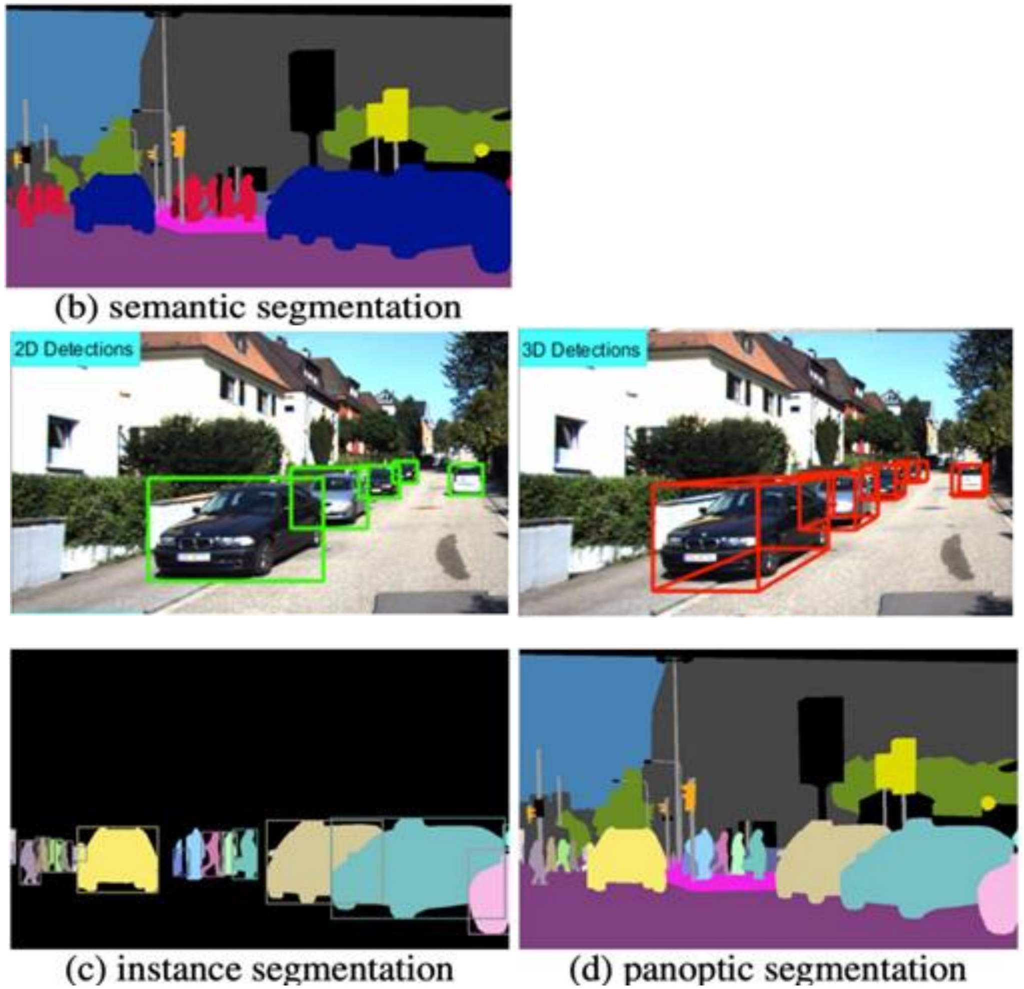
(a) image (b) semantic segmentation

(c) instance segmentation (d) panoptic segmentation

[Kirillov et al. “Panoptic Segmentation” CVPR 2019]

Answering the "where" question

- Semantic segmentation:
  - Faster / easier than instance segmentation
  - Allows "complete" explanation
  - Merges instances
  - Suitable for "staff" and "things"
- Object detection
  - Faster / easier than instance segmentation
  - Distinguishes instances
  - Inaccurate for some classes
  - Incomplete
  - Suitable for "things"
- Instance / Panoptic segmentation
  - Complete
  - Distinguish instances
  - Accurate
  - Harder / slower



(b) semantic segmentation
(c) instance segmentation
(d) panoptic segmentation

Detection vs classification

Detection is harder than classification:

- Localization errors
- Huge class disbalance
- Variation in scale and aspect ratio’s
- Tricky occlusions, including “intra”-occlusions

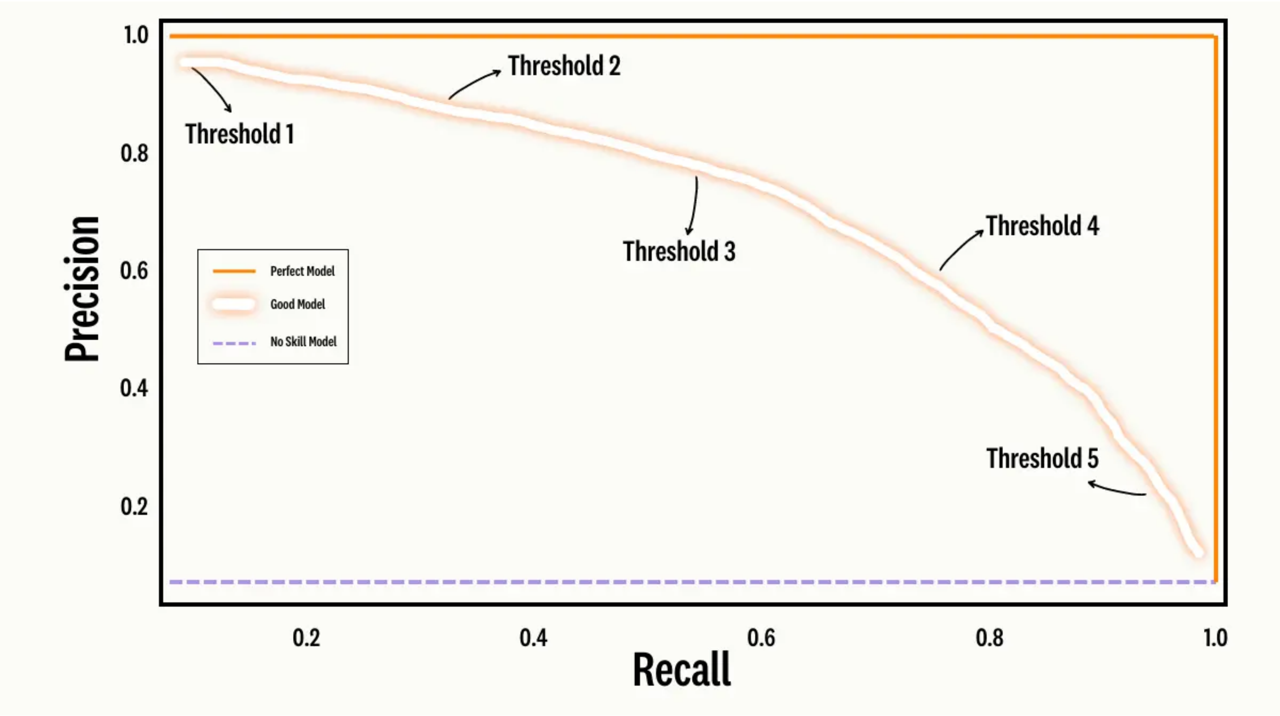
Intersection-over-Union measure

Common criterion for correct boxes:

Intersection / Union > threshold

threshold – 0.5 or 0.7+

Average precision

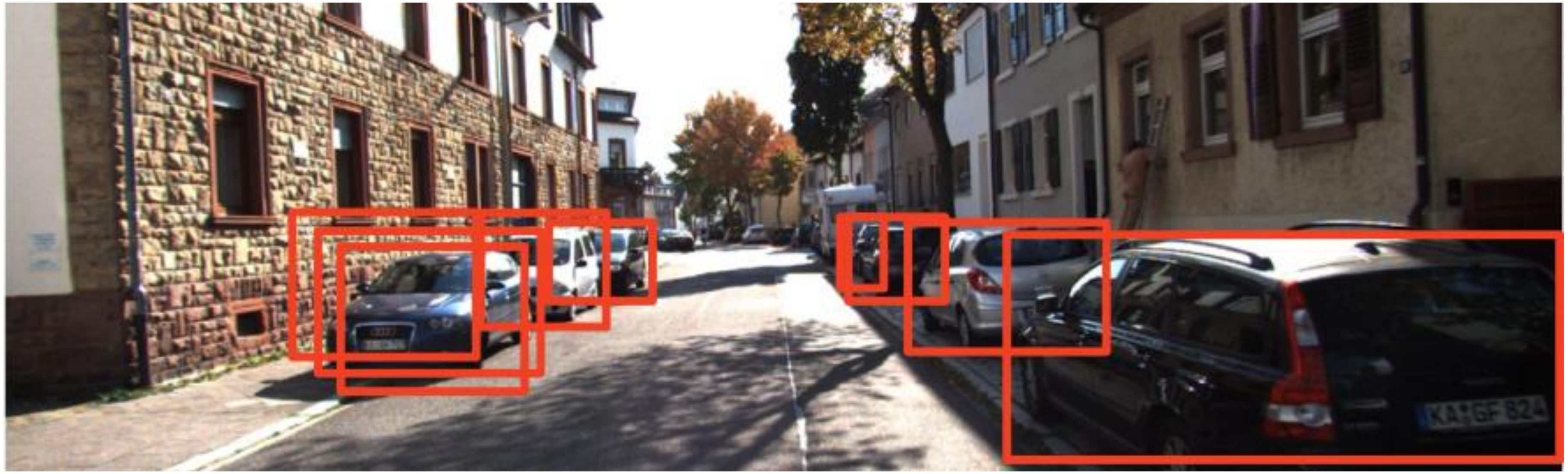


- AP@T – AP with T as threshold on iou
- mAP – averaging over all classes
- AP@0.5:0.95 – averaging over different thresholds on iou

Double detection

- Double detection of the same object is penalized as false positive

Non-maximum suppression



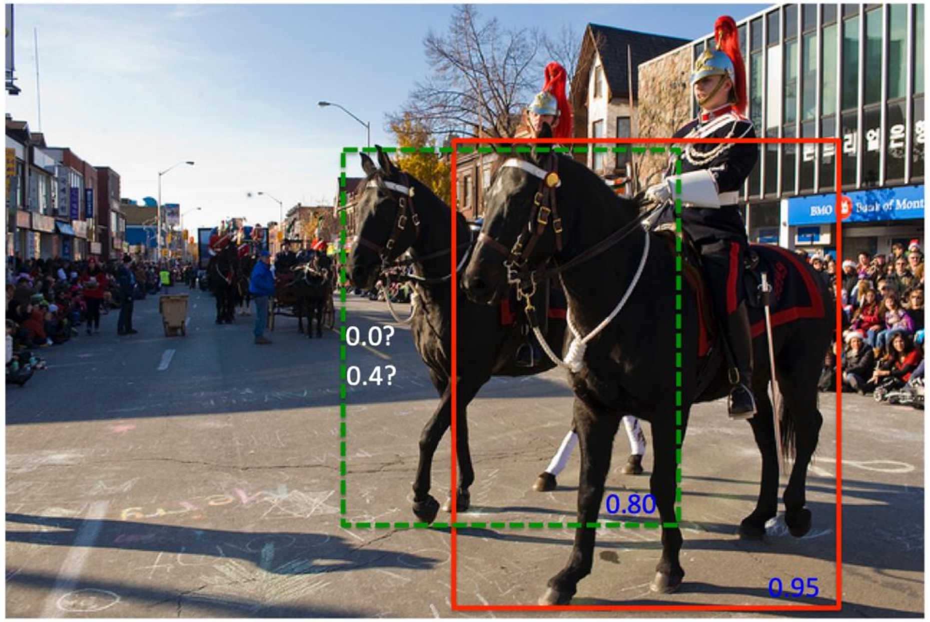
Input: set of detections ({Bi, si})

1) Sort in the descending order of si
2) For i = 1 to N
3) Pick the bounding box i
4) Suppress all subsequent boxes with IoU > T

Soft-NMS

Input: set of detections ($\{B_i, s_i\}$)

1) Sort in the descending order of s_i
2) For i = 1 to N
3) Pick the bounding box i
4) Update s_j for all subsequent boxes with IoU > T
5) Reorder boxes according to updated s_j



Option 1: $$s_i = \begin{cases} s_i, & \text{iou}(\mathcal{M}, b_i) < N_t \\ s_i(1 - \text{iou}(\mathcal{M}, b_i)), & \text{iou}(\mathcal{M}, b_i) \geq N_t \end{cases},$$

Option 2: $$s_i = s_i e^{-\frac{\text{iou}(\mathcal{M}, b_i)^2}{\sigma}}, \forall b_i \notin \mathcal{D}$$

[Bodla et al. "Soft-NMS -- Improving Object Detection With One Line of Code", ICCV 2017]

Soft-NMS

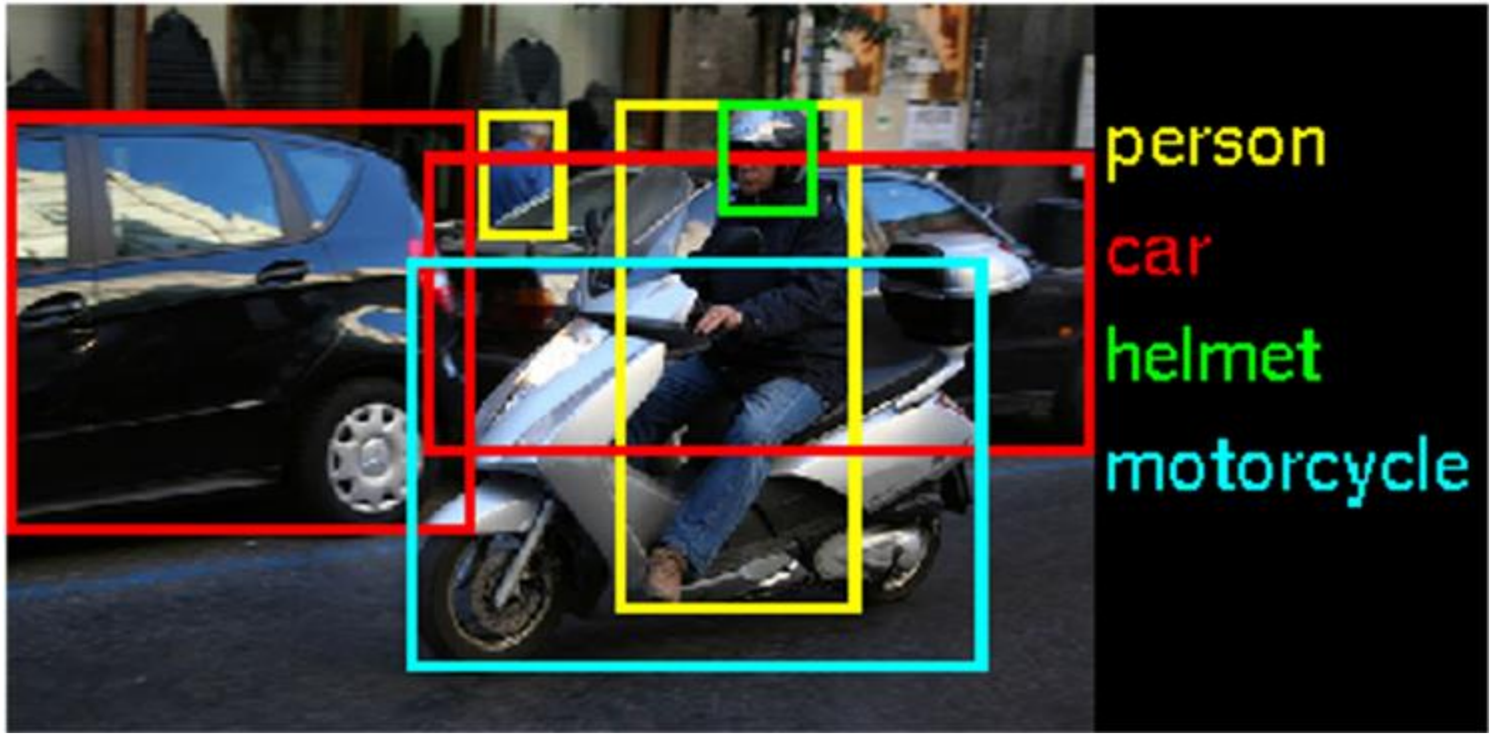
Method	Training data	Testing data	AP 0.5:0.95	AP @ 0.5	AP small	AP medium	AP large	Recall @ 10	Recall @ 100
R-FCN [16]	train+val35k	test-dev	31.1	52.5	14.4	34.9	43.0	42.1	43.6
R-FCN + S-NMS G	train+val35k	test-dev	**32.4**	**53.4**	**15.2**	**36.1**	**44.3**	**46.9**	**52.0**
R-FCN + S-NMS L	train+val35k	test-dev	32.2	**53.4**	15.1	36.0	44.1	46.0	51.0
F-RCNN [24]	train+val35k	test-dev	24.4	45.7	7.9	26.6	37.2	36.1	37.1
F-RCNN + S-NMS G	train+val35k	test-dev	**25.5**	46.6	**8.8**	**27.9**	**38.5**	**41.2**	45.3
F-RCNN + S-NMS L	train+val35k	test-dev	**25.5**	**46.7**	**8.8**	**27.9**	38.3	40.9	**45.5**
D-RFCN [3]	trainval	test-dev	37.4	59.6	17.8	40.6	51.4	46.9	48.3
D-RFCN S-NMS G	trainval	test-dev	**38.4**	**60.1**	**18.5**	**41.6**	**52.5**	**50.5**	**53.8**
D-RFCN + MST	trainval	test-dev	39.8	62.4	22.6	42.3	52.2	50.5	52.9
D-RFCN + MST + S-NMS G	trainval	test-dev	**40.9**	**62.8**	**23.3**	**43.6**	**53.3**	**54.7**	**60.4**

Table 1. Results on MS-COCO test-dev set for R-FCN, D-RFCN and Faster-RCNN (F-RCNN) which use NMS as baseline and our proposed Soft-NMS method. G denotes Gaussian weighting and L denotes linear weighting. MST denotes multi-scale testing.

[Bodla et al. "Soft-NMS -- Improving Object Detection With One Line of Code", ICCV 2017]

Multi-class detection

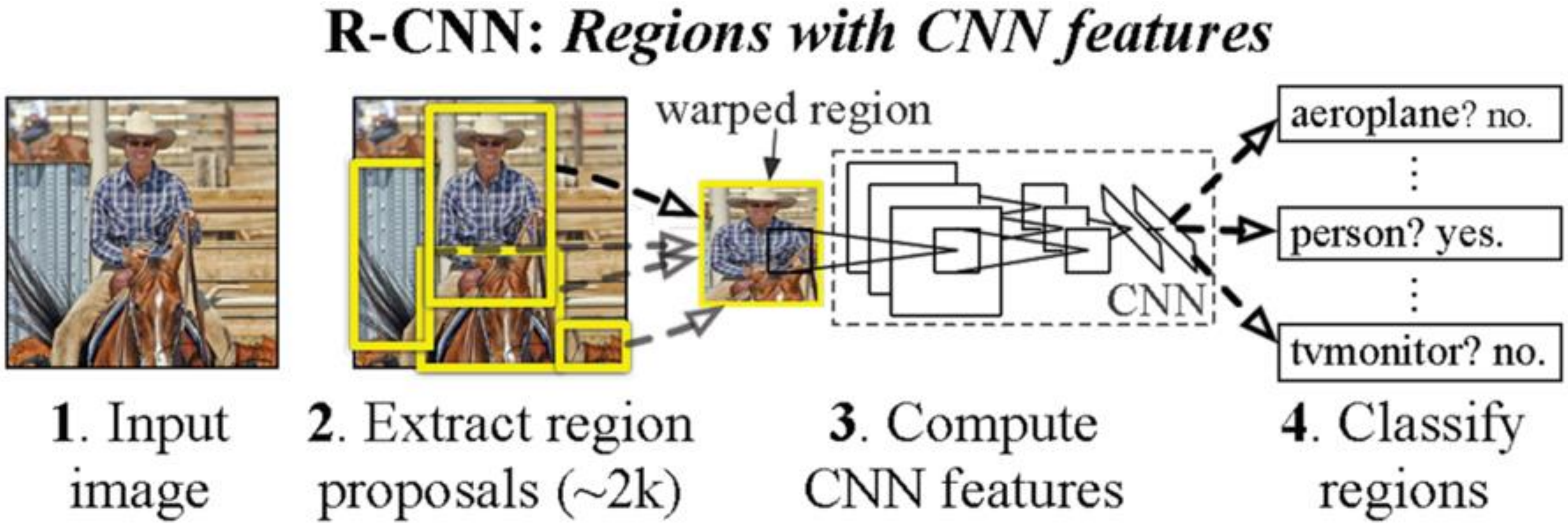
- Lots of research is going towards object detection for a large number of classes



General ideas for object detection

- **Sliding-window:** use binary classification to classify every possible subwindow (infeasible with DL)
- **Region proposal:** pick a subset of prospective regions and score them with a binary classifier
- **Bounding box regression:** predict the coordinates of the boxes as real-valued variables

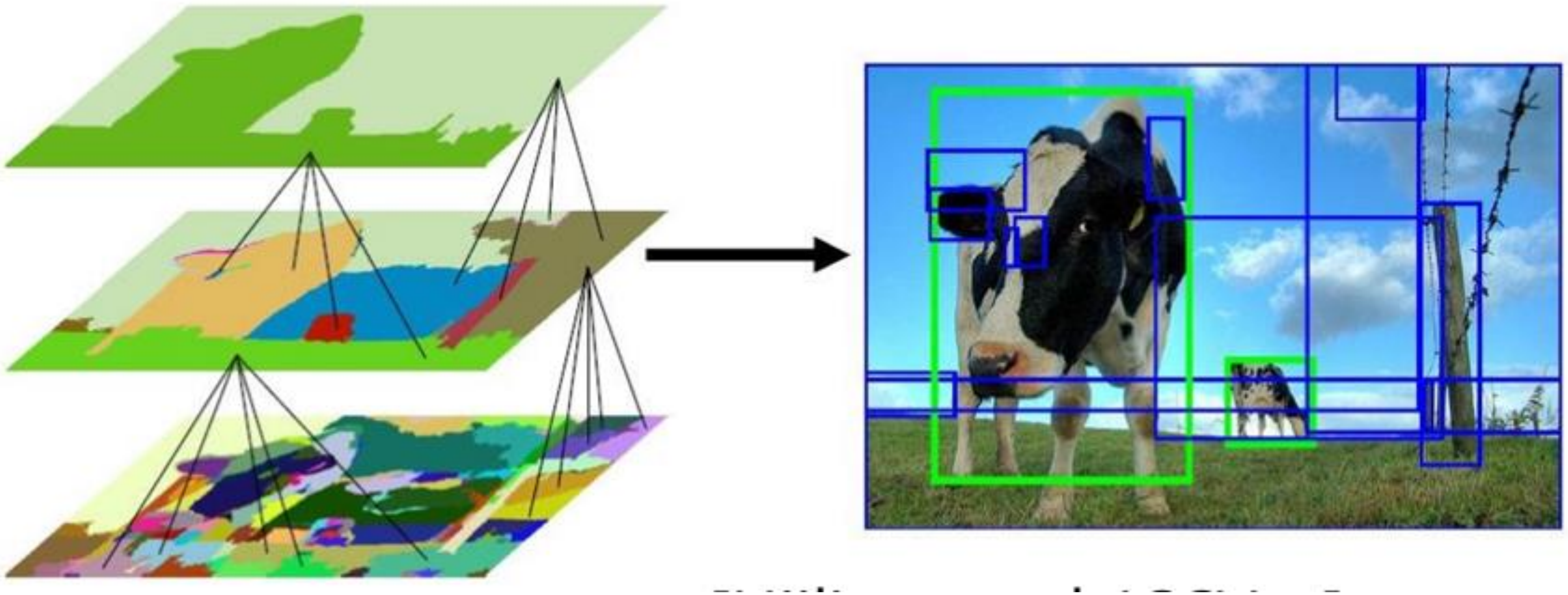
R-CNN framework



- Use an external box proposal method
- Fine-tune the ConvNet to score proposal

[Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014]

Example source of external proposals

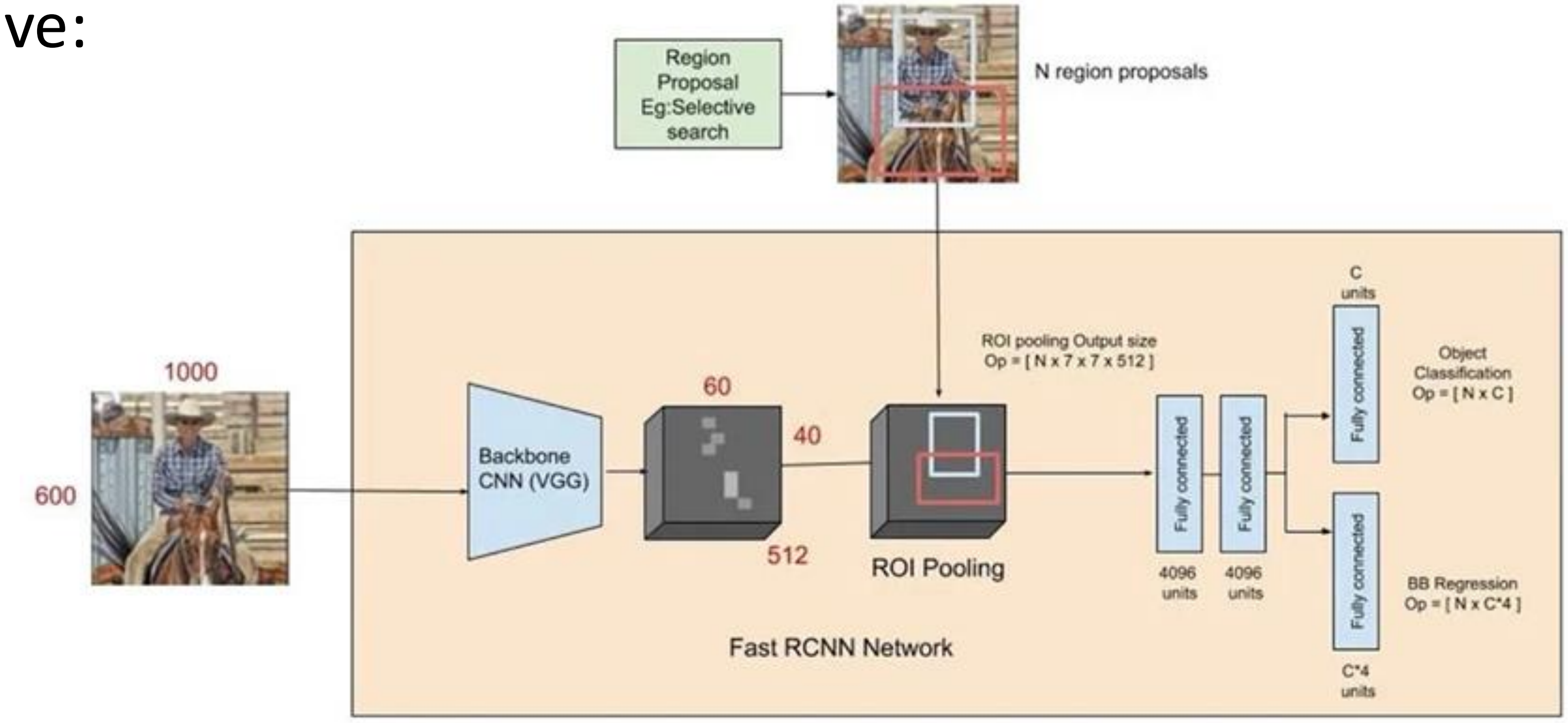


- Graph-based hierarchical segmentation based on maximum-spanning trees

[Van de Sande et al. "Segmentation as selective search for object recognition.", ICCV 2011]

Fast R-CNN

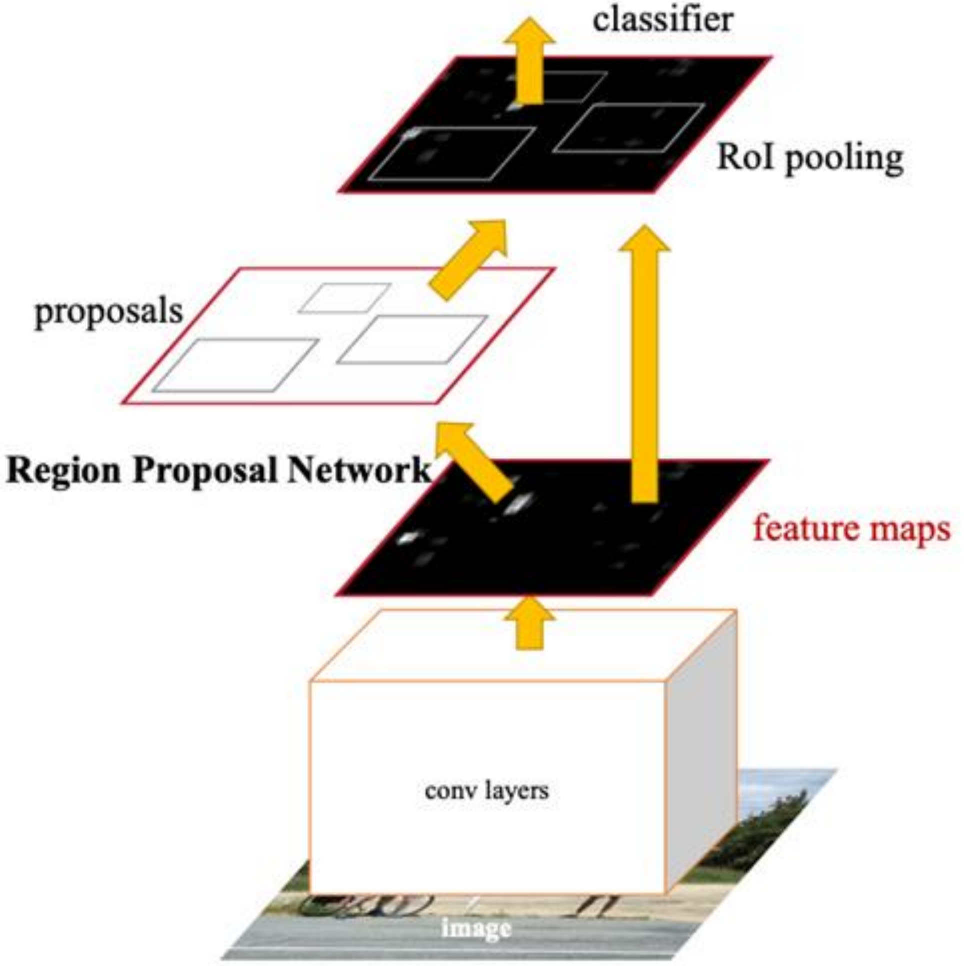
- Processing lots of overlapping boxes is inefficient
- Alternative:



[Ross Girshick "Fast R-CNN", ICCV 2015]

Faster R-CNN

- Key novelty: the proposals come from “sparse sliding window search”



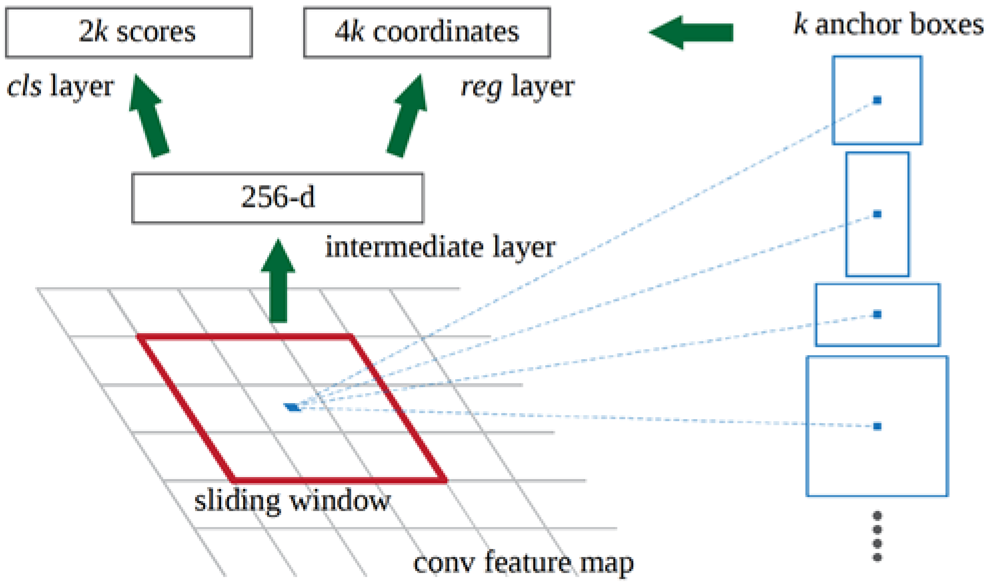
[Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks“, TPAMI 2016]

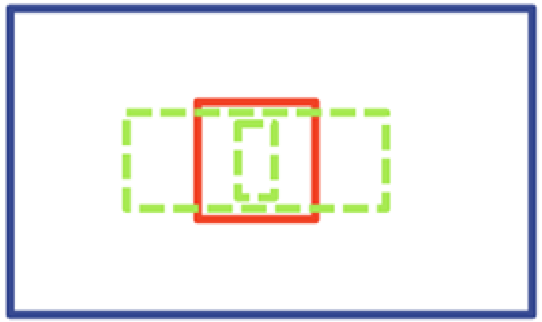
Region Proposal Network

- Two types of heads:
  - Anchor classification
  - Regression on encoded box parameters:

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a,$$
$$t_w = \log(w/w_a), \quad t_h = \log(h/h_a),$$

- Anchors:
  - Different scale (3 scales in paper)
  - Different aspect ratio (3 aspect ratios in the paper)
- Gt assignment based on IoU

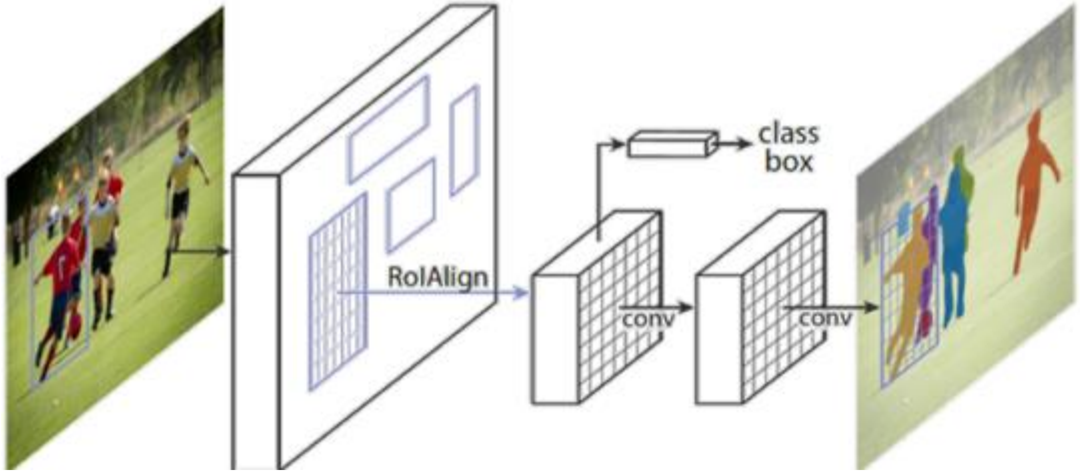




[Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", TPAMI 2016]

Extension for Instance Segmentation

- Mask R-CNN: adding mask prediction



- Masks for different classes are predicted and scored independently (decoupling classification and segmentation)
- Top-down approach to instance segmentation

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

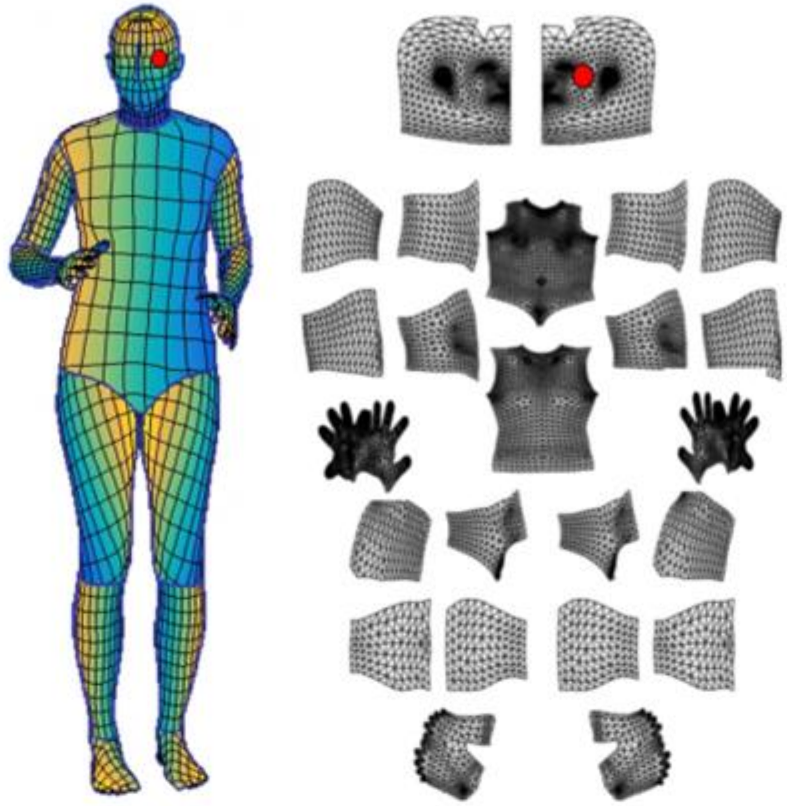
Mask R-CNN results

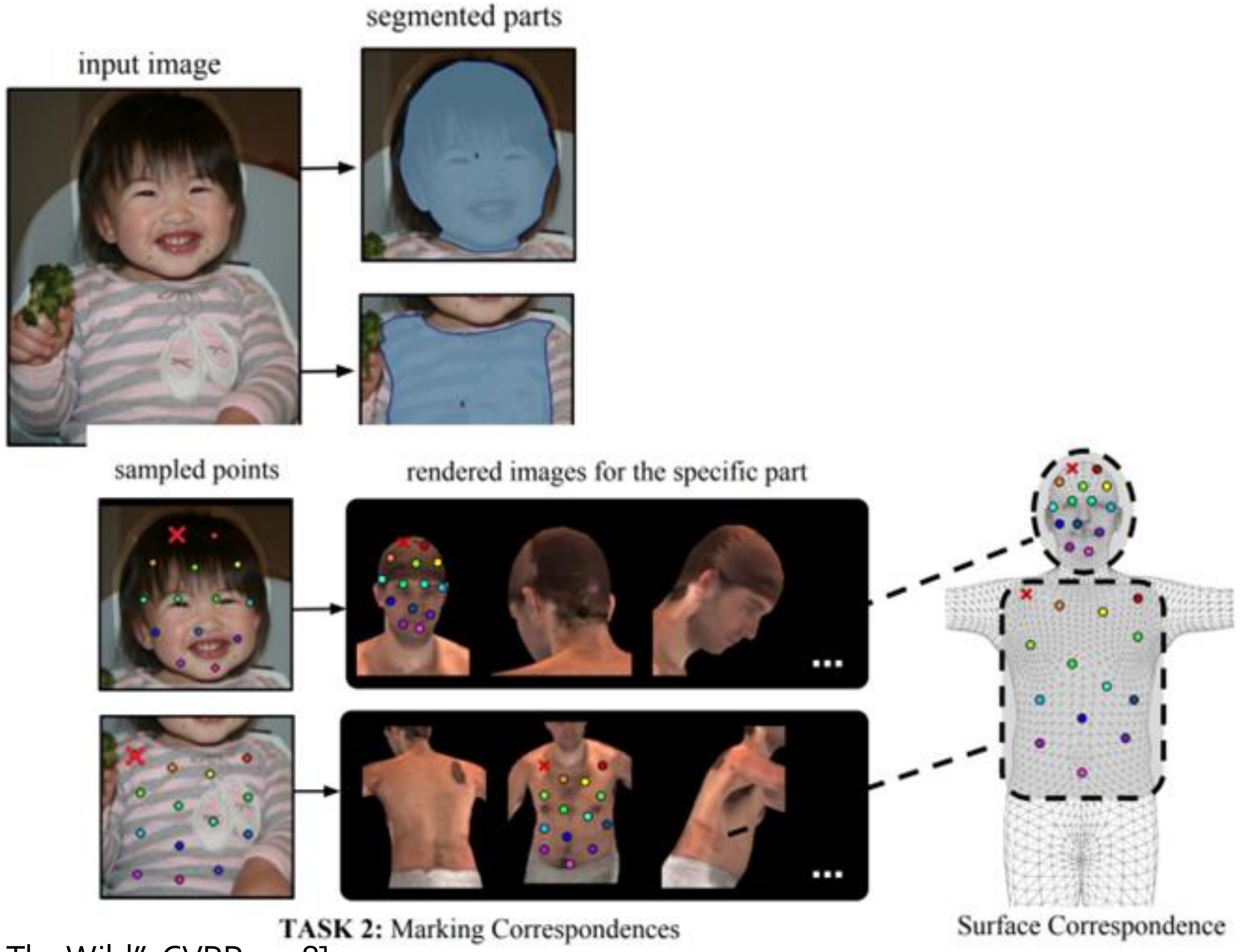
[He et al. "Mask R-CNN", ICCV 2017]

DensePose: closer loop at people

[Guler et al. "DensePose: Dense Human Pose Estimation In The Wild", CVPR 2018]

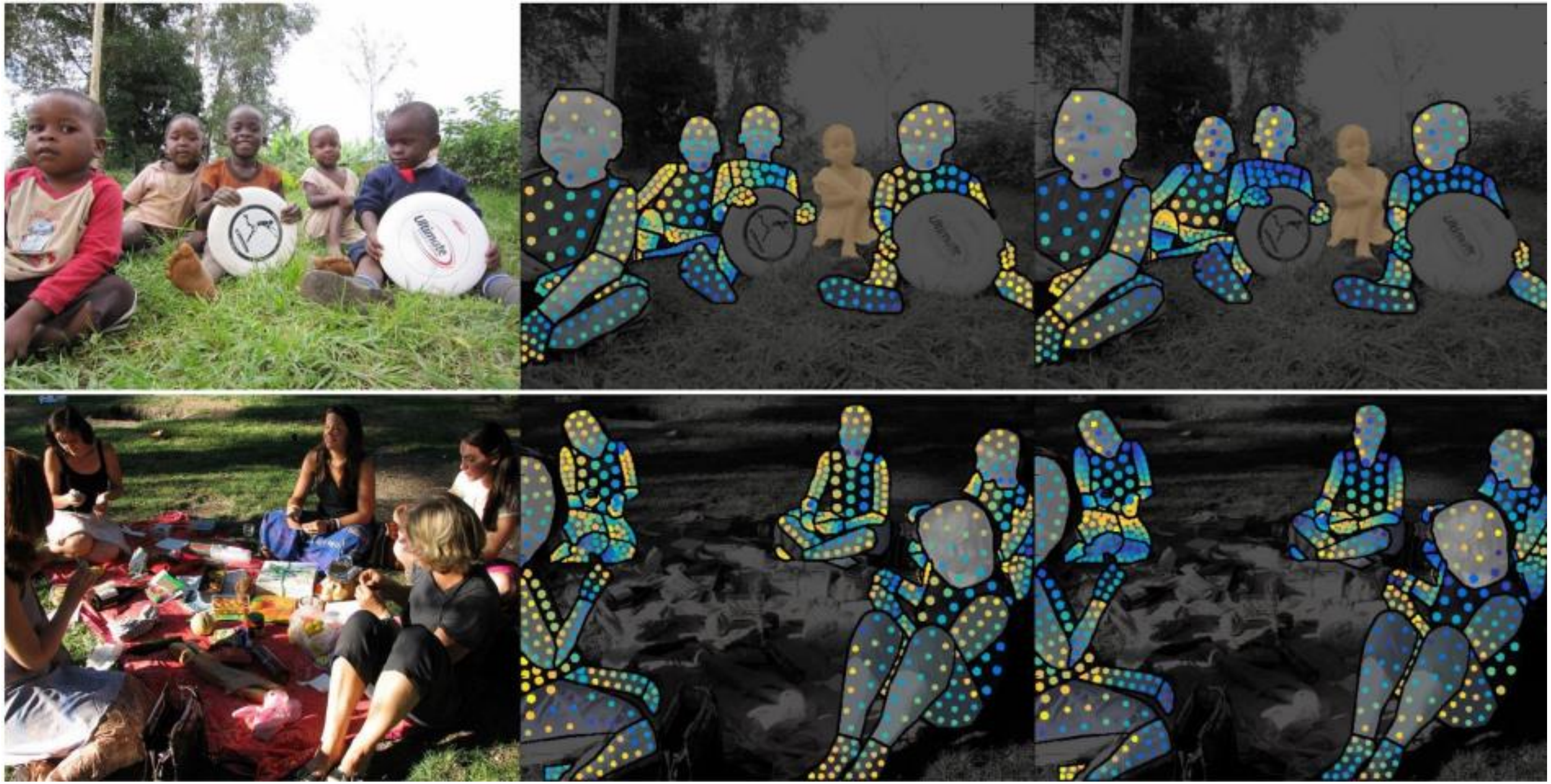
DensePose: format





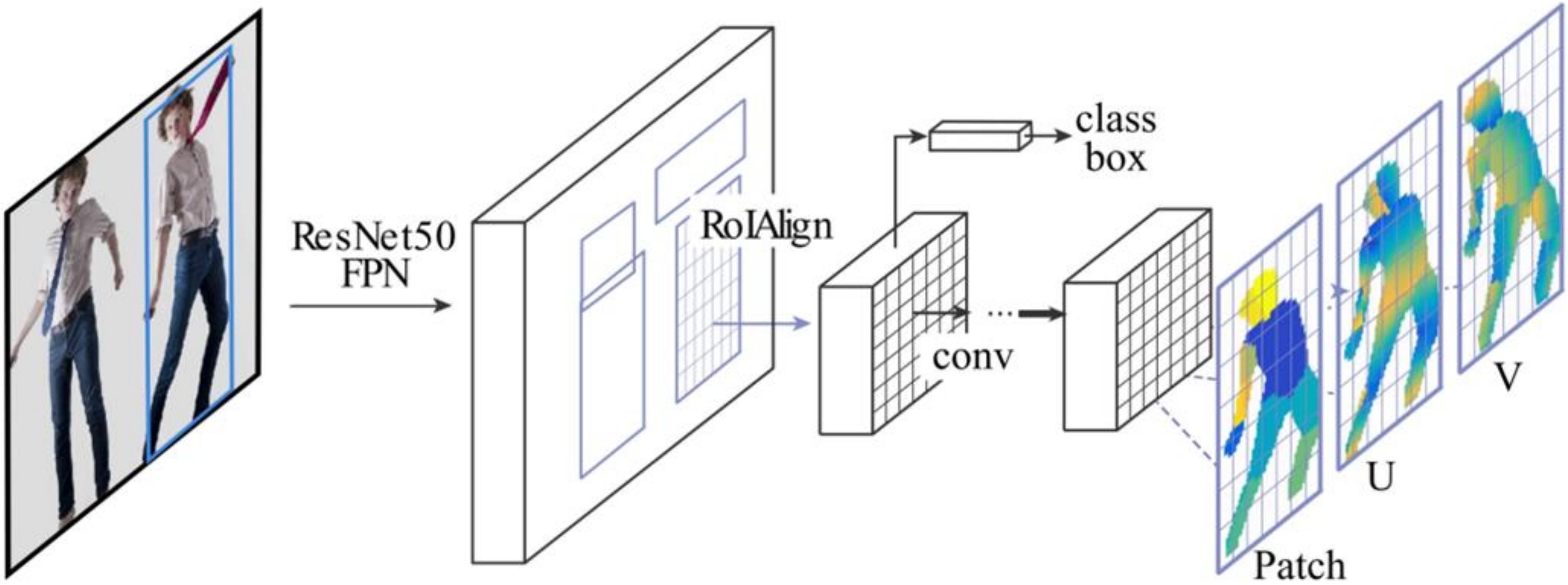
[Guler et al. “DensePose: Dense Human Pose Estimation In The Wild”, CVPR 2018]

DensePose: example annotations



[Guler et al. "DensePose: Dense Human Pose Estimation In The Wild", CVPR 2018]

DensePose: prediction

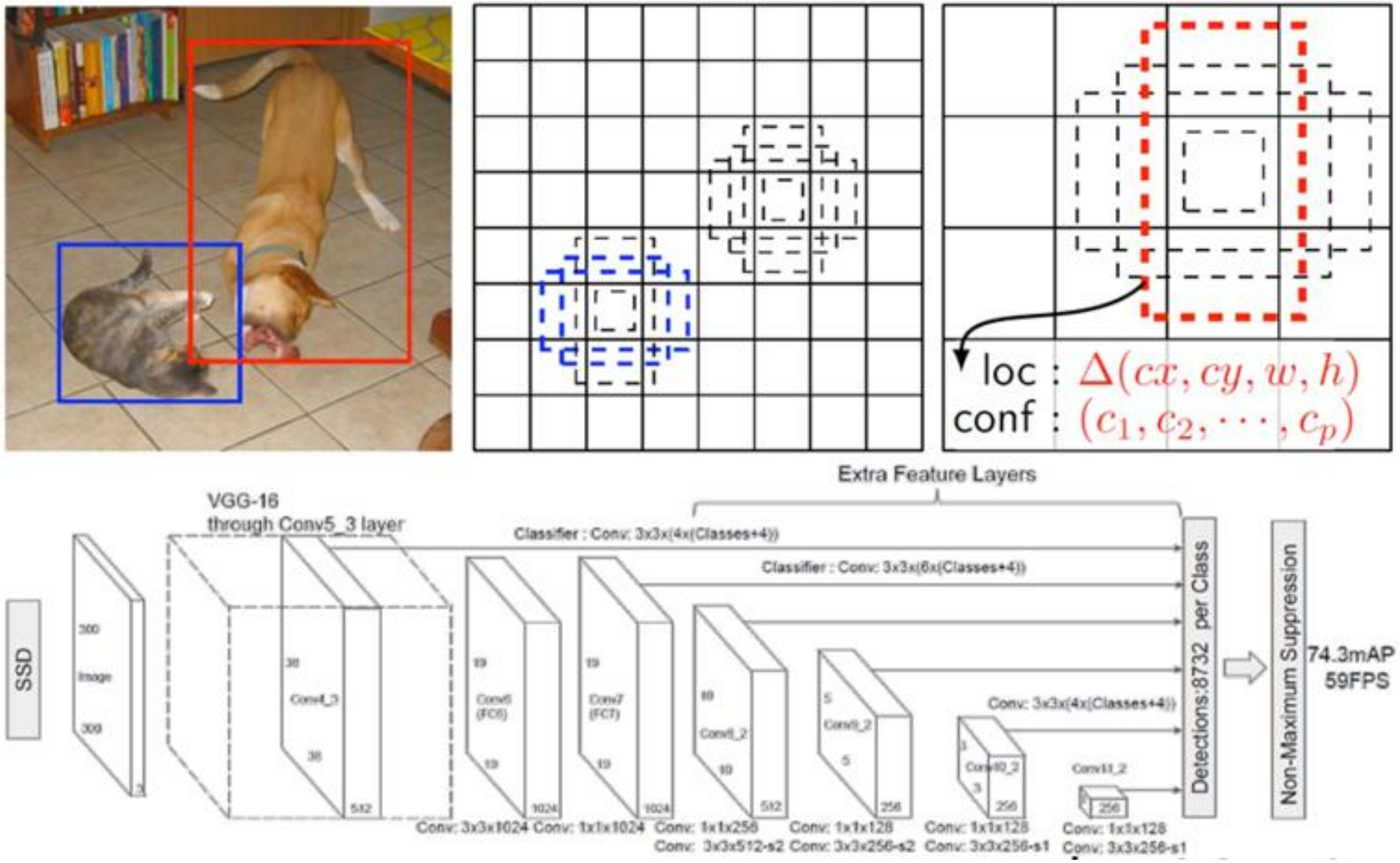


[Guler et al. "DensePose: Dense Human Pose Estimation In The Wild", CVPR 2018]

DensePose

[Guler et al. "DensePose: Dense Human Pose Estimation In The Wild", CVPR 2018]

SSD: Single-shot detector



1. One-stage detection: united model for region proposal and classification
2. Anchor boxes on different scales

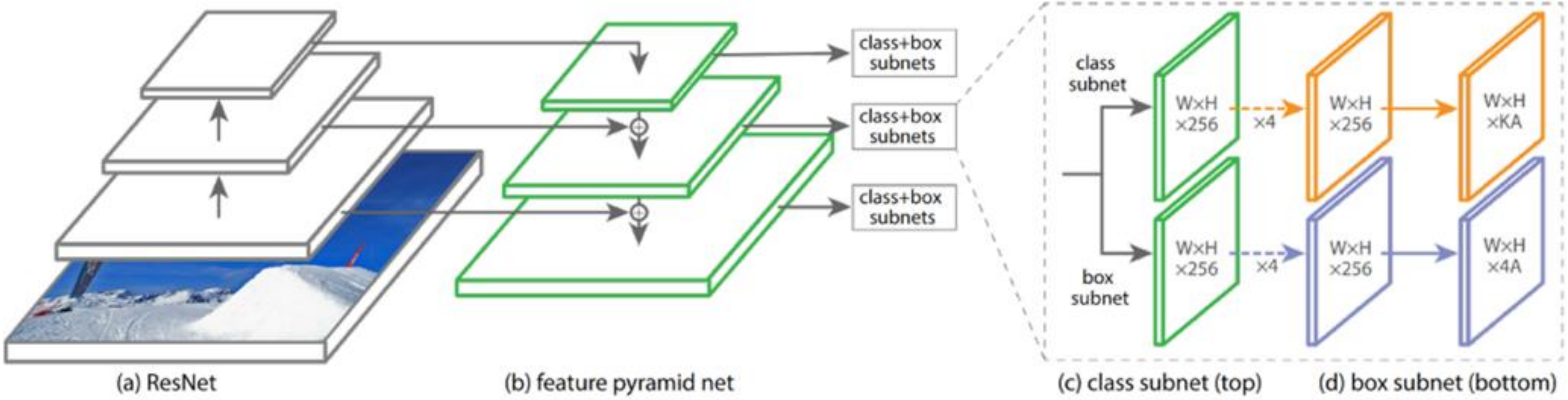
[Liu et al. " Ssd: Single shot multibox detector", ECCV 2016]

Examples: SSD detection

[Liu et al. " Ssd: Single shot multibox detector", ECCV 2016]

RetinaNet: improving one-shot detection

- Fixing neural architecture (better processing to small objects)



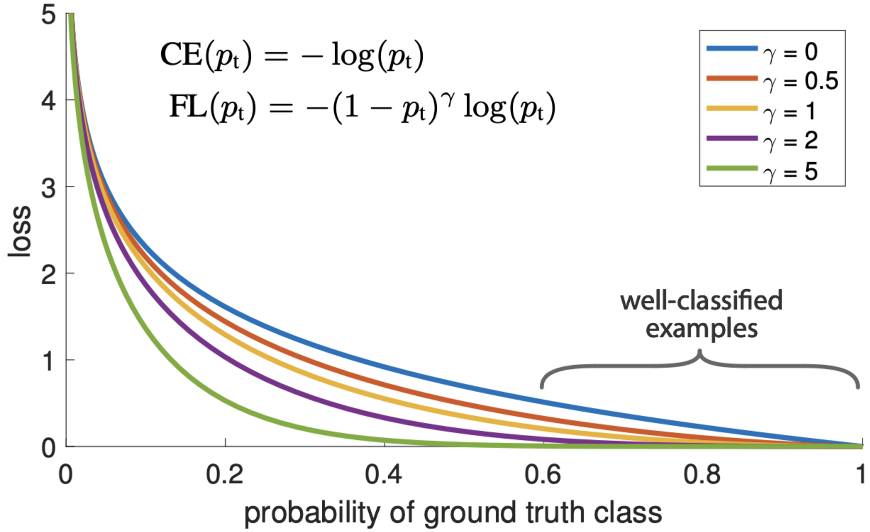
[Ross et al. " Focal loss for dense object detection", ICCV 2017]

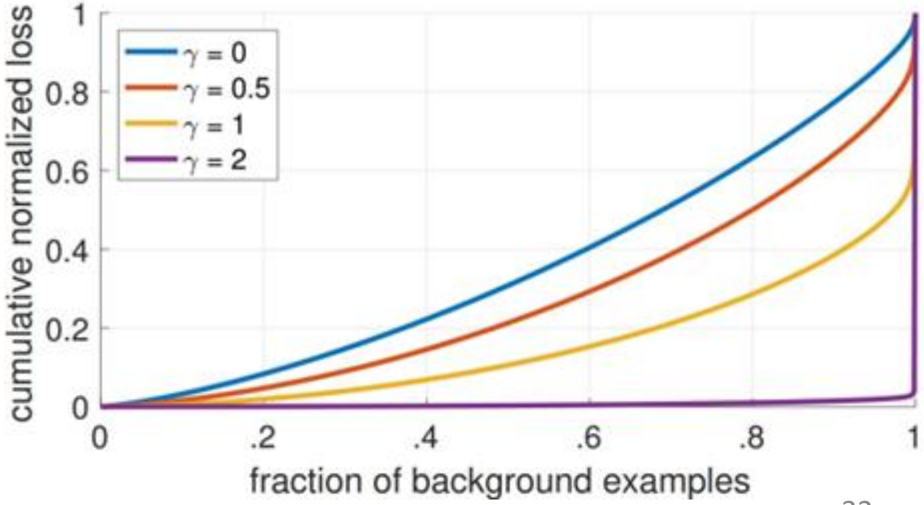
RetinaNet: improving one-shot detection

- New box classification loss (better for background boxes classification)

$$\mathrm{FL}(p, y) = \begin{cases} -\alpha(1-p)^{\gamma} \log p, & \text{if } y = 1 \\ -p^{\gamma} \log(1-p), & \text{otherwise.} \end{cases}$$

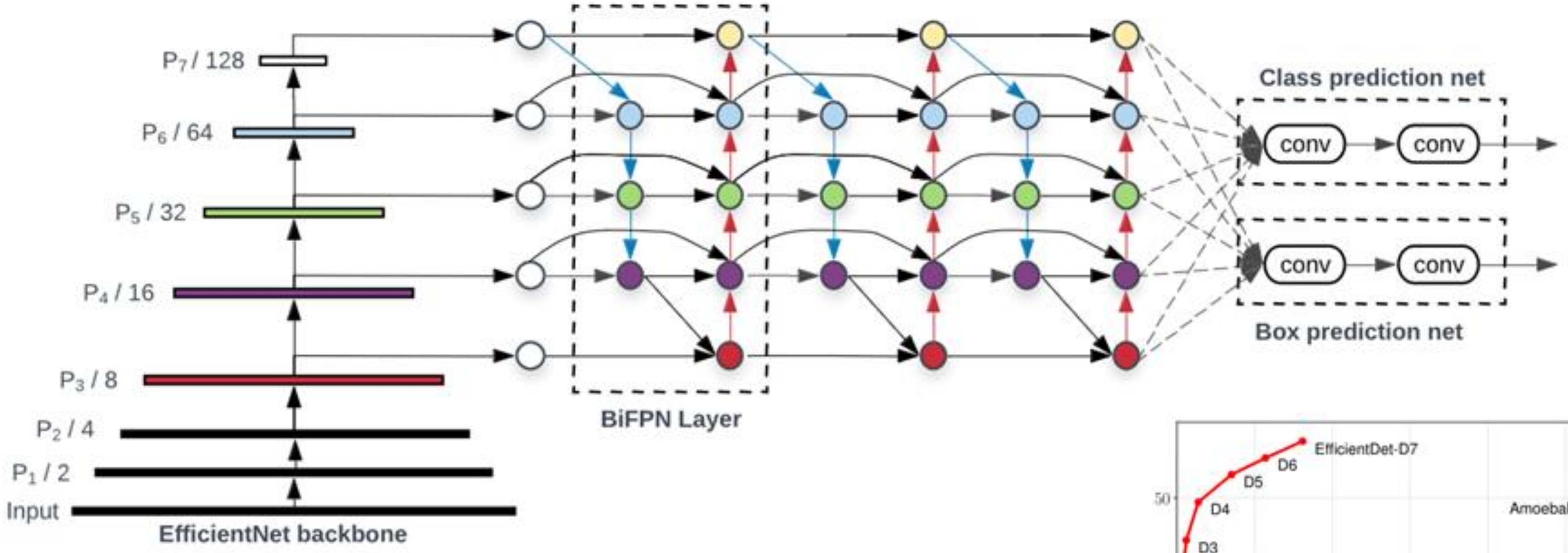
α = 0.25, γ = 2 used in practice

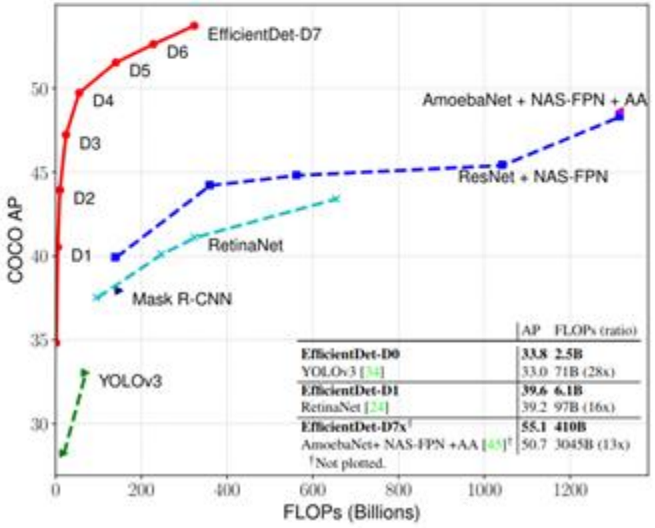




[Ross et al. " Focal loss for dense object detection", ICCV 2017]

EfficientDet

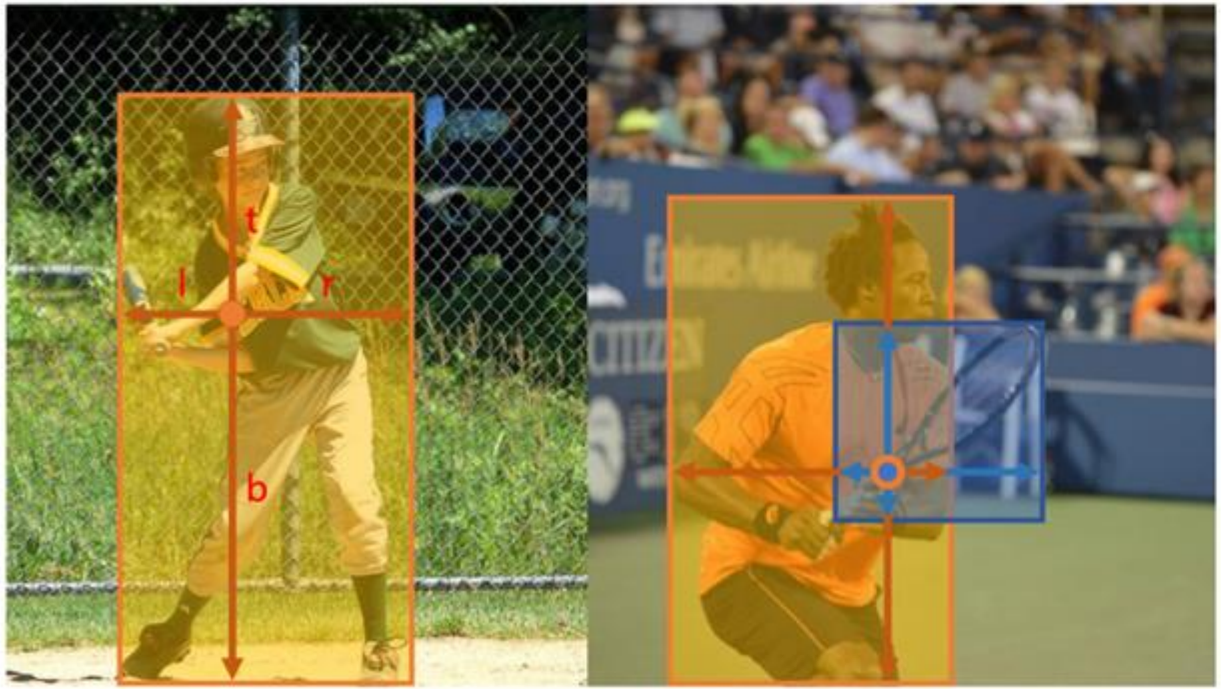




- Better backbone
- Combination of top-down and bottom-up streams with skip-connections

[Tan et al. " Efficientdet: Scalable and efficient object detection", CVPR 2020]

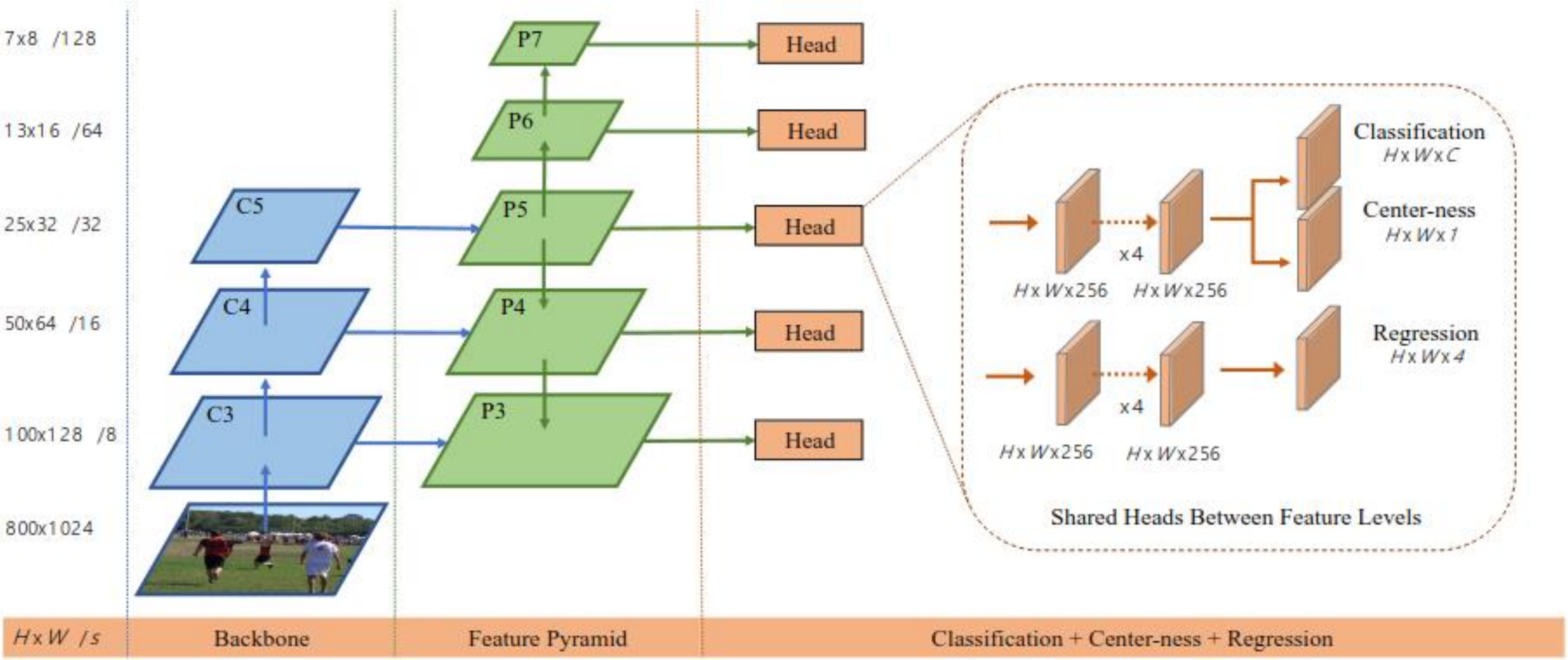
FCOS detector



- At each pixel predicting:
  - C binary labels
  - Offsets
  - Smaller boxes have priority for offsets

[Tian et al. " FCOS: A simple and strong anchor-free object detector", ICCV 2019]

FCOS detector



[Tian et al. " FCOS: A simple and strong anchor-free object detector", ICCV 2019]

FCOS losses

- Loss:

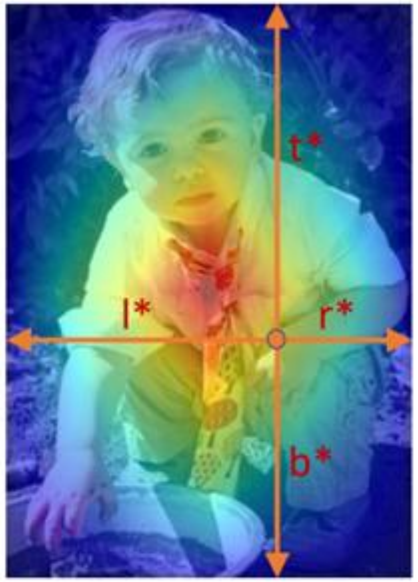
$$L(\{\boldsymbol{p}_{x,y}\},\{\boldsymbol{t}_{x,y}\}) = \frac{1}{N_{\text{pos}}}\sum_{x,y} L_{\text{cls}}(\boldsymbol{p}_{x,y}, c^*_{x,y}) + \frac{\lambda}{N_{\text{pos}}}\sum_{x,y} \mathbb{1}_{\{c^*_{x,y}>0\}} L_{\text{reg}}(\boldsymbol{t}_{x,y}, \boldsymbol{t}^*_{x,y}),$$

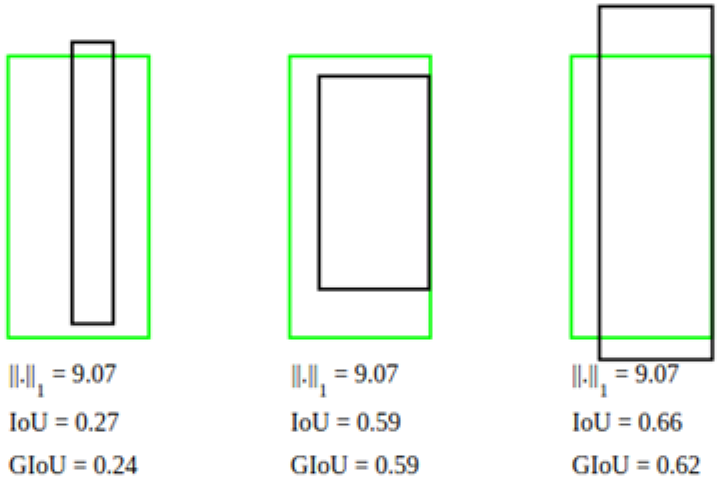
- Centerness:

$$\text{centerness}^* = \sqrt{\frac{\min(l^*, r^*)}{\max(l^*, r^*)} \times \frac{\min(t^*, b^*)}{\max(t^*, b^*)}}.$$

- GIoU:

$$GIoU = IoU - \frac{|C \backslash (A \cup B)|}{|C|} \qquad IoU = \frac{|A \cap B|}{|A \cup B|}$$





[Tian et al. " FCOS: A simple and strong anchor-free object detector", ICCV 2019]