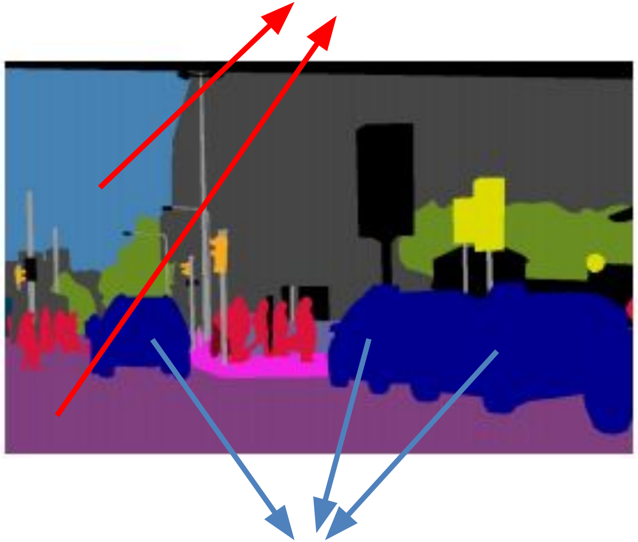
Instance segmentation

Semantic segmentation

Label every pixel, including the background (sky, grass, road)

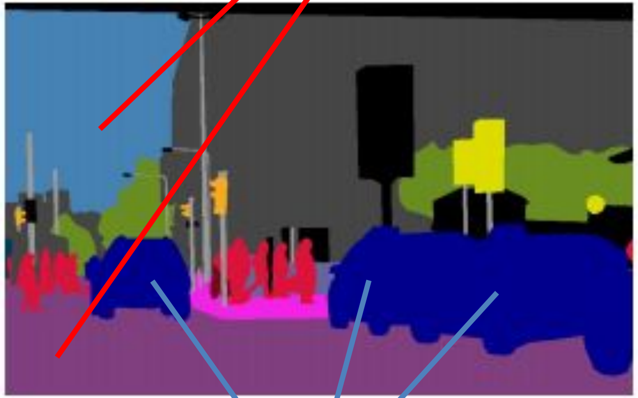


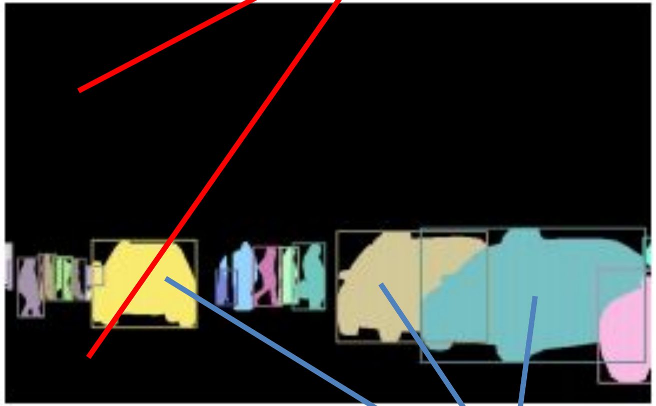
Do not differentiate between the pixels coming from instances of the same class

Instance segmentation

Label every pixel, including the background (sky, grass, road)

Do not label pixels coming from uncountable objects (sky, grass, road)





Do not differentiate between the pixels coming from instances of the same class

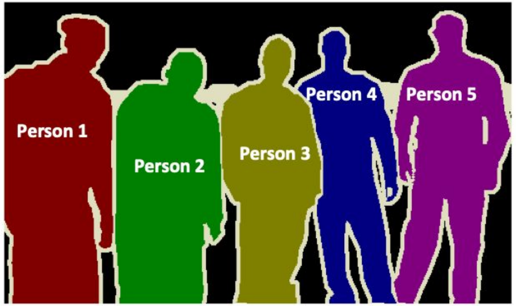
Differentiate between the pixels coming from instances of the same class

Instance segmentation methods

Proposal-based

1. Proposals

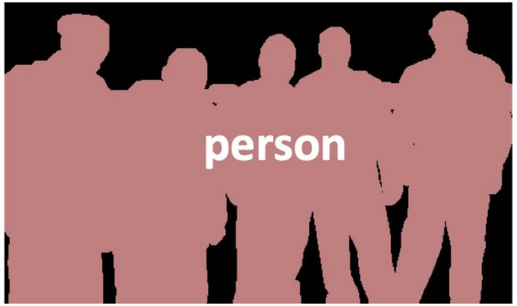
2. Assign a class



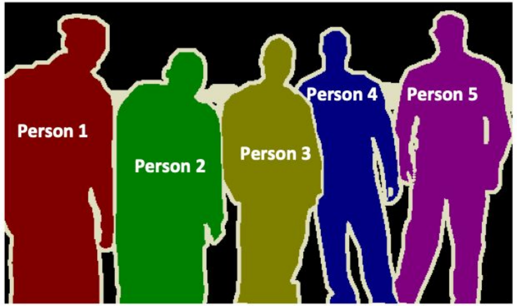
vs.

FCN-based

1. Semantic segmentation



2. Find instances

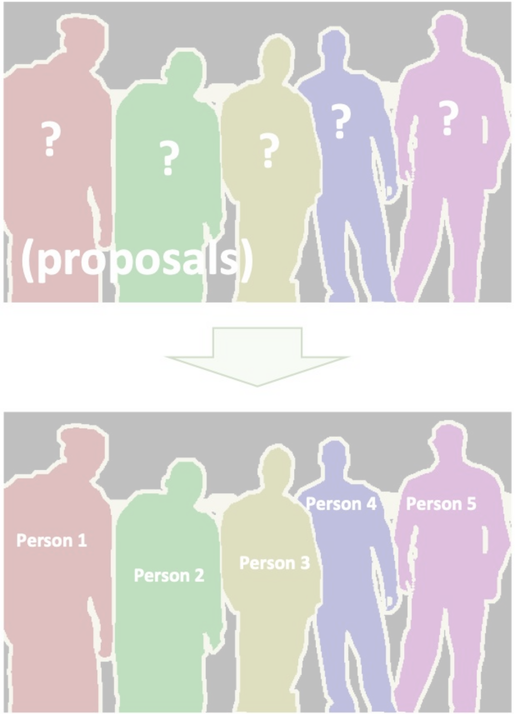


Instance segmentation methods

Proposal-based

1. Proposals

2. Assign a class

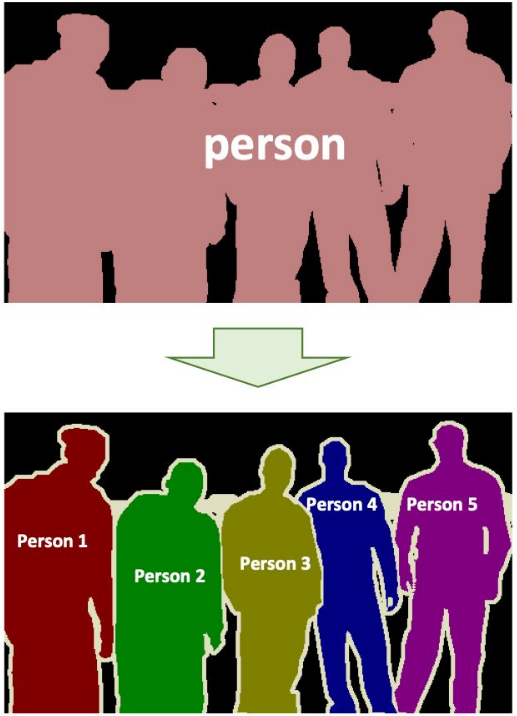


vs.

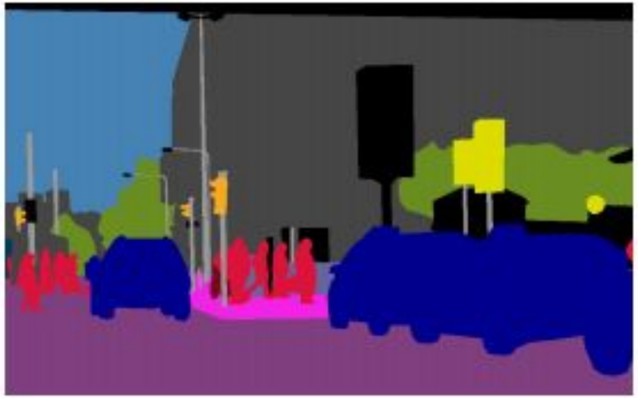
FCN-based

1. Semantic segmentation

2. Find instances



FCN-based methods

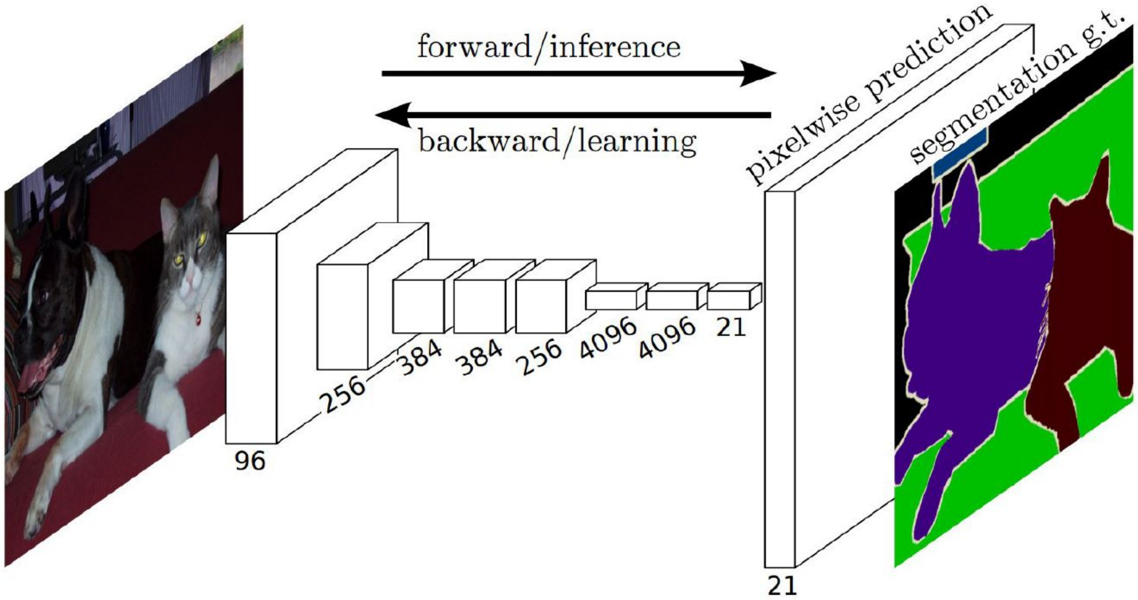


A semantic map

We already know how to obtain this!

Why FCN-based?

- Fully Convolutional Networks for Semantic Segmentation

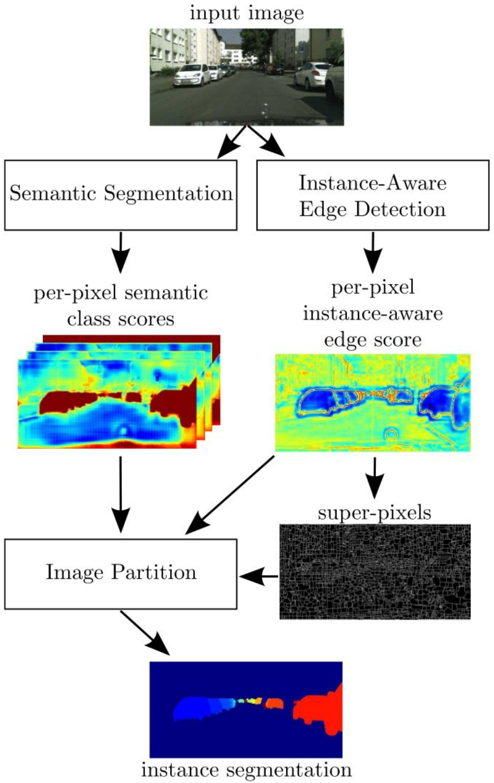


Long, Shelhamer, Darrell - Fully Convolutional Networks for Semantic Segmentation, CVPR 2015, PAMI 2016

FCN-based methods

- X. Liang et al. “Proposal-free Network for Instance-level Object Segmentation“. Arxiv 2015
- A. Kirillov et al. „InstanceCut: from Edges to Instances with MultiCut“. CVPR 2017
- M. Bai and R. Urtasun “Deep Watershed Transform for Instance Segmentation “. CVPR 2017

Instances through clustering



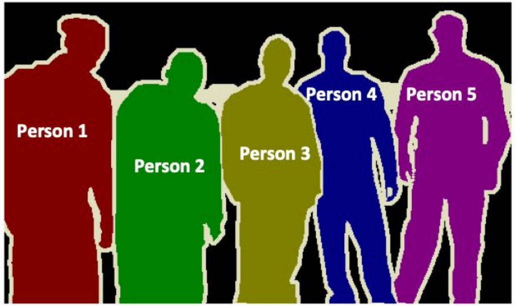
A. Kirillov et al. „InstanceCut: from Edges to Instances with MultiCut". CVPR 2017

Instance segmentation methods

Proposal-based

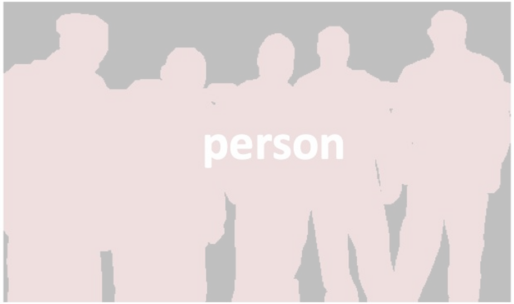
1. Proposals

2. Assign a class

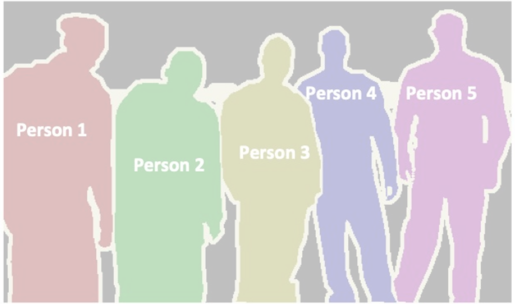


VS.

FCN-based



1. Semantic segmentation

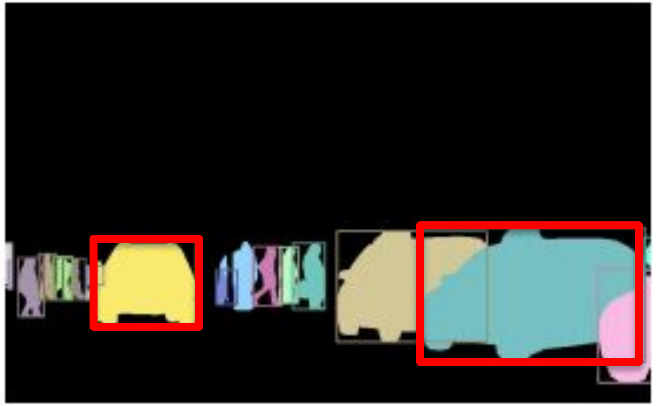


2. Find instances

Proposal-based methods

Bounding boxes.....

We already know how to obtain those!

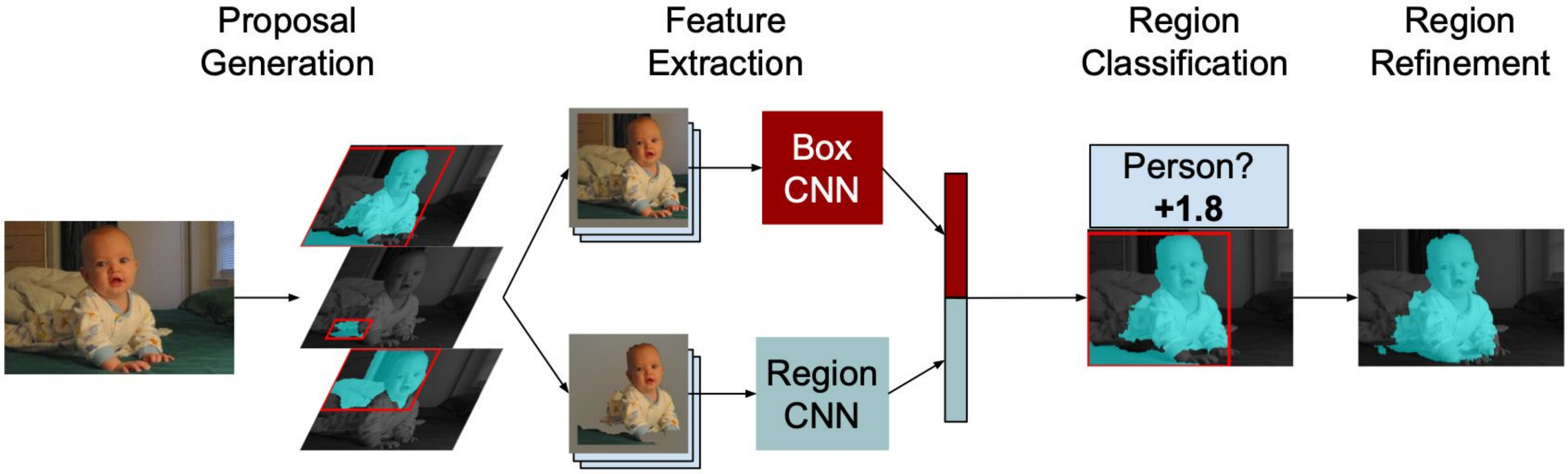


Proposal-based methods

- B. Hariharan et al. "Simultaneous Detection and Segmentation". ECCV 2014
  - Follow-up work: B. Hariharan et al. "Hypercolumns for Object Segmentation and Fine-grained Localization ". CVPR 2015
- Dai et al. „Instance-aware Semantic Segmentation via Multi-task Network Cascades". CVPR 2016
  - Previous work: Dai et al. "Convolutional Feature Masking for Joint Object and Stuff Segmentation". CVPR 2015

SDS

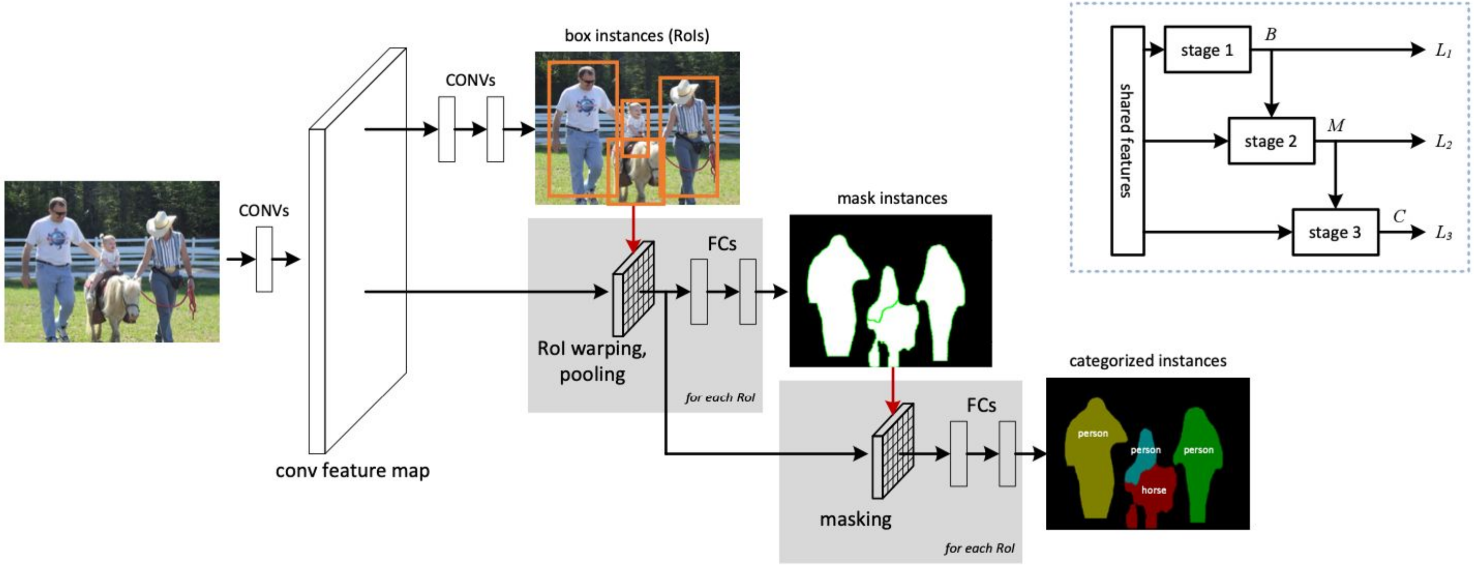
- SDS: Simultaneous Detection and Segmentation



B. Hariharan et al. "Simultaneous Detection and Segmentation". ECCV 2014

MNC

- MNC: Multi-task network cascades



Dai et al. „Instance-aware Semantic Segmentation via Multi-task Network Cascades". CVPR 2016

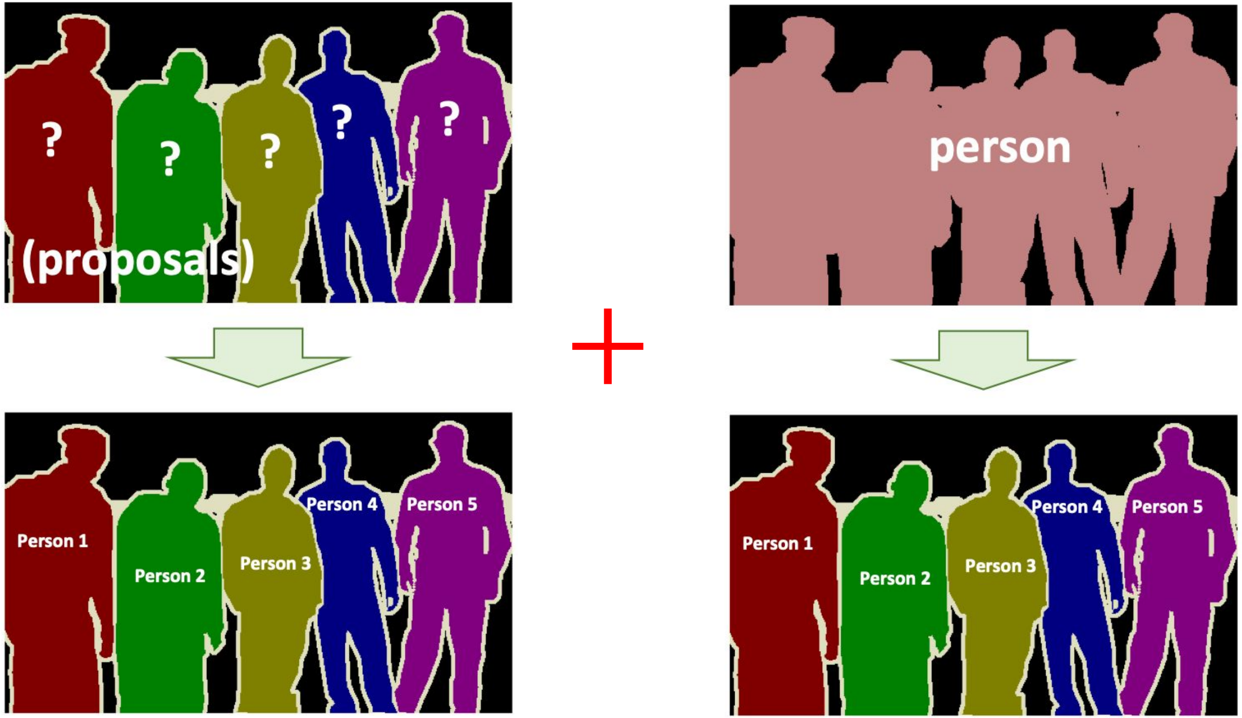
IS: the best of both worlds

Proposal-based

FCN-based

1. Proposals

1. Semantic segmentation



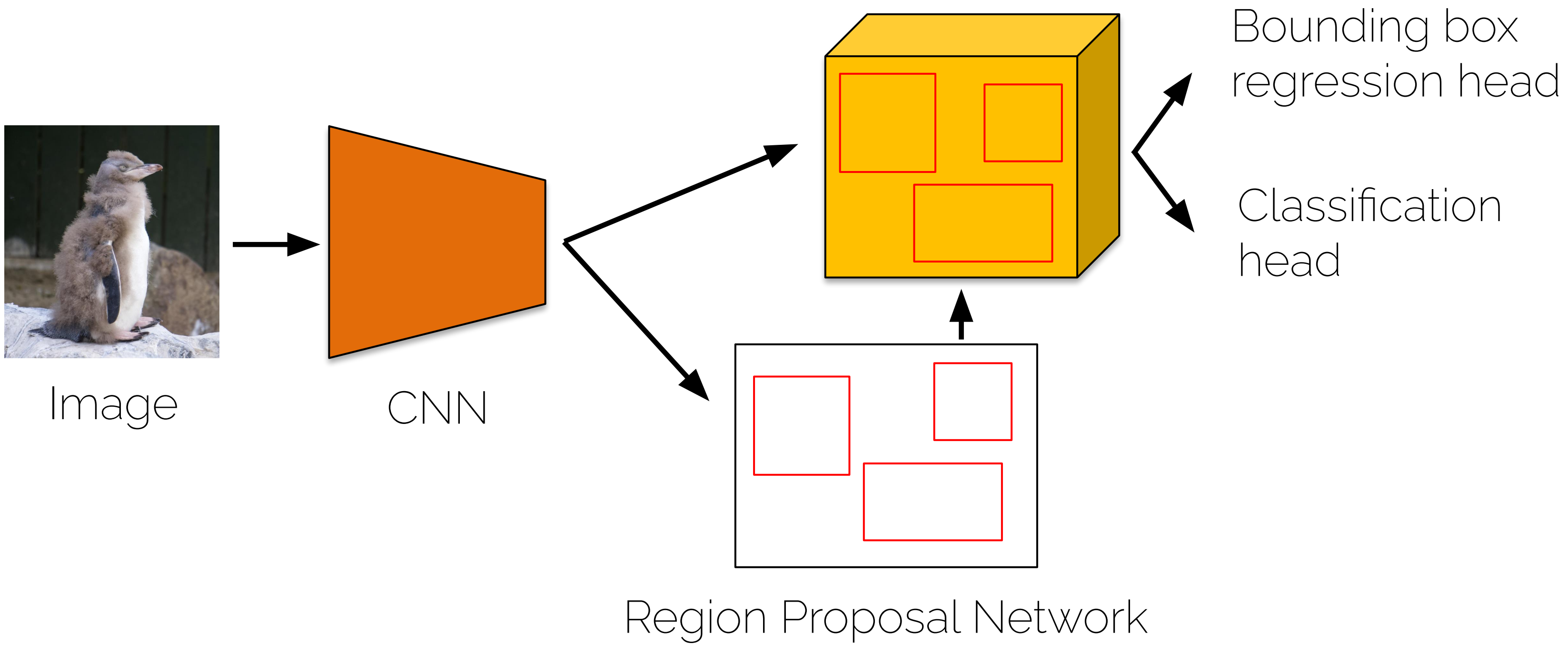
2. Assign a class

2. Find instances

Mask R-CNN

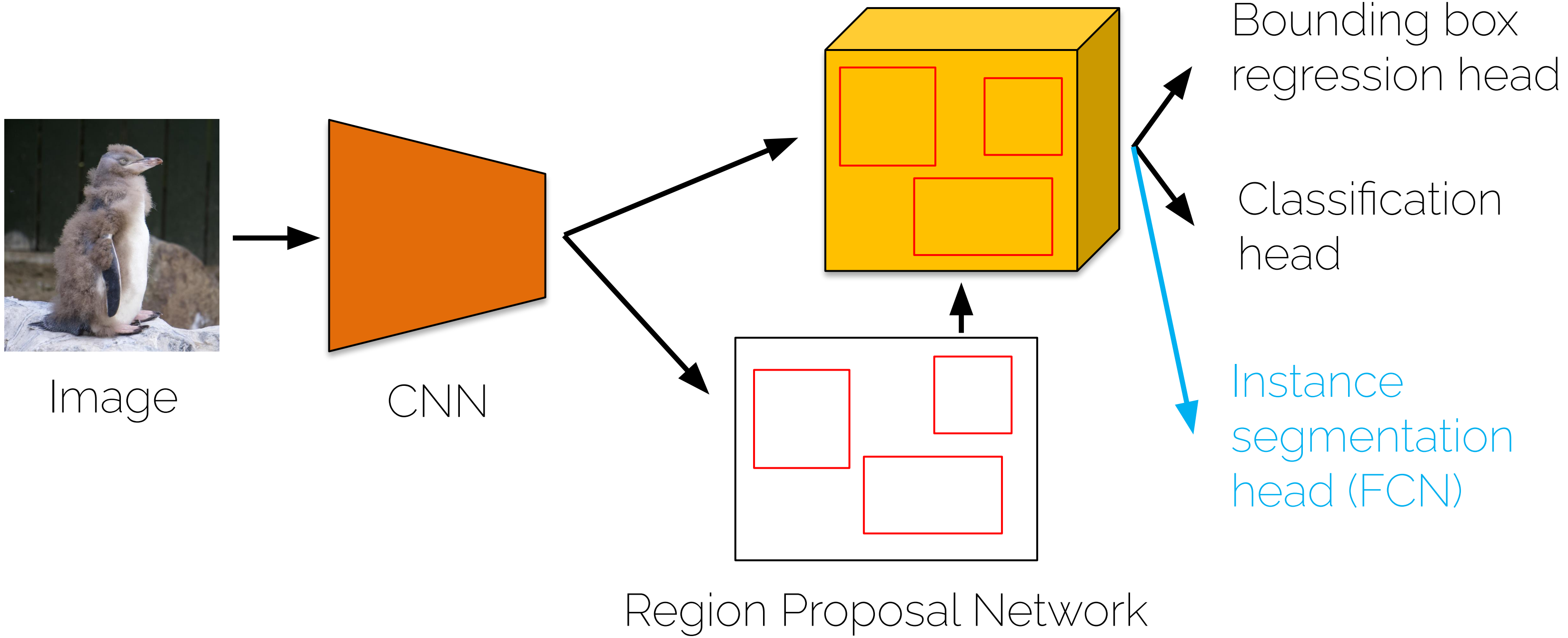
What is Mask-RCNN?

- Starting from the Faster R-CNN architecture



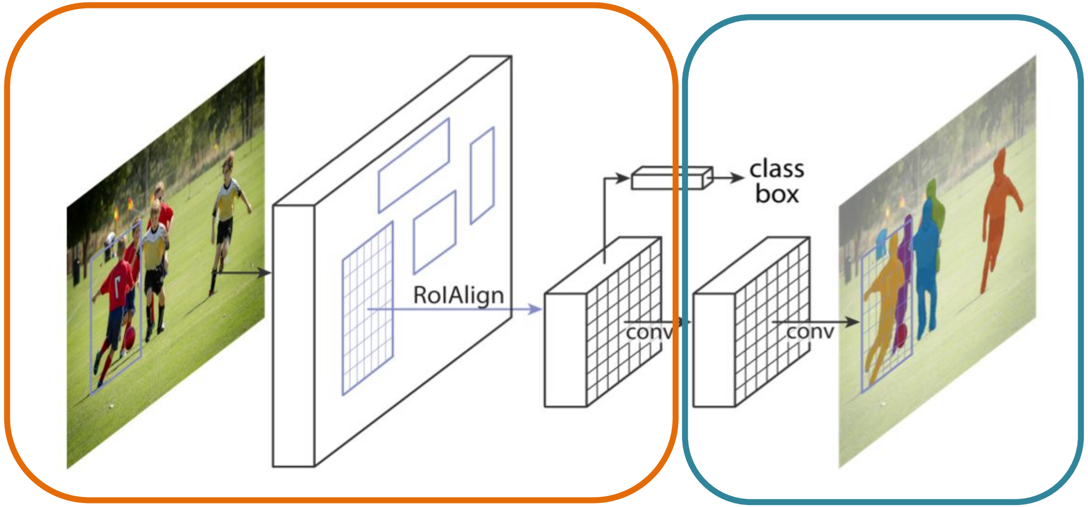
What is Mask-RCNN?

- Faster R-CNN + FCN for segmentation



What is Mask-RCNN?

- Faster R-CNN + FCN for segmentation



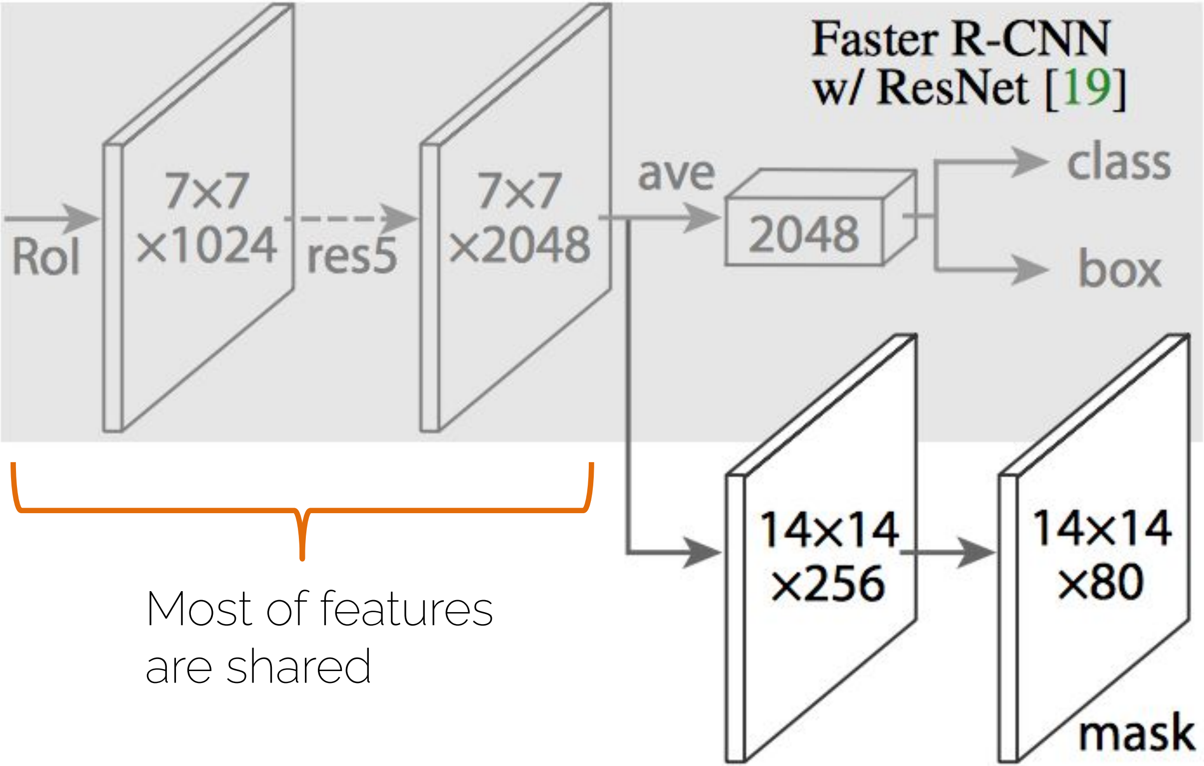
Mask loss = binary cross entropy per pixel for the k semantic classes

Faster R-CNN

FCN-like

He at al. “Mask R-CNN” ICCV 2017

Mask R-CNN

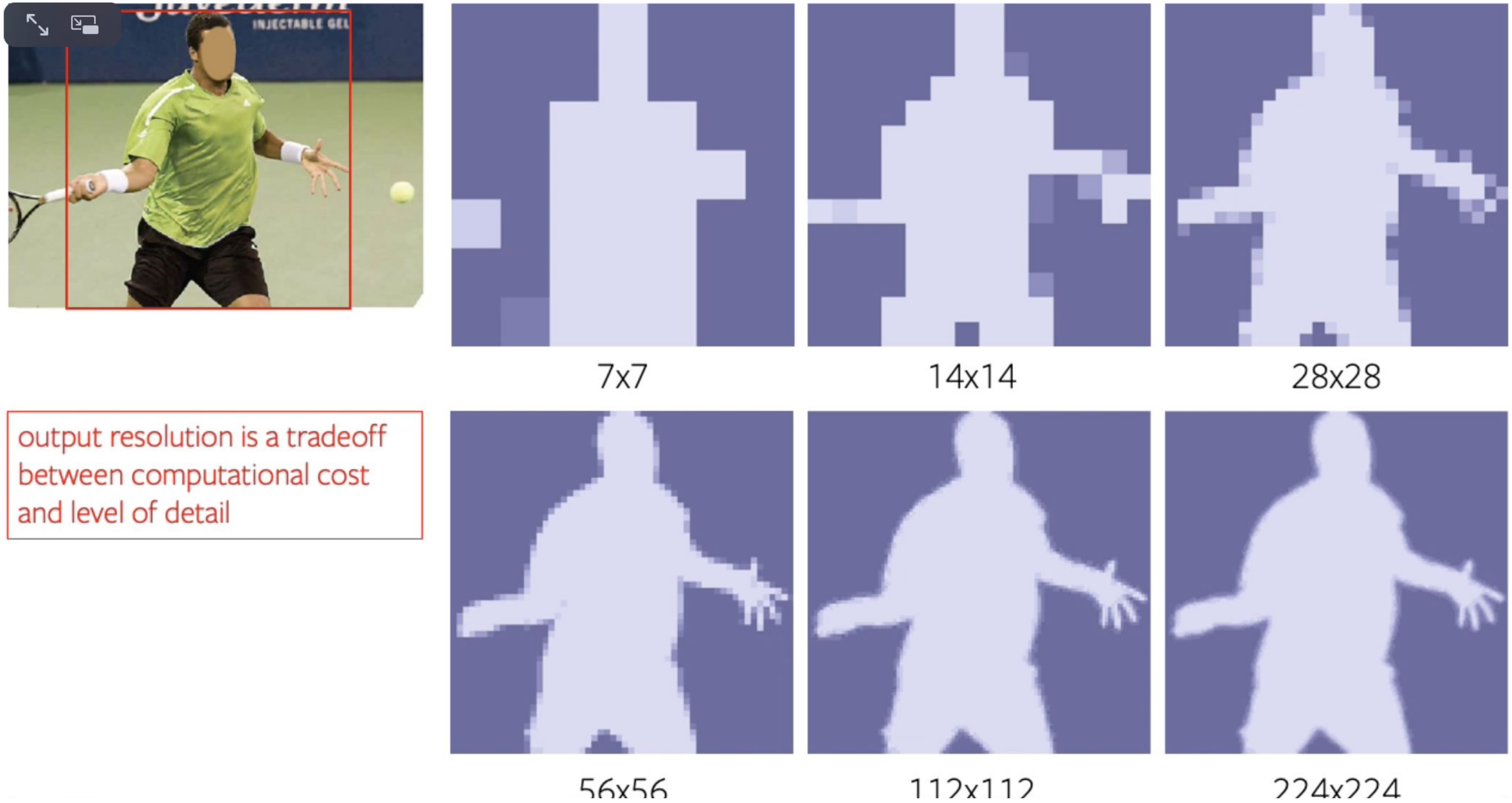


Object recognition head

Segmentation head

Most of features are shared

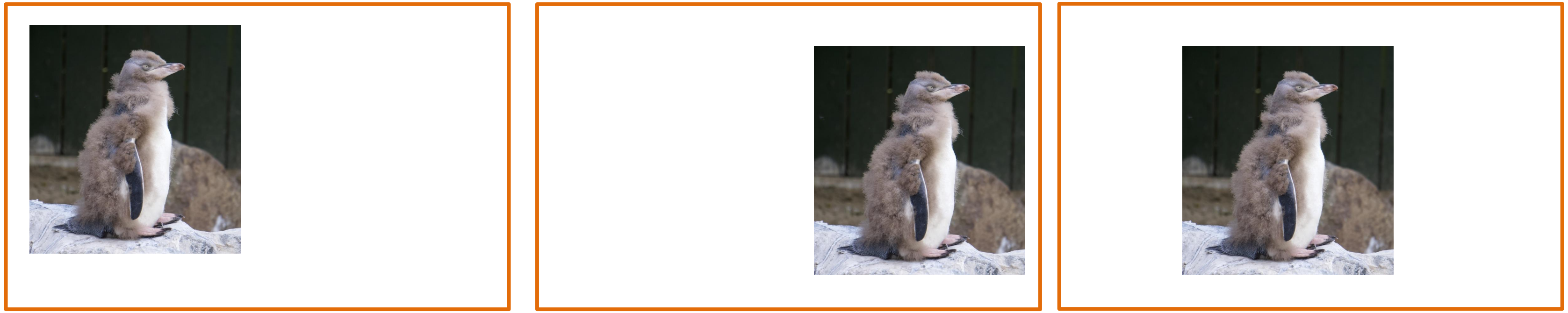
Resolution vs. computation



Slide by: A. Kirillov

Detection vs. segmentation

- Detection: for object classification, you require invariant representations



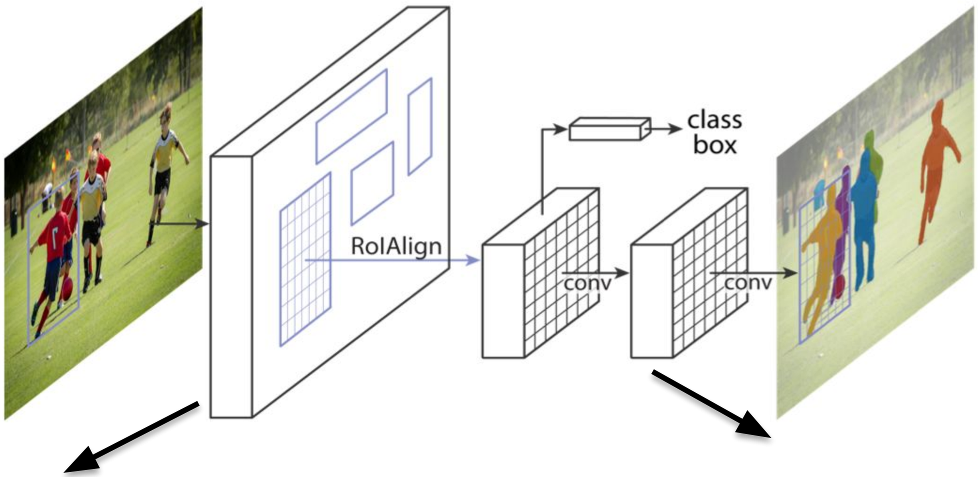
Translation invariance: wherever the penguin is in the image, I still want to have "penguin" as my classification output

Detection vs. segmentation

- Detection: for object classification, you require invariant representations
- Segmentation: you require equivariant representations
  - Translated object Translated mask
  - Scaled object scaled mask
  - For semantic segmentation, small objects are less important (less pixels), but for instance segmentation, all objects (no matter the size) are equally important

Mask-RCNN: operations

- What operations are equivariant?



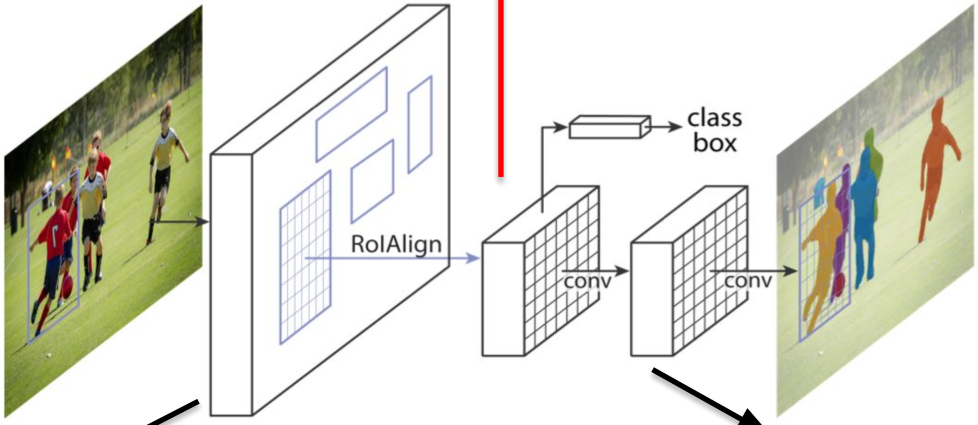
Features extraction = convolutional layers equivariant

Segmentation head is a fully convolutional network equivariant

Mask-RCNN: operations

- What operations are equivariant?

Fully connected layers and global pooling layers give invariance!

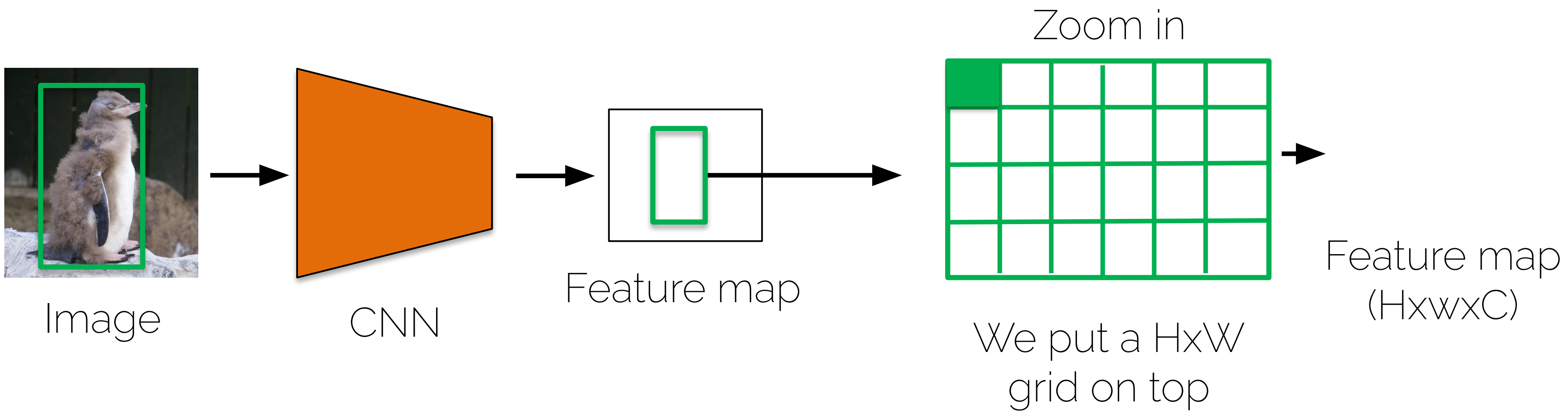


Features extraction = convolutional layers equivariant

Segmentation head is a fully convolutional network equivariant

Recall: RoI pooling

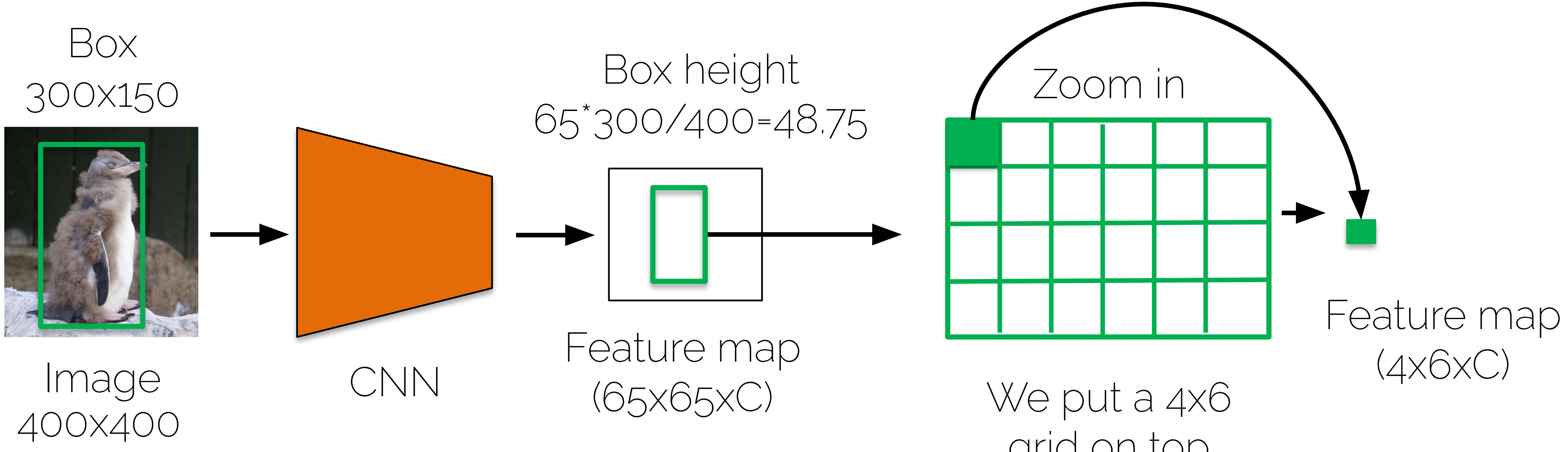
- Region of Interest Pooling: for every proposal



Recall: RoI pooling

- Let us look at sizes

Not suitable to extract pixel-wise precise masks



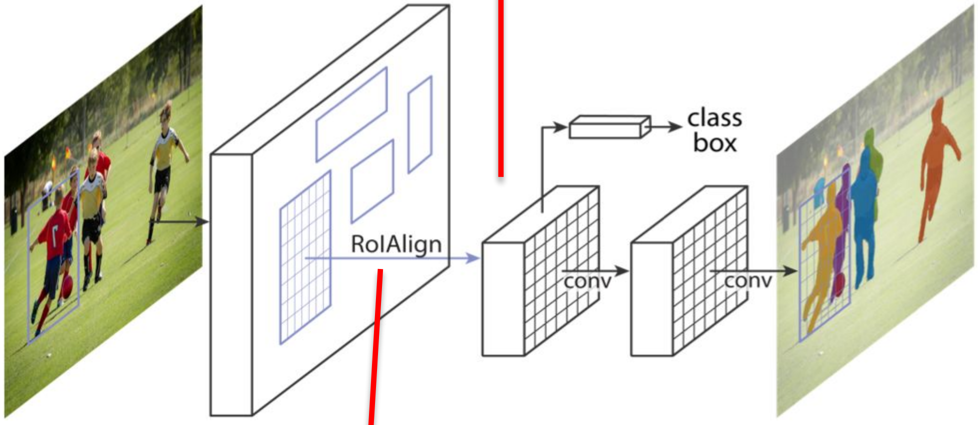
Quantization effect = chose 48

Quantization effect

Mask-RCNN: operations

- Make all operations equivariant

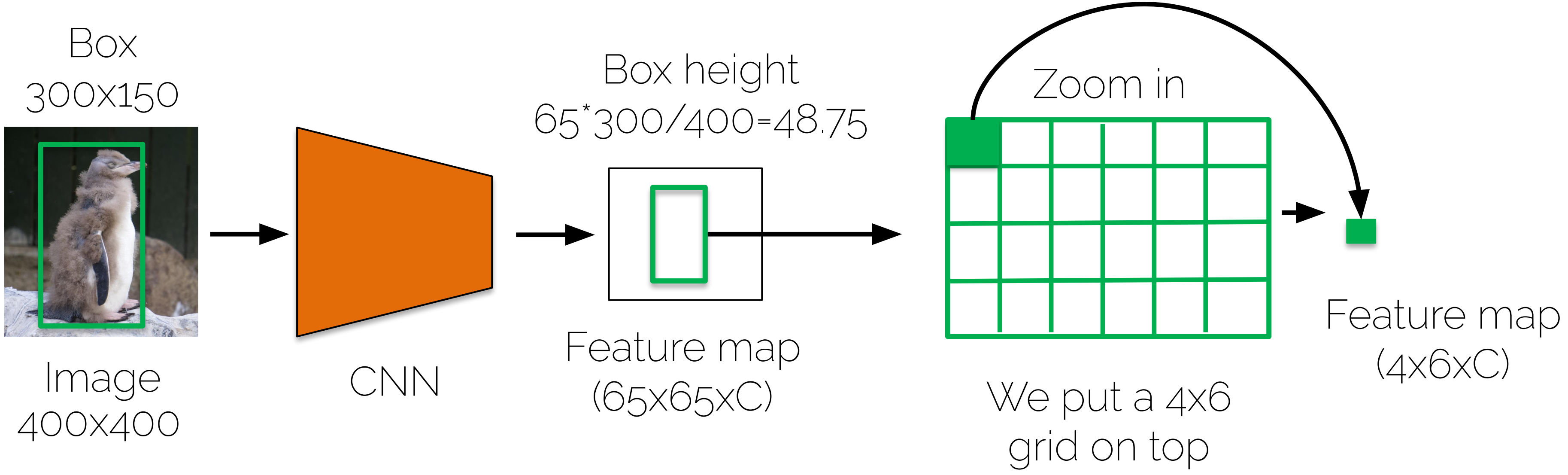
Fully connected layers and global pooling layers give invariance!



Exchange RoI pooling by an equivariant operation = RoI Align

RoIAlign

- Erase quantization effects



Chose 48.75

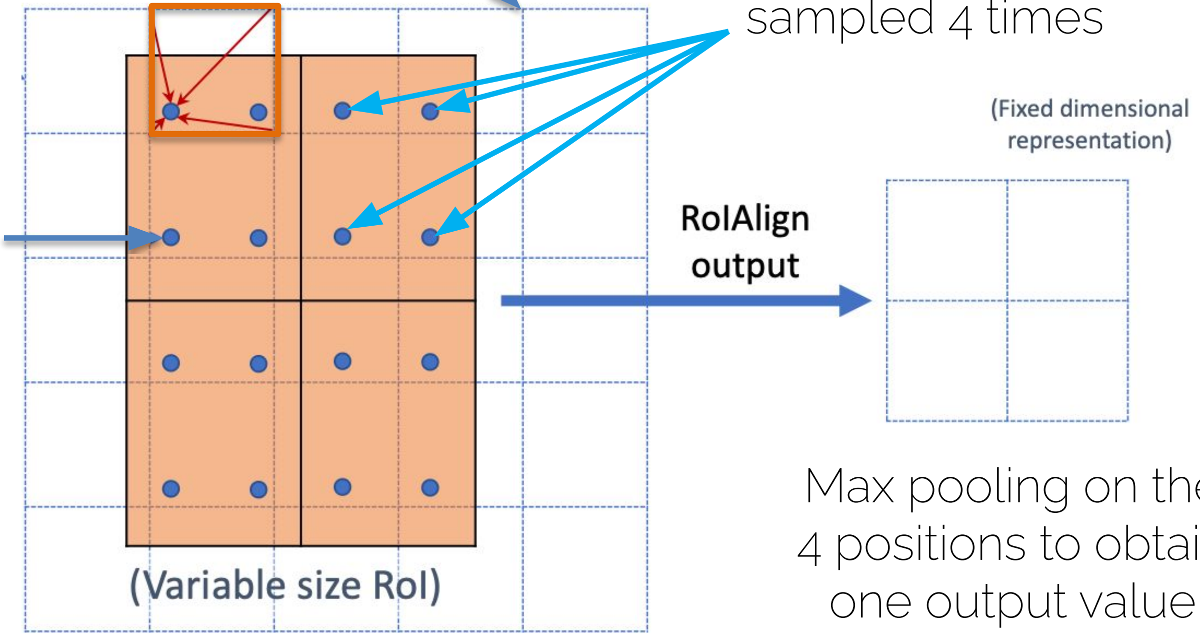
ROIAlign

To obtain the value use bilinear interpolation

Feature map

Each unit is sampled 4 times

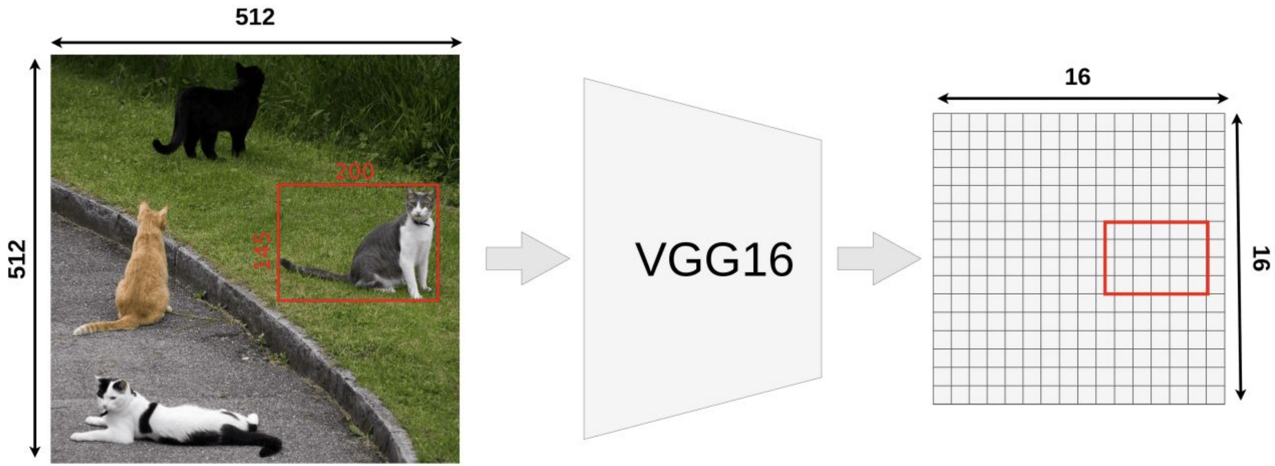
Grid points for bilinear interpolation



Max pooling on the 4 positions to obtain one output value

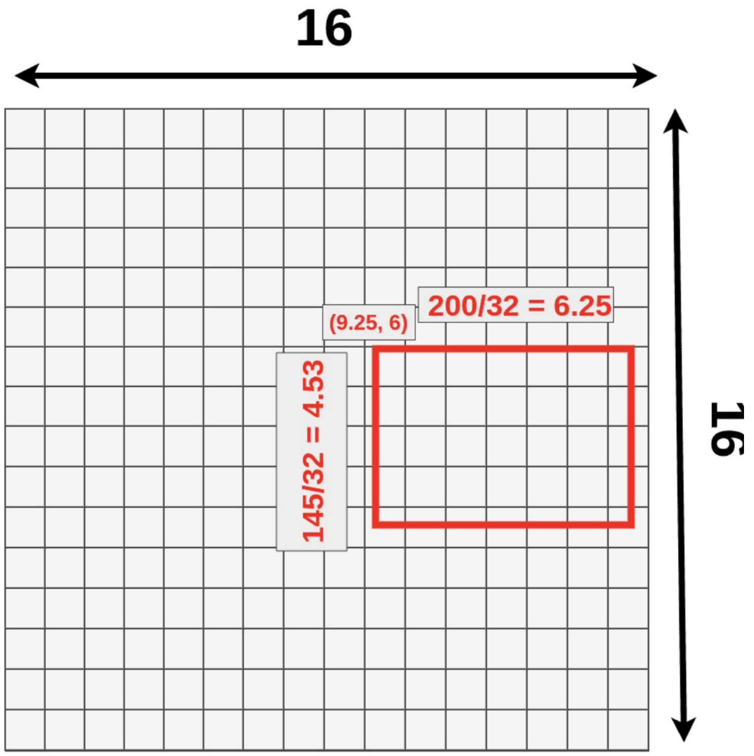
Image: Kaiming He

RoI Pooling: Recall



Model feature mapping process

RoI Pooling: Recall

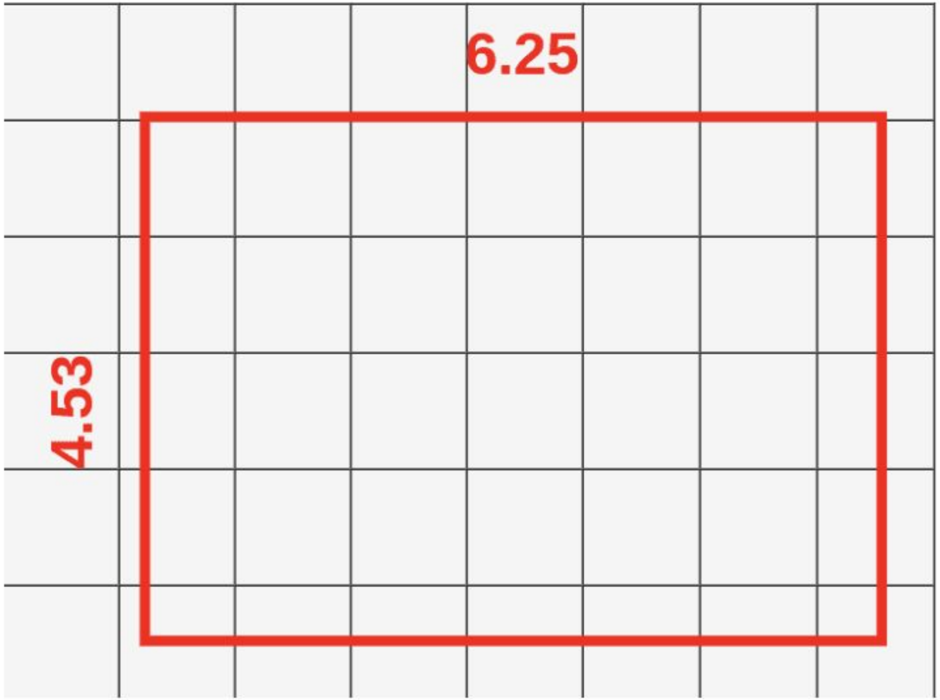


RoI placement

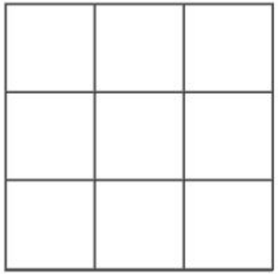
Next, we're using one of the proposed RoIs (145x200 box) and try to map it onto the feature map. Because not all of our object dimensions can be divided by 32, we're placing RoI not align with our grid.

- (9.25,6) - top left corner
- 6.25 - width
- 4.53 - height

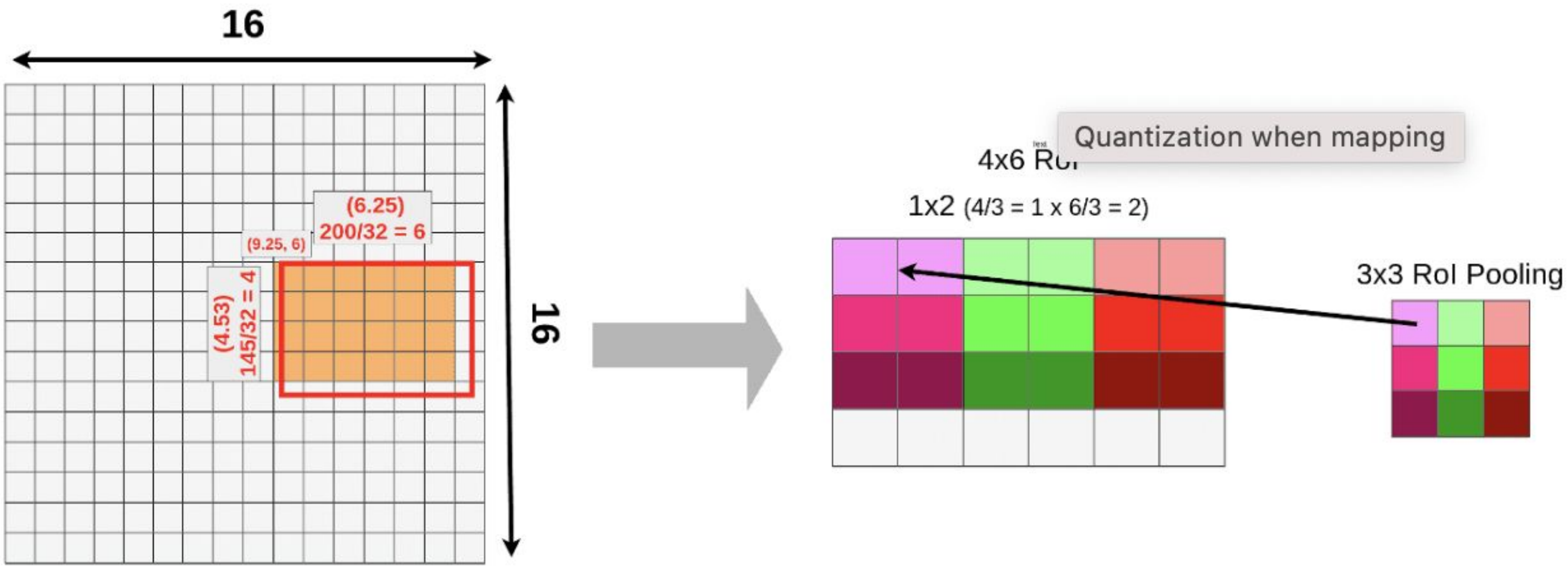
RoI Pooling: Recall



Pooling layer

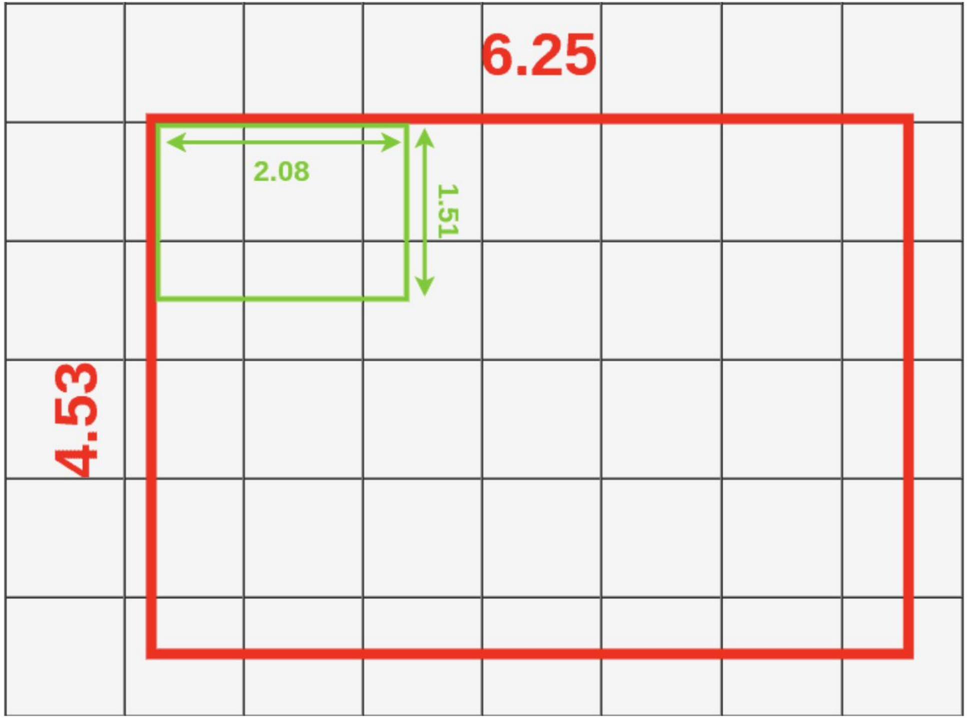


RoI Pooling: Recall



Quantization when mapping and pooling

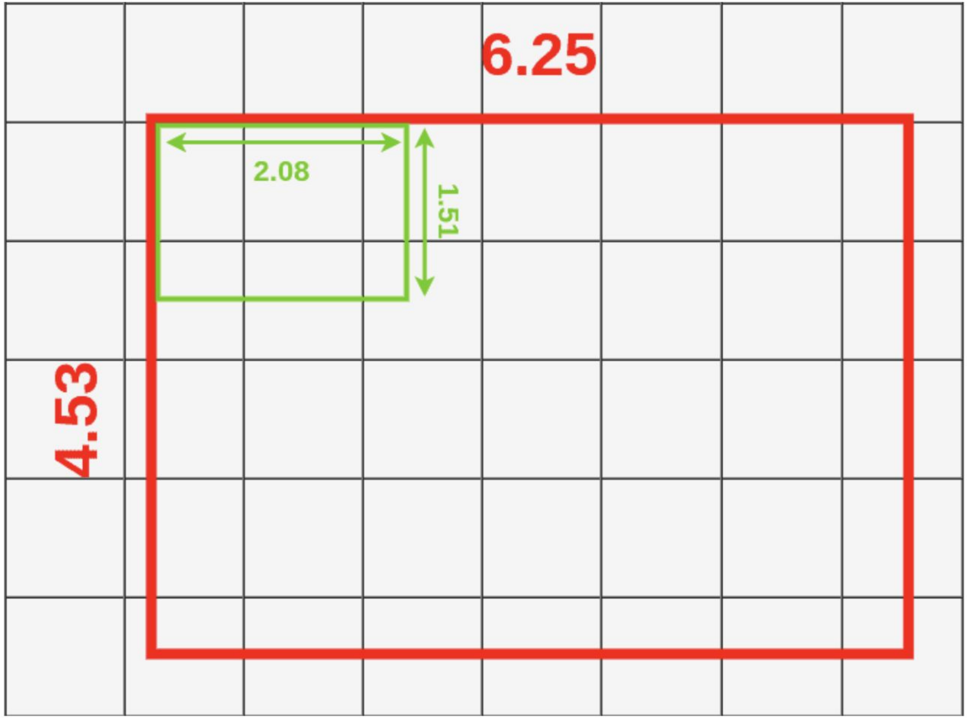
RoI Align



RoI Box size

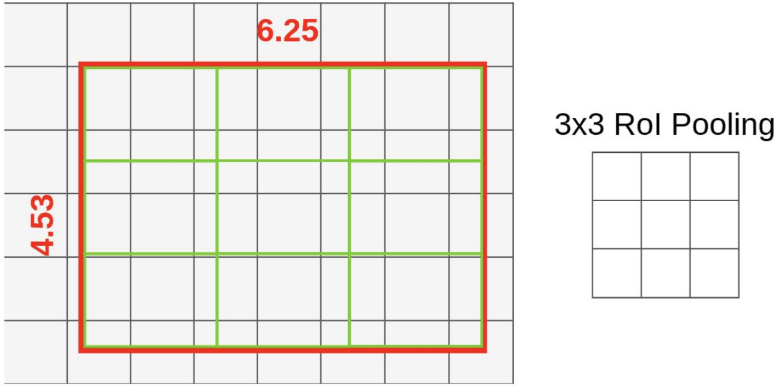
Divide it into 9 boxes (because in our case the dimensions of our RoI Align are 3 x 3).

RoI Align



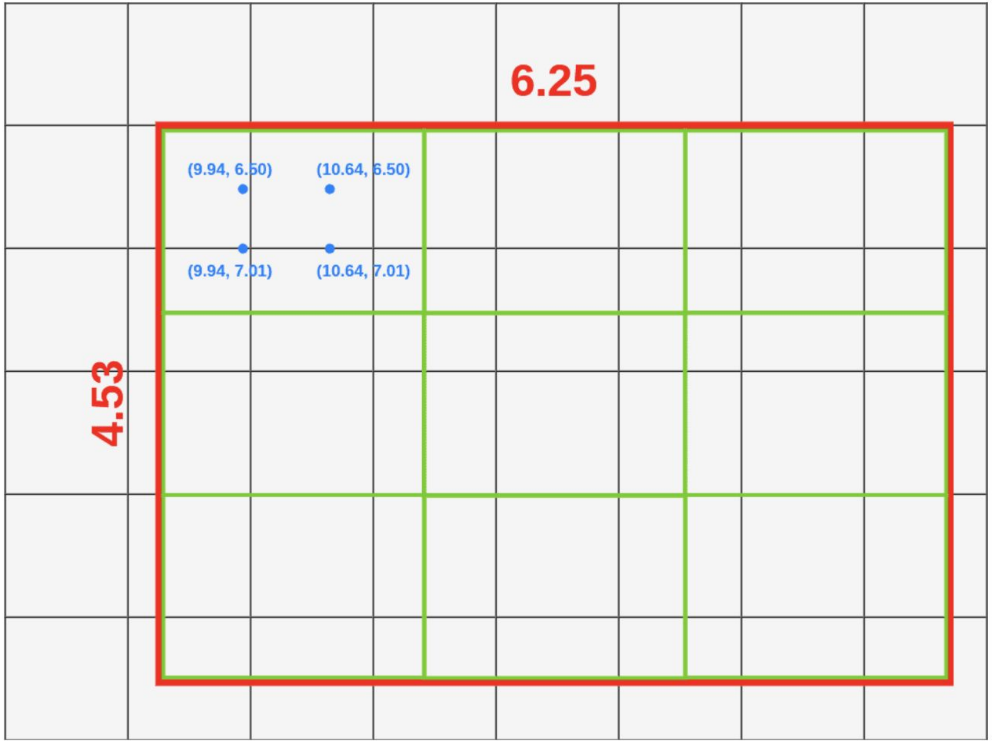
RoI Box size

Divide it into 9 boxes (because in our case the dimensions of our RoI Align are 3 x 3). That gives us a box with a height of 1.51 and a width of 2.08



RoI divided into boxes

RoI Align



Sampling points distribution

Get 4 sampling points. We get them by dividing the height and the width of the box by 3. For the first point:

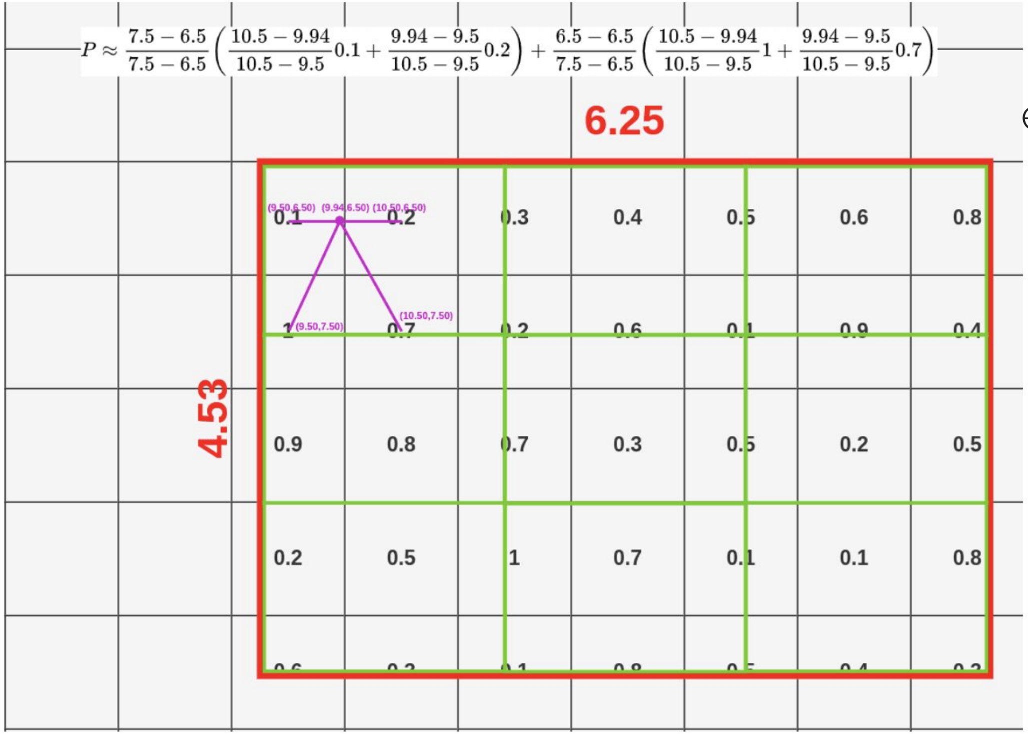
- $X = X\_box + (width/3) * 1 = 9.94$
- $Y = Y\_box + (height/3) * 1 = 6.50$

where (X_box, Y_box) = (9.25, 6).

For the second point:

- $X = X\_box + (width/3) * 1 = 9.94$
- $Y = Y\_box + (height/3) * 2 = 7.01$

RoI Align



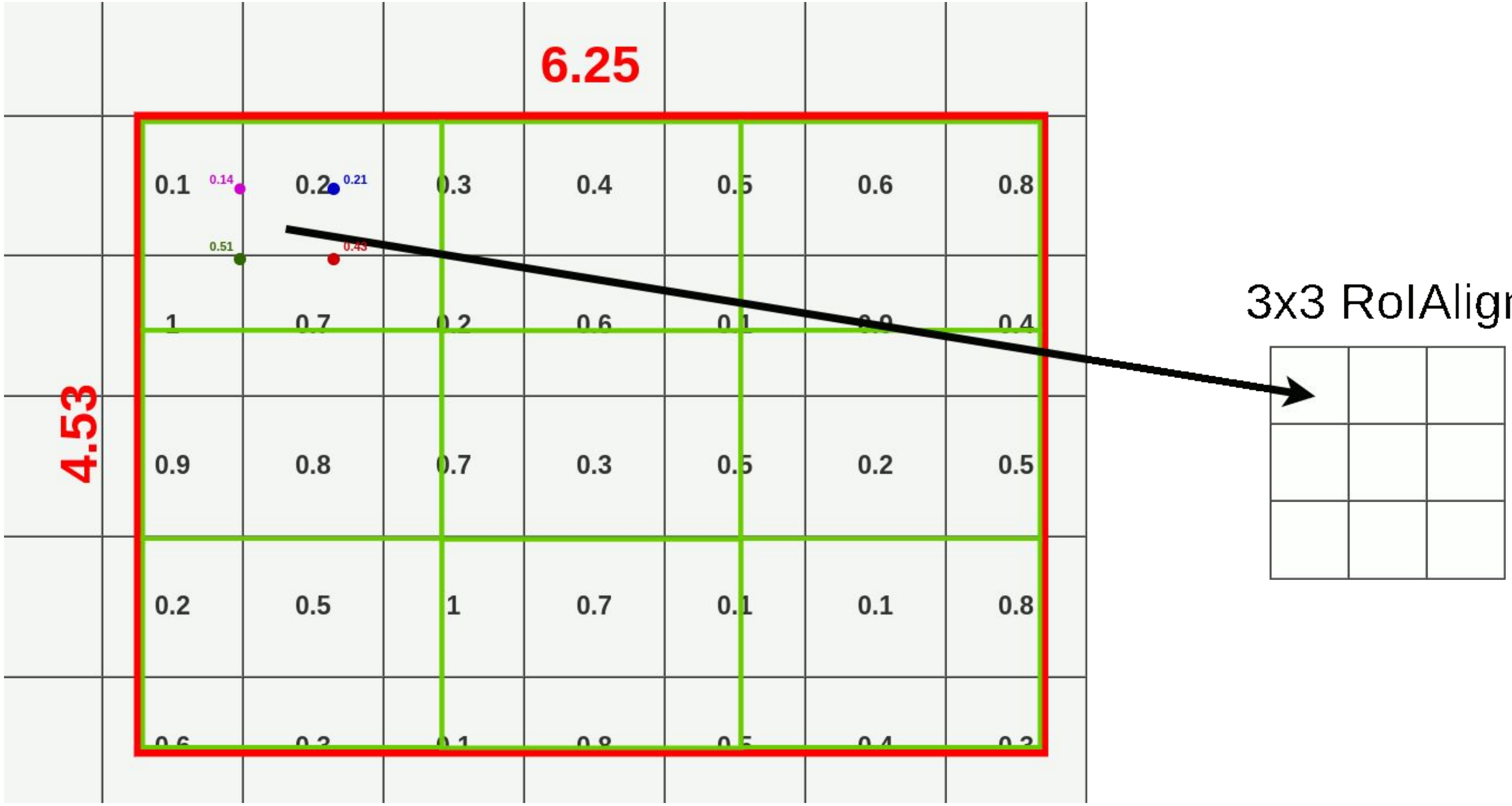
Bilinear Interpolation for the first point

Do bilinear interpolation following the equation:

$$P \approx \frac{y_2-y}{y_2-y_1}\left(\frac{x_2-x}{x_2-x_1}Q_{11}+\frac{x-x_1}{x_2-x_1}Q_{21}\right)+\frac{y-y_1}{y_2-y_1}\left(\frac{x_2-x}{x_2-x_1}Q_{12}+\frac{x-x_1}{x_2-x_1}Q_{22}\right)$$

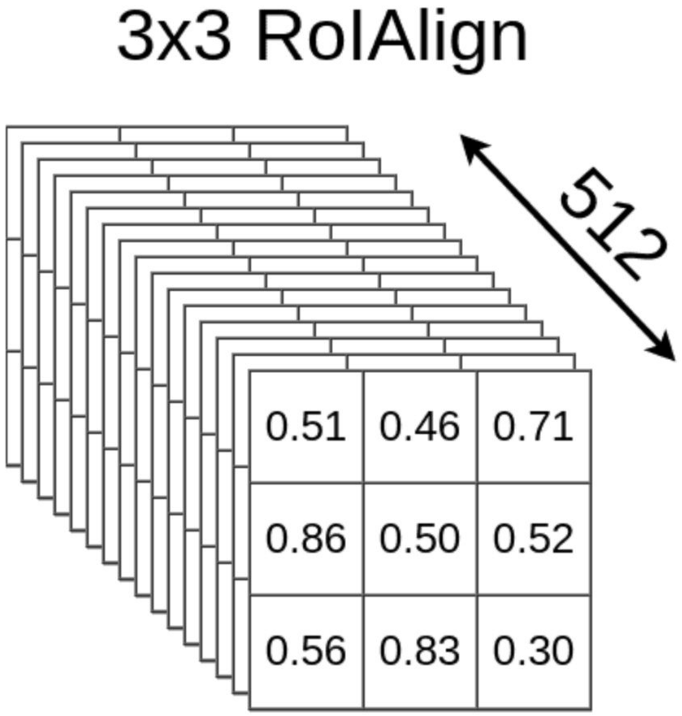
Bilinear Interpolation equation

RoI Align



6.25
4.53
0.1
0.14
0.2
0.21
0.3
0.4
0.5
0.6
0.8
0.51
0.43
1
0.7
0.2
0.6
0.1
0.4
0.9
0.8
0.7
0.3
0.5
0.2
0.5
0.2
0.5
1
0.7
0.1
0.1
0.8
3x3 RoIAlign

RoI Align

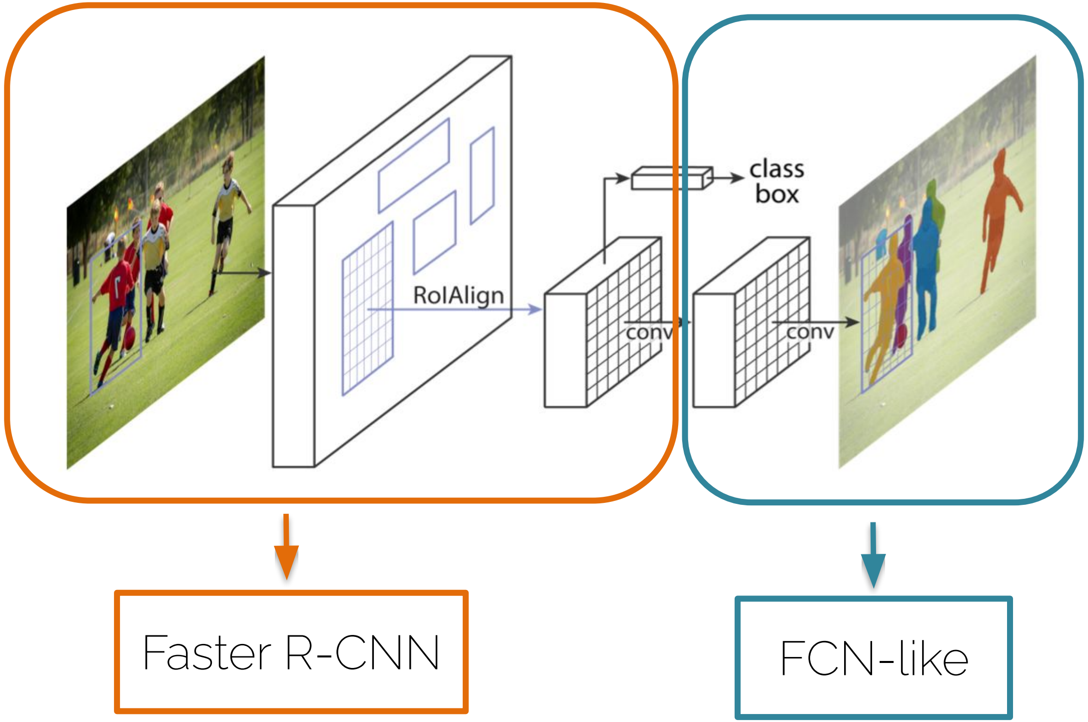


RoIAlign full size

Do this for every channel.

What is Mask-RCNN?

- Faster R-CNN + FCN for segmentation



Mask loss = binary cross entropy per pixel for the k semantic classes

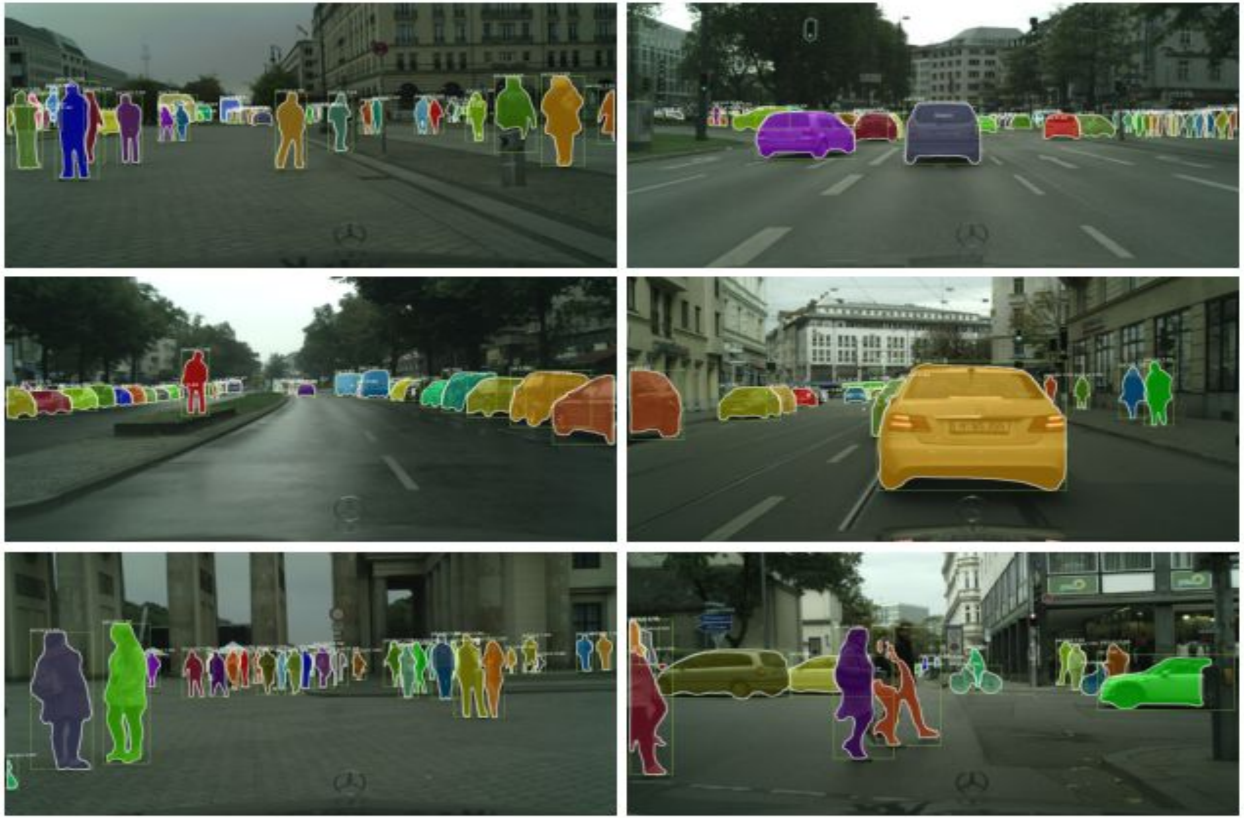
He at al. "Mask R-CNN" ICCV 2017

Mask R-CNN: qualitative results

Mask R-CNN: qualitative results

Mask R-CNN: qualitative results

Mask R-CNN: qualitative results



Mask R-CNN: extended for joints

Model a keypoint's location as a one-hot mask, and adopt Mask R-CNN to predict K masks, one for each of K keypoint types (e.g., left shoulder, right elbow). This demonstrates the flexibility of Mask R-CNN.

Improving Mask-RCNN

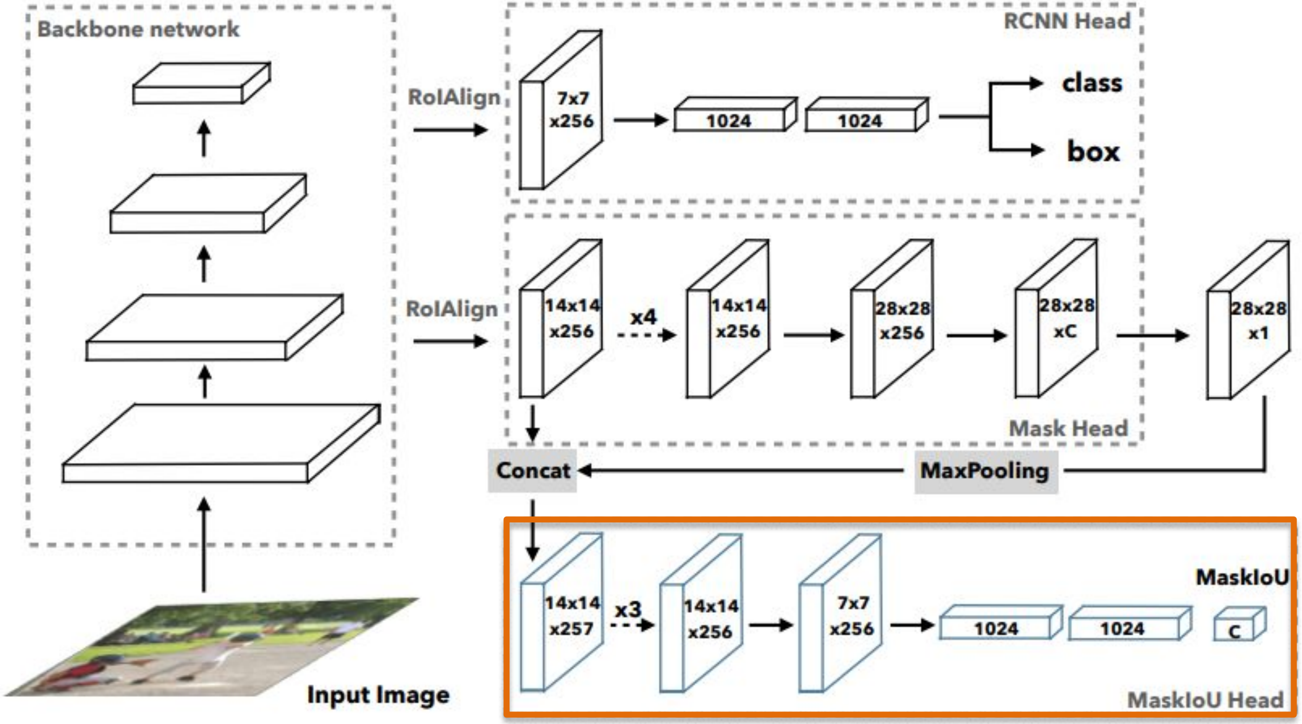
- One problem with Mask R-CNN is that the mask quality score is computed as the confidence score for the bounding box

The only way the "instance" is evaluated is through the box loss

Recall the mask loss just evaluates if the pixels have the correct semantic class, not the correct instance!

Both instances have the same class = person

Mask IoU head



Measure the intersection over union between the predicted mask and ground truth mask

Huang et al., "Mask Scoring R-CNN", CVPR 2019

Mask confidence score

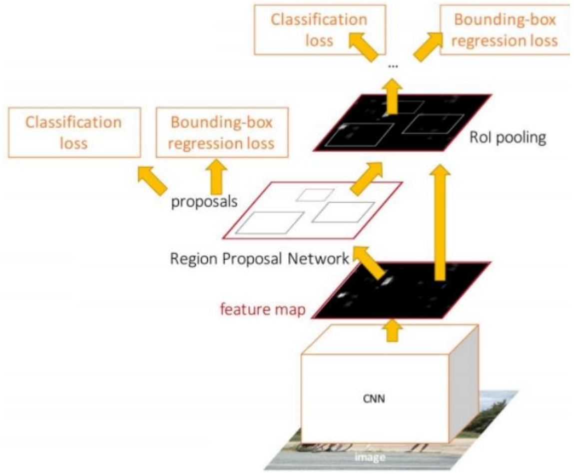
Typically, Mask scoring R-CNN gives lower confidence scores than Mask R-CNN, which corresponds to masks not being perfect (IoU < 1.0).

This tiny modification achieves SOTA results.

Is one-stage vs two-stage also applicable to masks?

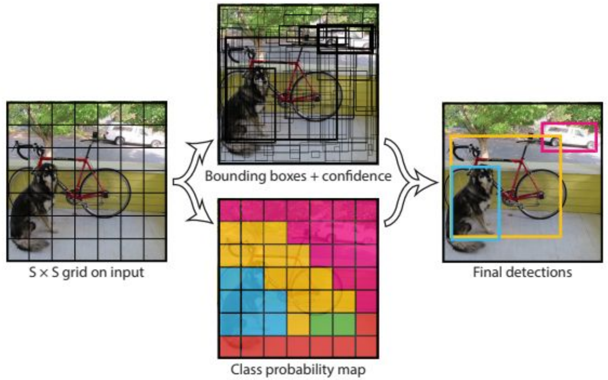
One-stage vs two-stage detectors

Faster R-CNN



Slower, but has higher performance

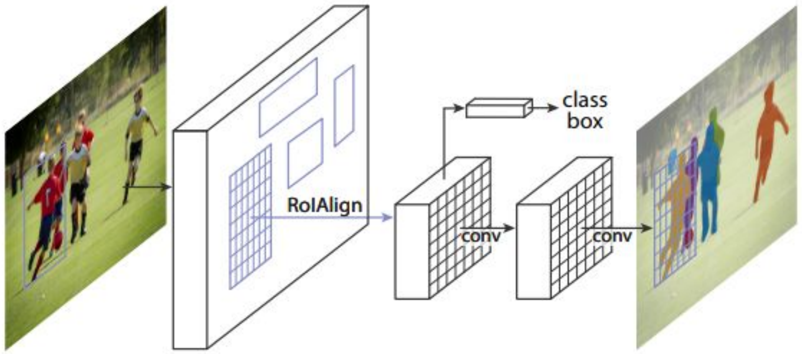
YOLO



Faster, but has lower performance

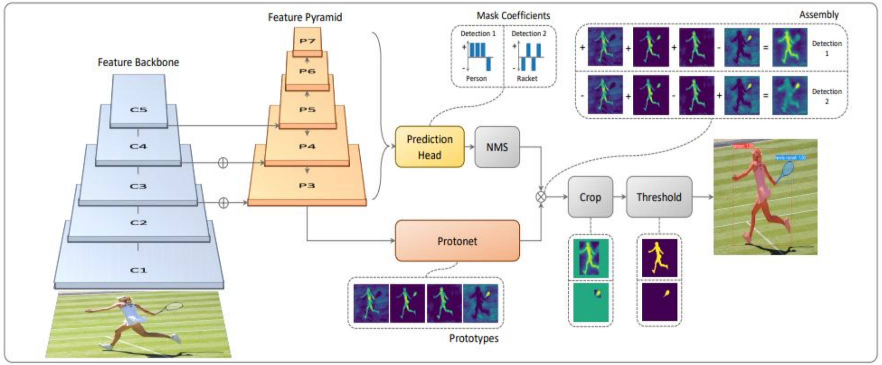
One-stage vs two-stage instance segmenters

Mask R-CNN



Slower, but has higher performance

YOLACT



Faster, but has lower performance

YOLO with masks?

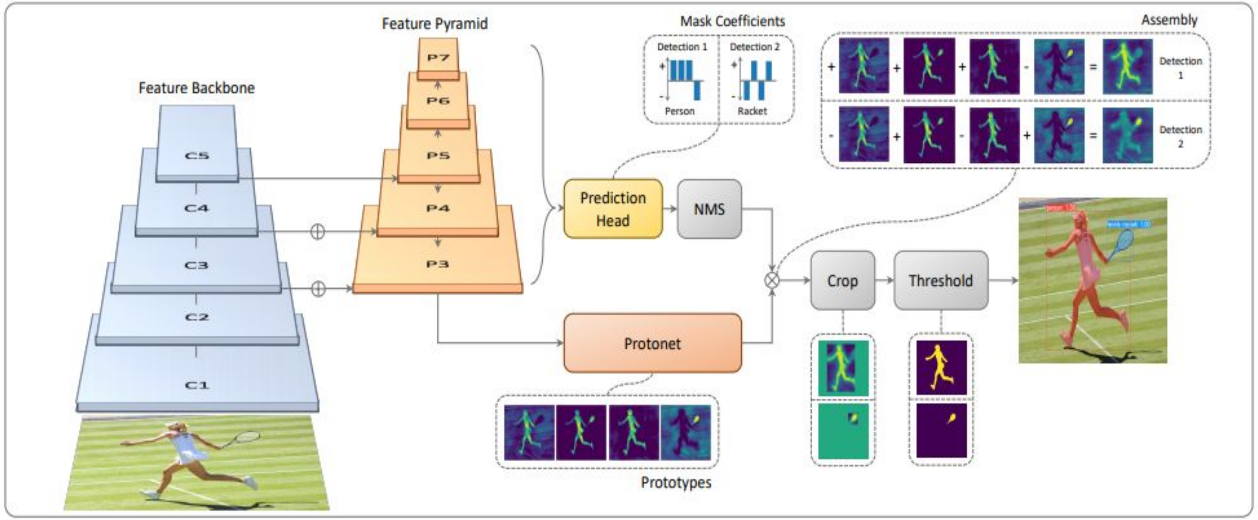
"Boxes are stupid anyway though, I'm probably a true believer in masks except I can't get YOLO to learn them."

– Joseph Redmon, YOLOv3

YolACT*

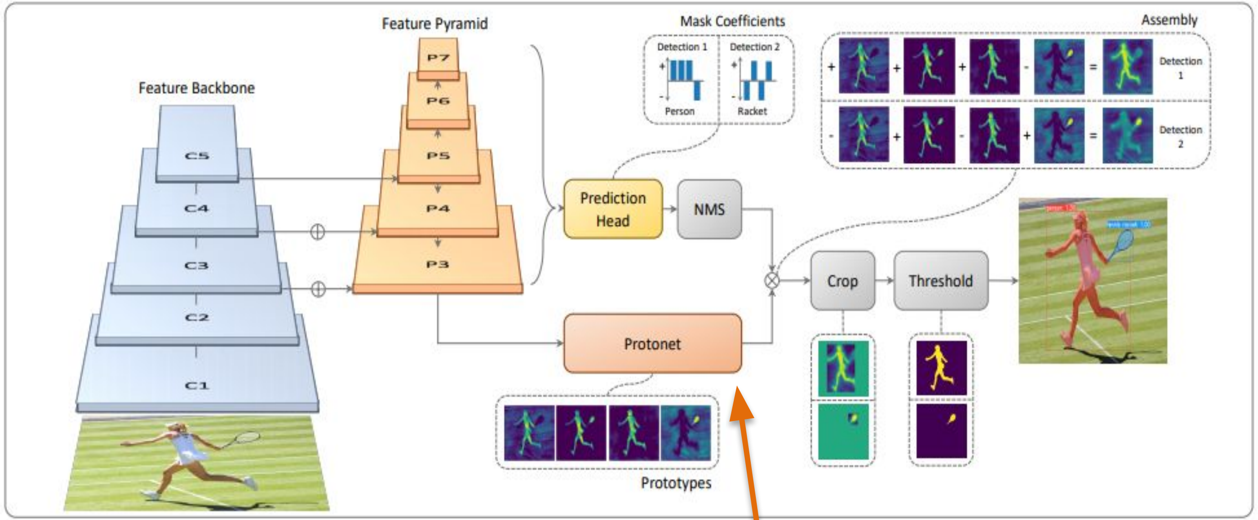
*You Only Look At CoefficienTs

YOLACT: idea



D. Bolya et al. "YOLACT: Real-time Instance Segmentation". ICCV 2019

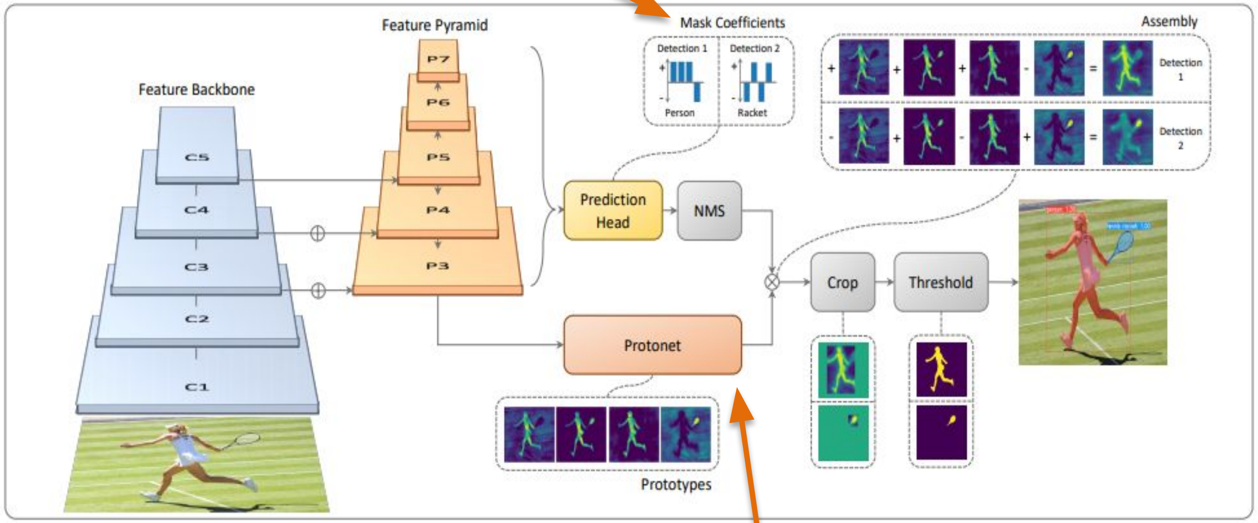
YOLACT: idea



1) Generate mask prototypes

YOLACT: idea

2) Generate mask coefficients

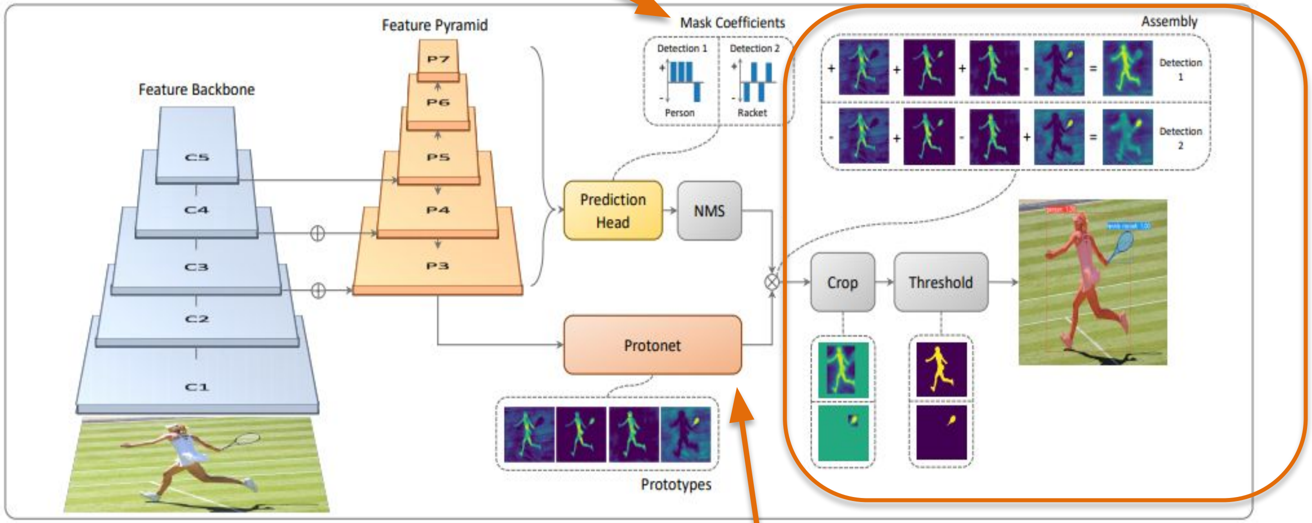


1) Generate mask prototypes

YOLACT: idea

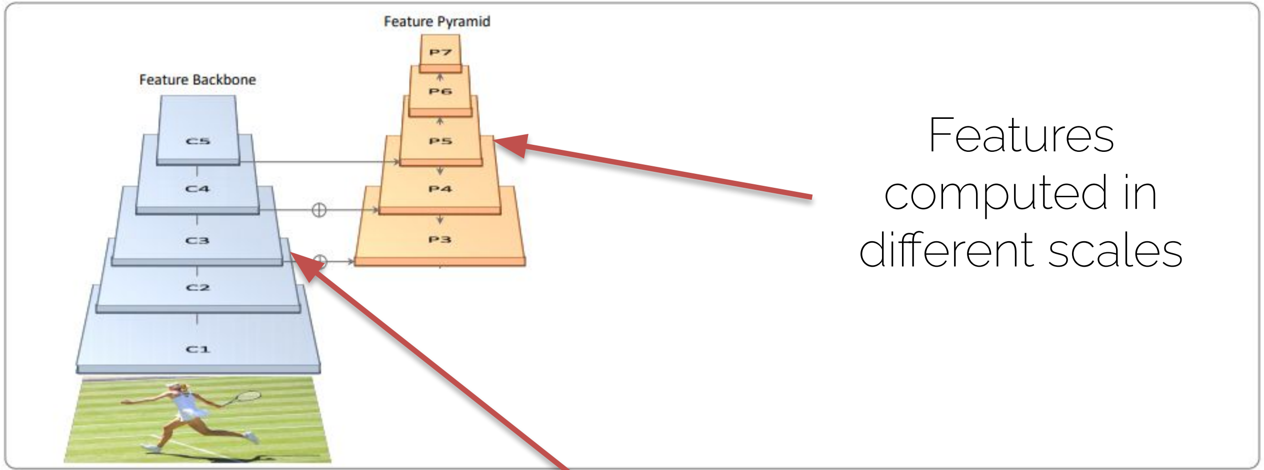
2) Generate mask coefficients

3) Combine (1) and (2)



1) Generate mask prototypes

YOLACT: backbone

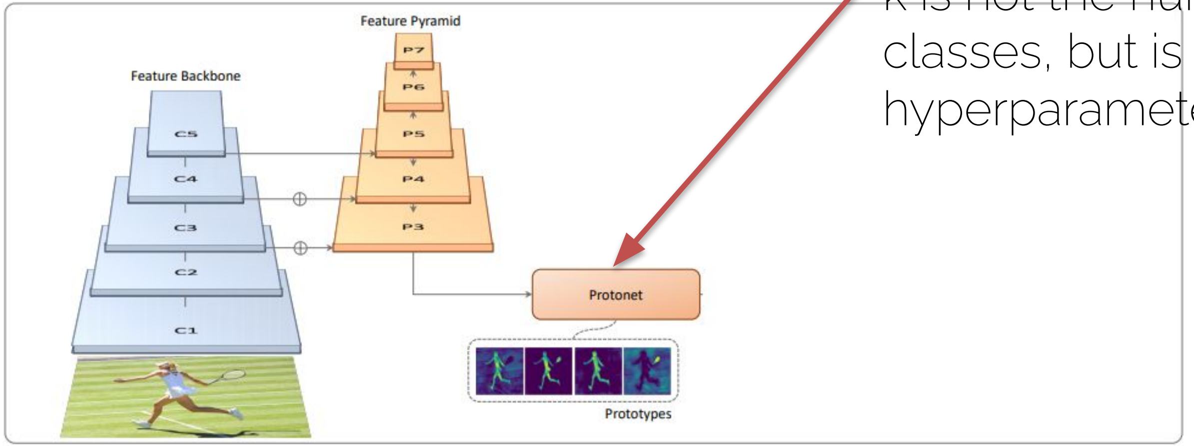


Features computed in different scales

ResNet-101

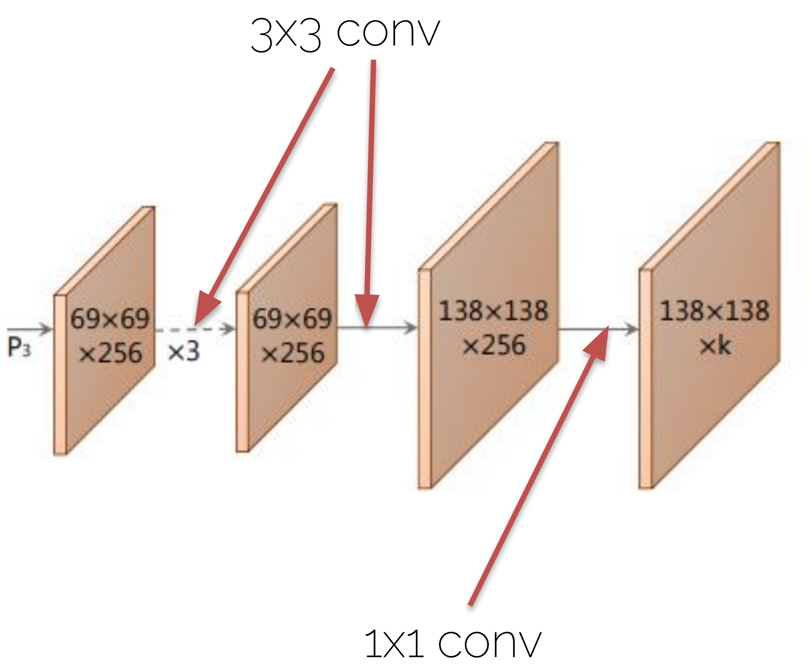
YOLACT: protonet

Generate k prototype masks.
k is not the number of classes, but is a hyperparameter.



YOLACT: protonet

- Fully convolutional network

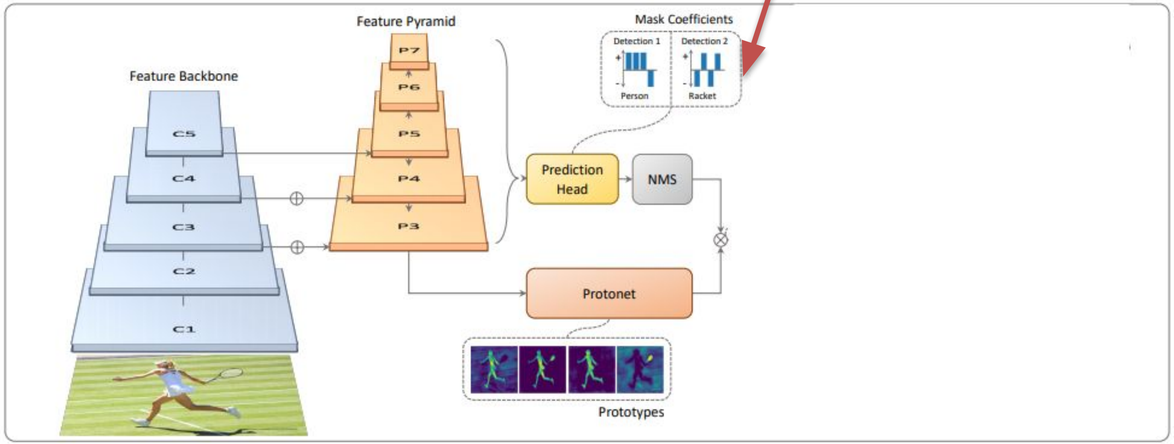


Similar to the mask branch in Mask R-CNN.

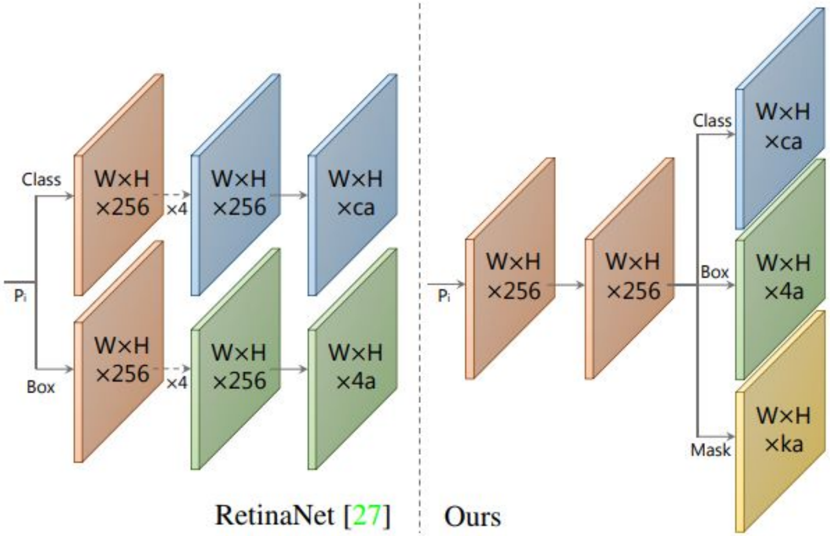
However, no loss function is applied on this stage.

YOLACT: mask coefficients

Predict a coefficient for every predicted mask.



YOLACT: mask coefficients



Predict one class per anchor box

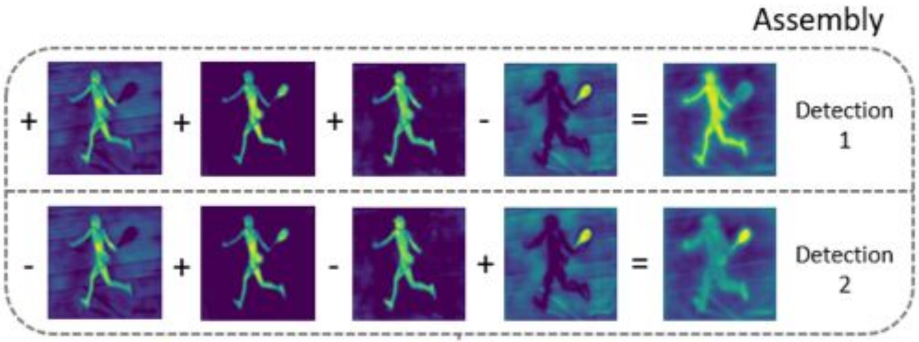
Predict the regression per anchor box

Predict k coefficients (one per prototype mask) per anchor

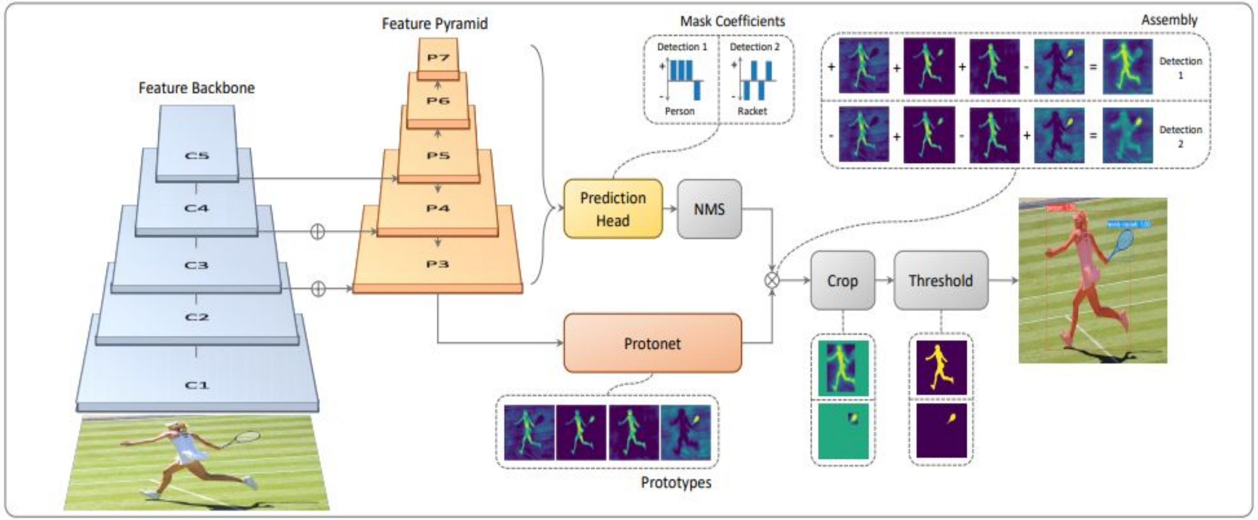
The network is similar but shallower than RetinaNet

YOLACT: mask assembly

1. Do a linear combination between the mask coefficients and the mask prototypes.
2. Predict the mask as M = $\sigma(PC^T)$ where P is a (HxWxK) matrix of prototype masks, C is a (NxK) matrix of mask coefficients surviving NMS, and σ is a nonlinearity.



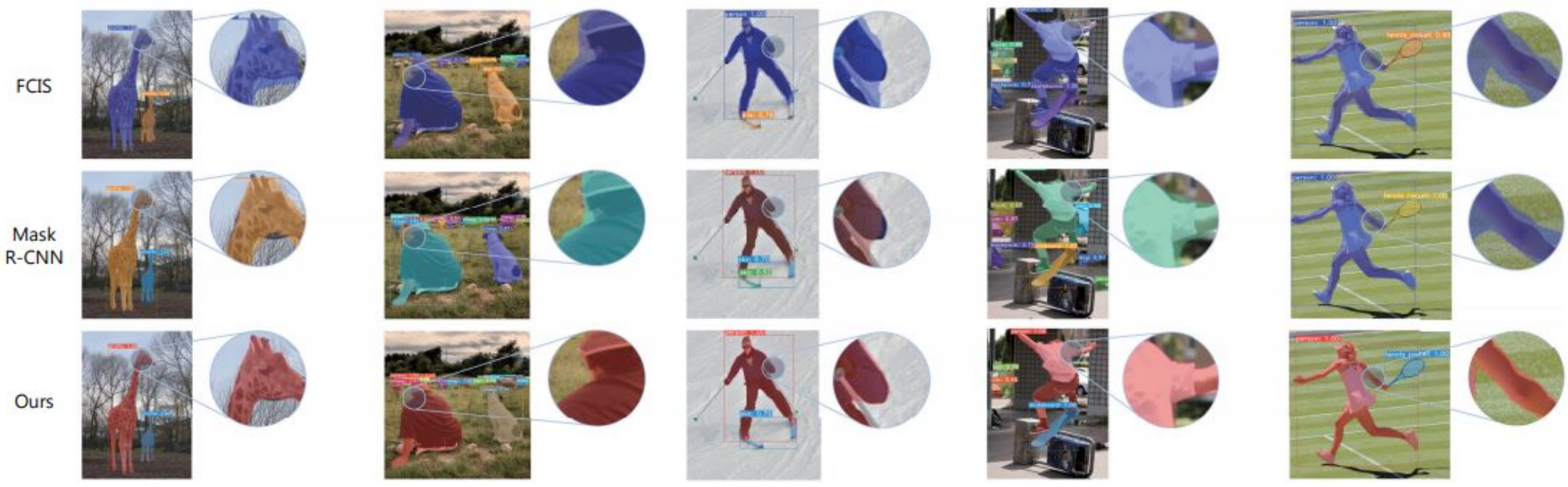
YOLACT: loss function



Cross-entropy between the assembled masks and the ground truth, in addition to the standard losses (regression for the bounding box, and classification for the class of the object/mask).

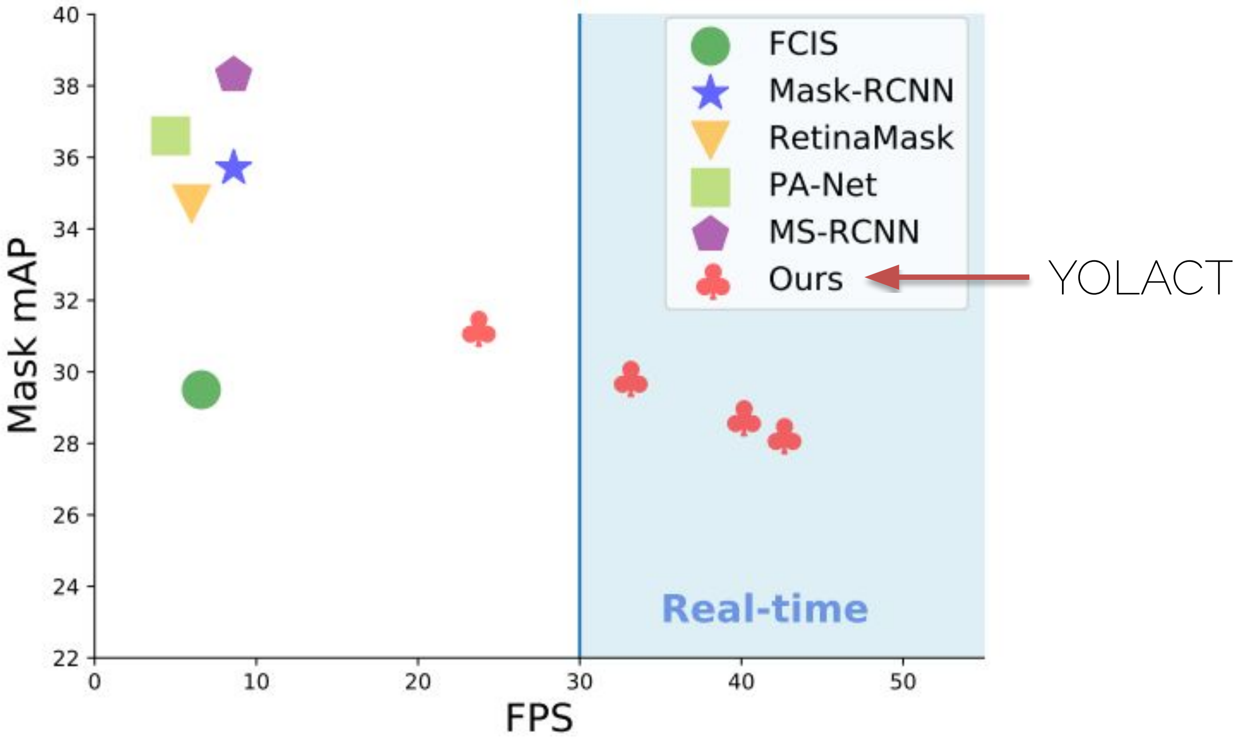
YOLACT: qualitative results

YOLACT: qualitative results



For large objects, the quality of the masks is even better than those of two-stage detectors

So, which segmenter to use?



YOLACT++: improvements

- A specially designed version of NMS, in order to make the procedure faster.

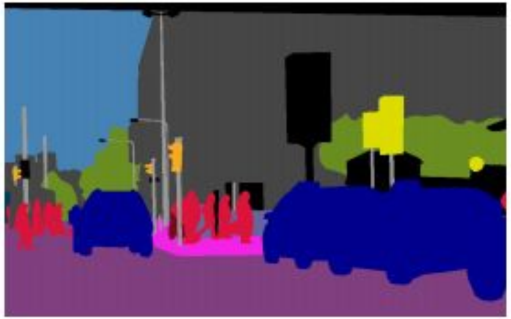
- An auxiliary semantic segmentation loss function performed on the final features of the FPN.
The module is not used during the inference stage.

- D. Boyla et al. "YOLACT++: Better real-time instance segmentation". arXiv:1912.06218 2019

Panoptic segmentation

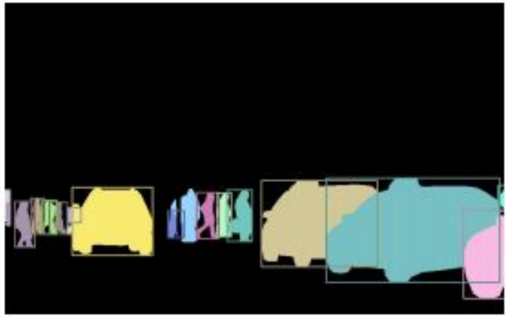
Panoptic segmentation

Semantic segmentation



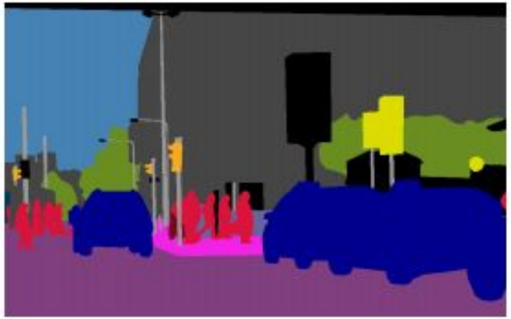
+

Instance segmentation



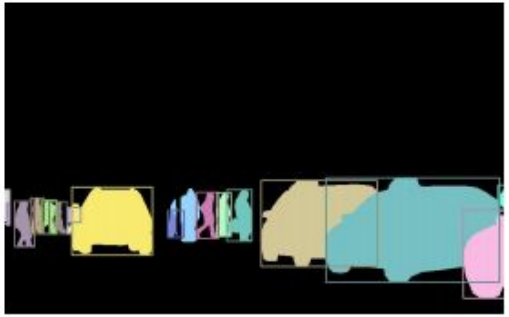
Panoptic segmentation

Semantic segmentation



\+

Instance segmentation

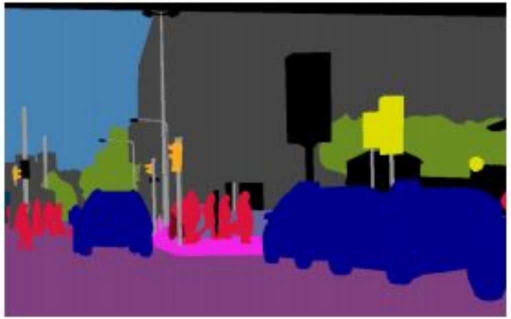


FCN-like

Mask R-CNN

Panoptic segmentation

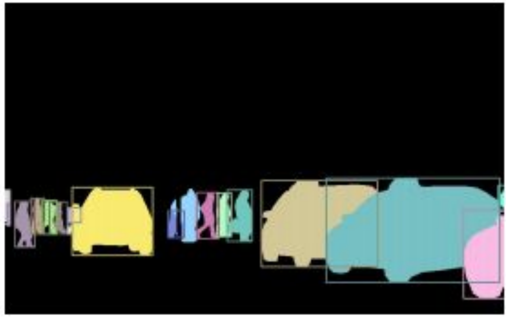
Semantic segmentation



FCN-like

+

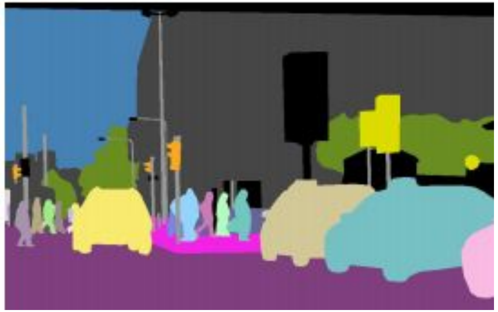
Instance segmentation



Mask R-CNN

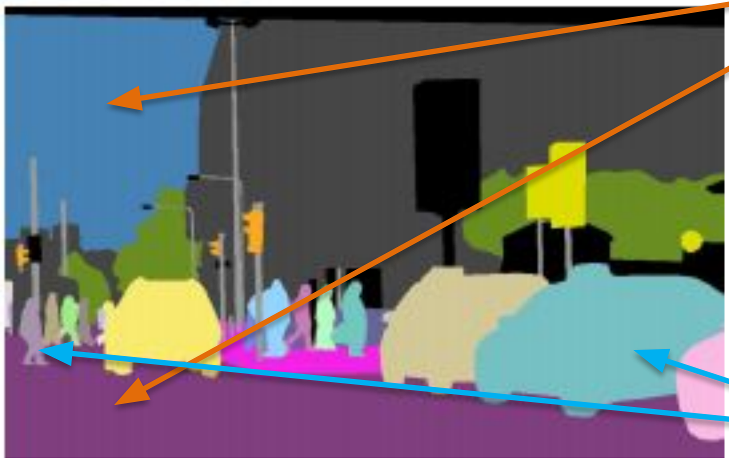
=

Panoptic segmentation



UPSNet

Panoptic segmentation



It gives labels to uncountable objects called "stuff" (sky, road, etc), similar to FCN-like networks.

It differentiates between pixels coming from different instances of the same class (countable objects) called "things" (cars, pedestrians, etc).

Panoptic segmentation

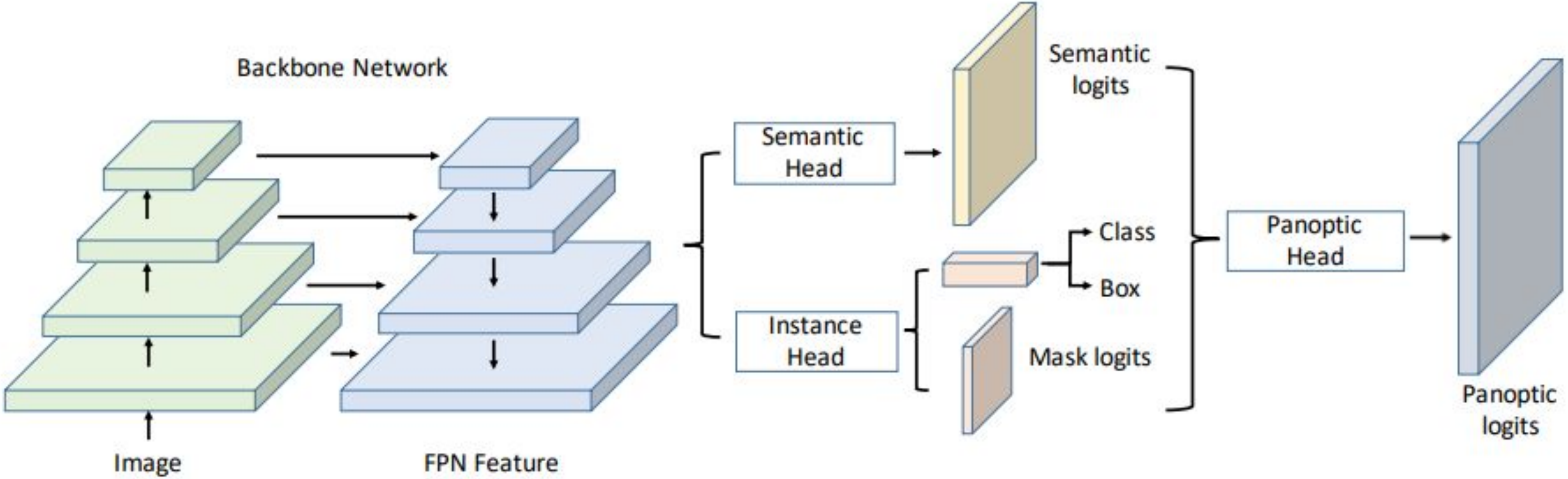
Problem: some pixels might get classified as stuff from FCN network, while at the same time being classified as instances of some class from Mask R-CNN (conflicting results)!

Panoptic segmentation

Solution: Parametric-free panoptic head which combines the information from the FCN and Mask R-CNN, giving final predictions.

Xiong et al., "UPSNet: A Unified Panoptic Segmentation Network". CVPR 2019

Network architecture

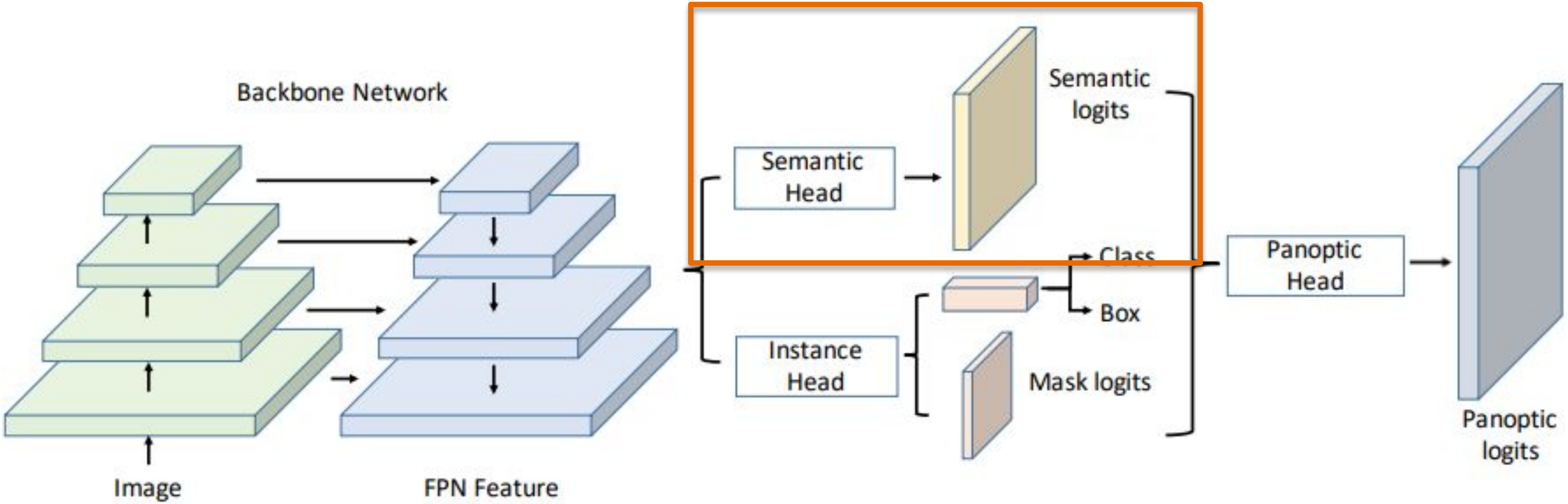


Shared features

Separate heads

Putting it together

Network architecture

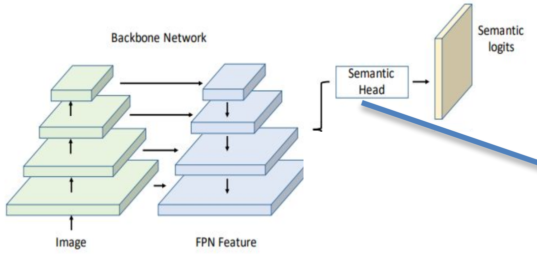


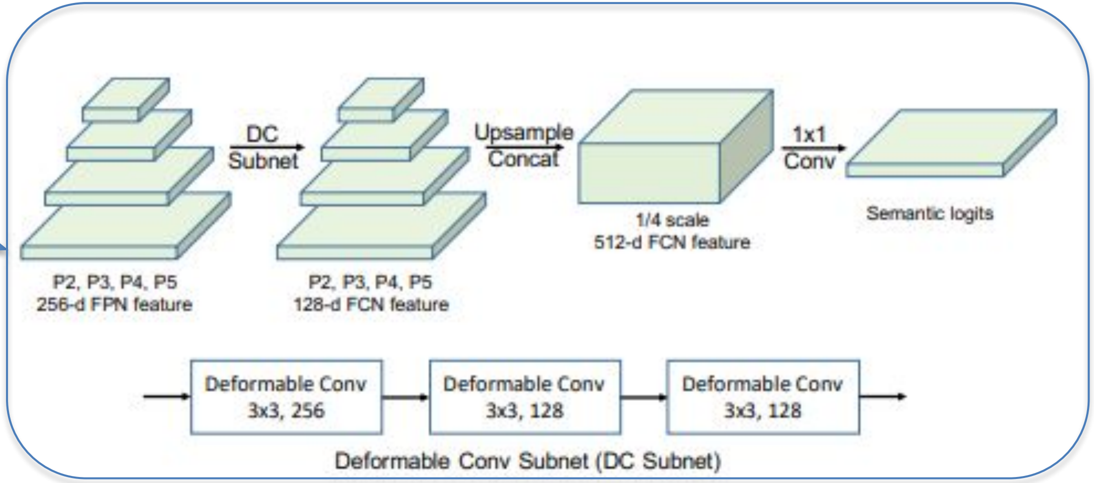
Shared features

Separate heads

Putting it together

The semantic head

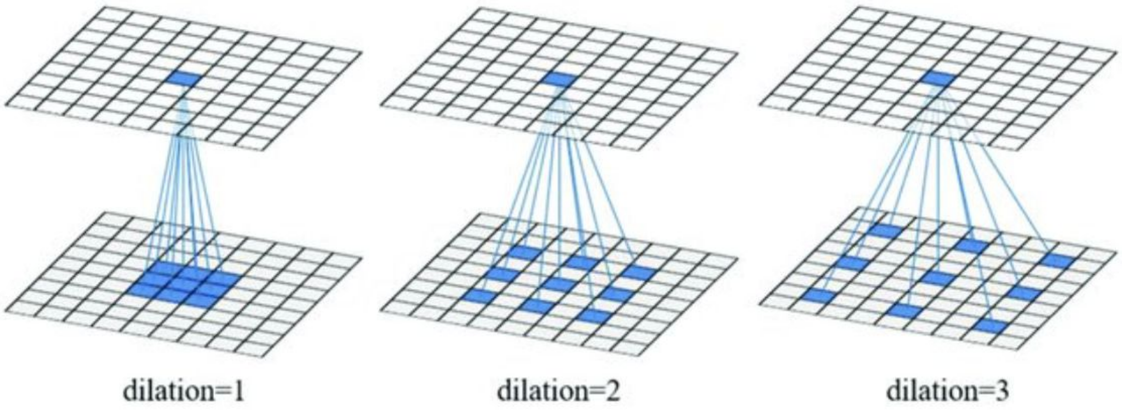




As all semantic heads, fully convolutional network.

New: deformable convolutions!

Dilated (atrous) convolutions 2D

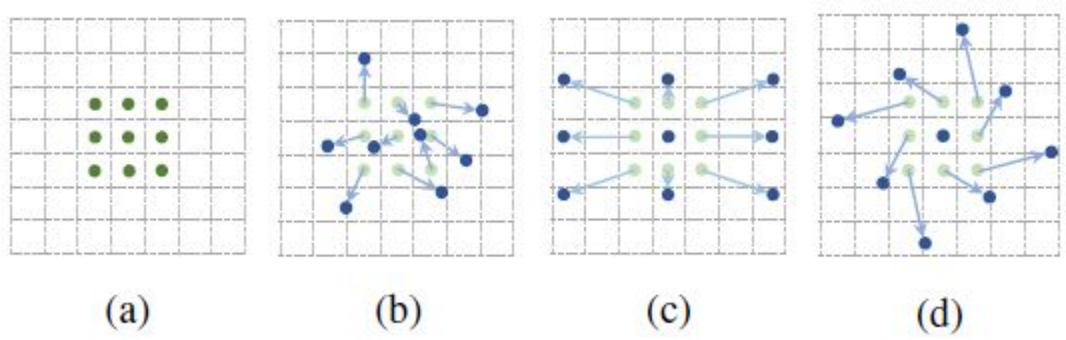


(a) the dilation parameter is 1, and each element produced by this filter has receptive field of 3x3.

(b) the dilation parameter is 2, and each element produced by it has receptive field of 5x5.

(c) the dilation parameter is 3, and each element produced by it has receptive field of 7x7.

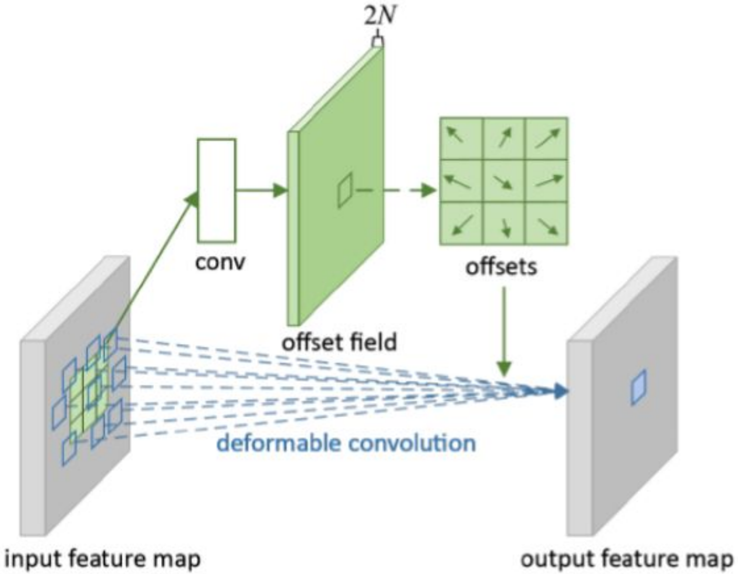
Deformable convolutions



(a) Conventional Convolution, (b) Deformable Convolution, (c) Special Case of Deformable Convolution with Scaling, (d) Special Case of Deformable Convolution with Rotation

Deformable convolutions: generalization of dilated convolutions when you learn the offset

Deformable convolutions



Regular convolution

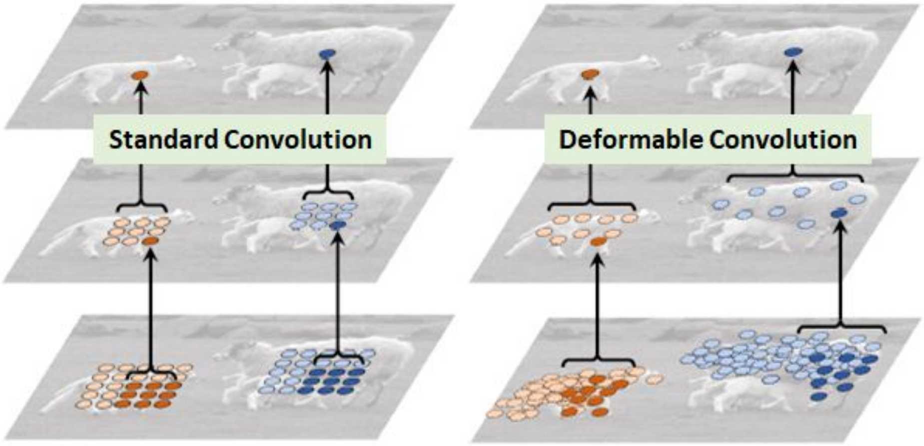
$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n)$$

Deformable convolution

$$\mathbf{y}(\mathbf{p}_0) = \sum_{\mathbf{p}_n \in \mathcal{R}} \mathbf{w}(\mathbf{p}_n) \cdot \mathbf{x}(\mathbf{p}_0 + \mathbf{p}_n + \Delta\mathbf{p}_n)$$

where $\Delta\mathbf{p}_n$ is generated by a sibling branch of regular convolution

Deformable convolutions



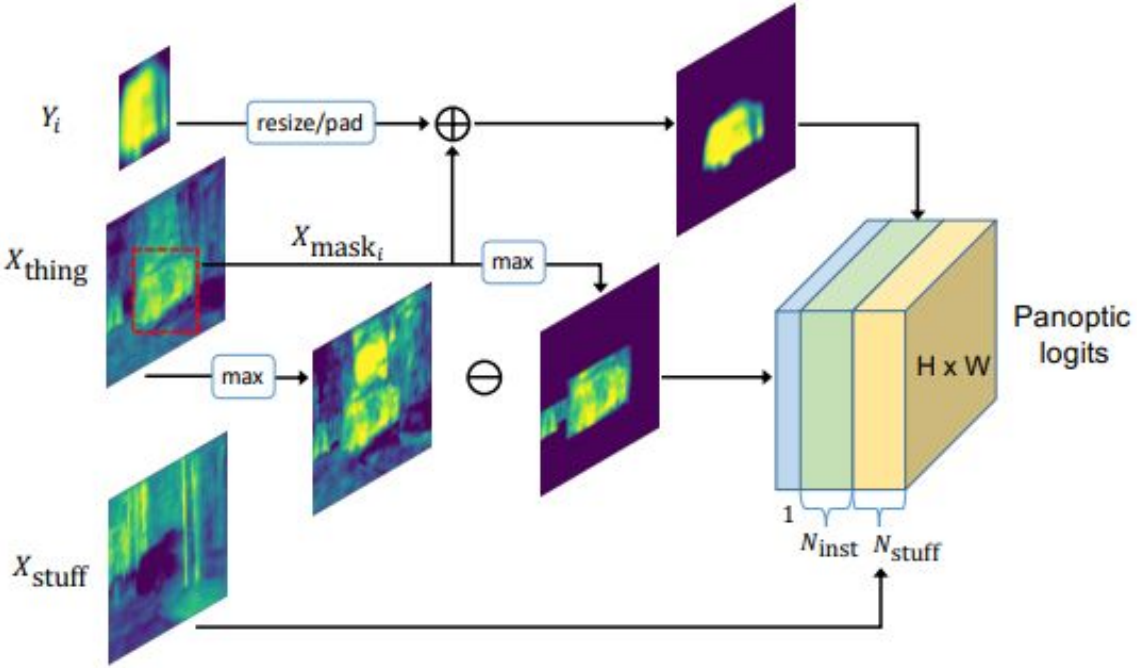
The deformable convolution will pick the values at different locations for convolutions conditioned on the input image of the feature maps.

The Panoptic head

Mask logits from the instance head

Object logits coming from the semantic head (e.g., car)

Stuff logits coming from the semantic head (e.g., sky)

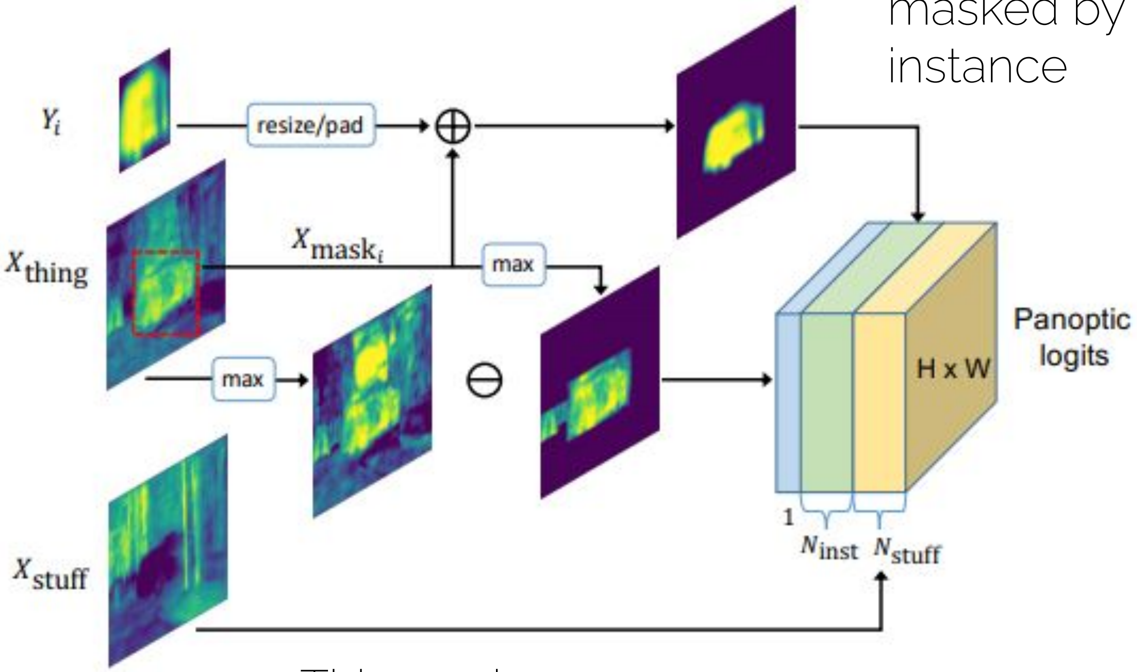


The Panoptic head

Mask logits from the instance head

Object logits coming from the semantic head (e.g., car)

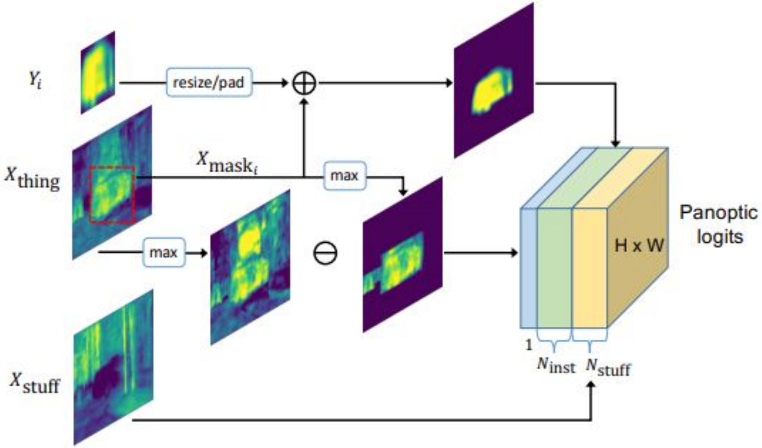
Stuff logits coming from the semantic head (e.g., sky)



Objects need to be masked by the instance

This can be evaluated directly

The Panoptic head



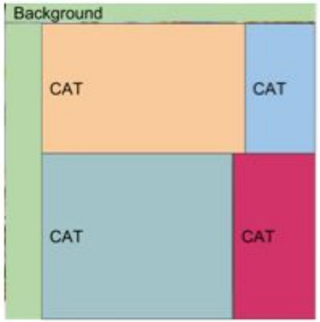
Perform softmax over the panoptic logits. If the maximum value falls into the first stuff channels, then it belongs to one of the stuff classes. Otherwise the index of the maximum value tells us the instance ID the pixel belongs to.

Read the details on how to use the unknown class

Xiong et al., "UPSNet: A Unified Panoptic Segmentation Network". CVPR 2019

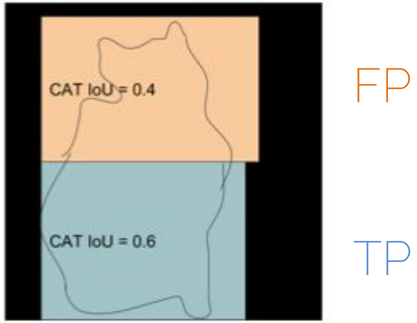
Panoptic quality

- As in detection, we have to "match ground truth and predictions. In this case we have segment matching.



Predictions

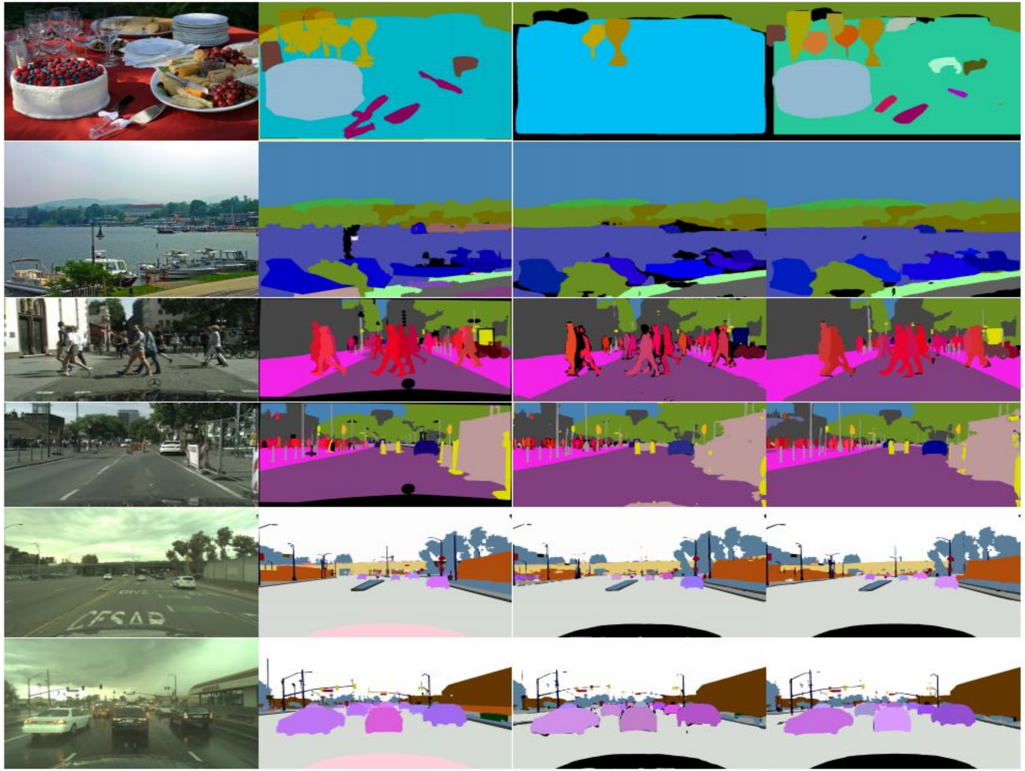
Ground truth



IoU measures

- Segment is matched if IoU>0.5. No pixel can belong to two predicted segments.

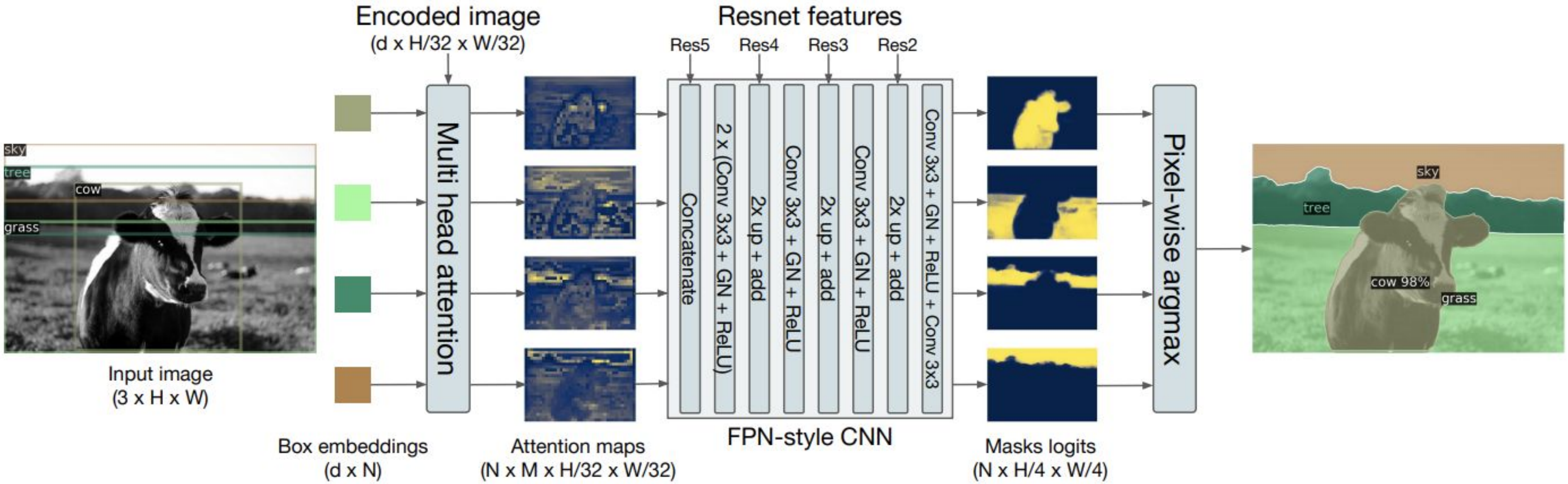
Panoptic segmentation: qualitative



Panoptic segmentation: qualitative

Image Ground truth Combined method [17] Ours

Of course, it can be done with Transformers

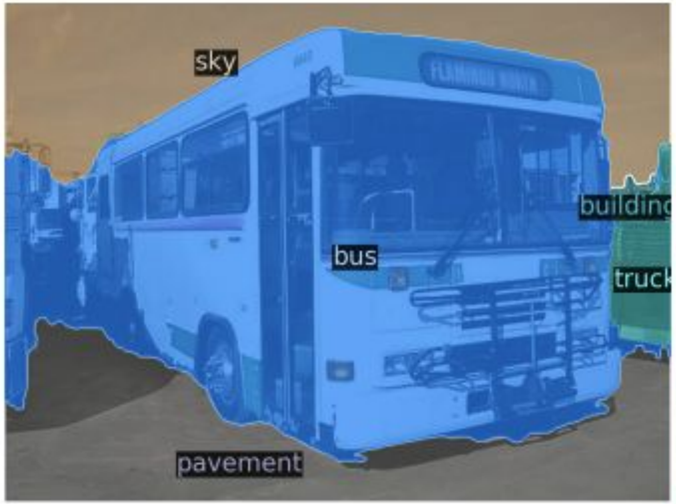


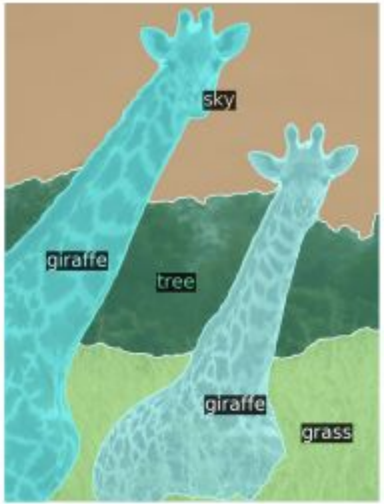
- A binary mask is generated in parallel for each detected object, then the masks are merged using pixel-wise argmax.

DETR Panoptic Segmentation – results

Model	Backbone	PQ	SQ	RQ	PQ^{th}	SQ^{th}	RQ^{th}	PQ^{st}	SQ^{st}	RQ^{st}	AP
PanopticFPN++	R50	42.4	79.3	51.6	49.2	82.4	58.8	32.3	74.8	40.6	37.7
UPSnet	R50	42.5	78.0	52.5	48.6	79.4	59.6	33.4	75.9	41.7	34.3
UPSnet-M	R50	43.0	79.1	52.8	48.9	79.7	59.7	34.1	78.2	42.3	34.3
PanopticFPN++	R101	44.1	79.5	53.3	**51.0**	**83.2**	60.6	33.6	74.0	42.1	**39.7**
DETR	R50	43.4	79.3	53.8	48.2	79.8	59.5	36.3	78.5	45.3	31.1
DETR-DC5	R50	44.6	79.8	55.0	49.4	80.5	60.6	**37.3**	**78.7**	**46.5**	31.9
DETR-R101	R101	**45.1**	**79.9**	**55.5**	50.5	80.9	**61.7**	37.0	78.5	46.0	33.0

DETR Panoptic Segmentation – results



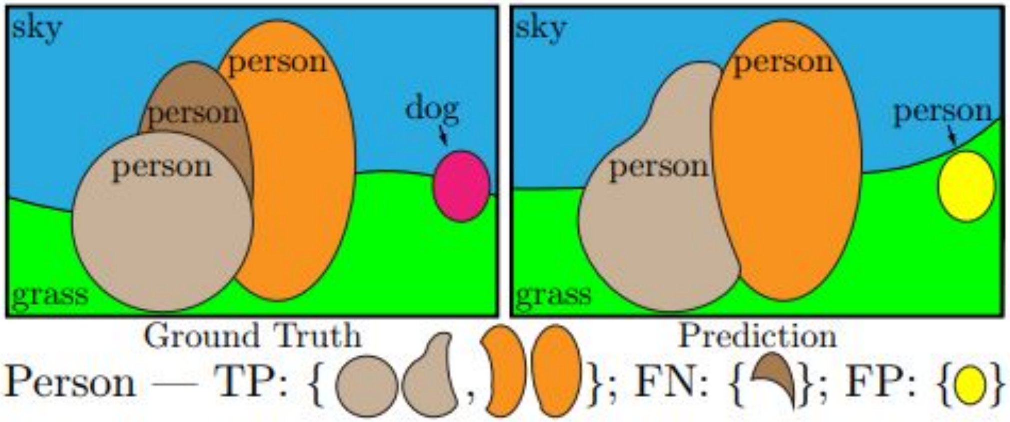


Metrics

Panoptic quality

$$PQ = \underbrace{\frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP|}}_{\text{SQ}} \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{RQ}}$$

TP = True positive, FN = False negative, FP = false positive



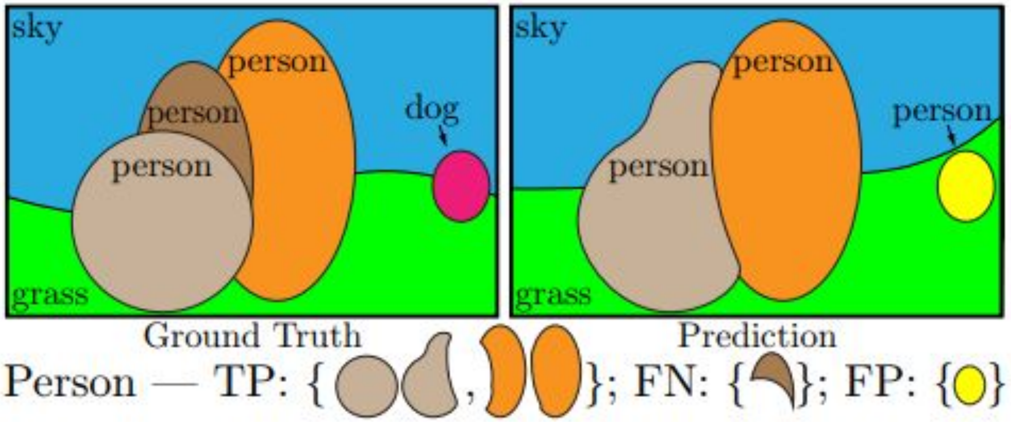
- SQ: Segmentation Quality = how close the predicted segments are to the ground truth segment

(does not take into account bad predictions!)

Panoptic quality

TP = True positive, FN = False negative, FP = false positive

$$PQ = \underbrace{\frac{\sum_{(p,g)\in \mathit{TP}} \mathrm{IoU}(p,g)}{|\mathit{TP}|}}_{\text{SQ}} \underbrace{\frac{|\mathit{TP}|}{|\mathit{TP}| + \frac{1}{2}|\mathit{FP}| + \frac{1}{2}|\mathit{FN}|}}_{\text{RQ}}$$



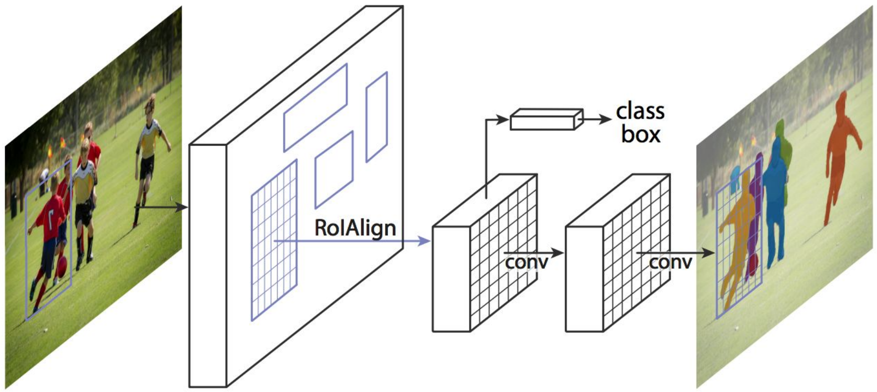
- RQ: Recognition Quality = just like for detection, we want to know if we are missing any instances (FN) or we are predicting more instances (FP).

Instance segmentation

Object Instance Segmentation as Voting

Sliding Window Approach

- DPM, RCNN families
- Densely enumerate box proposals + classify
- Tremendously successful paradigm, very well engineered
- SOTA methods are still based on this paradigm



Generalized Hough Transform

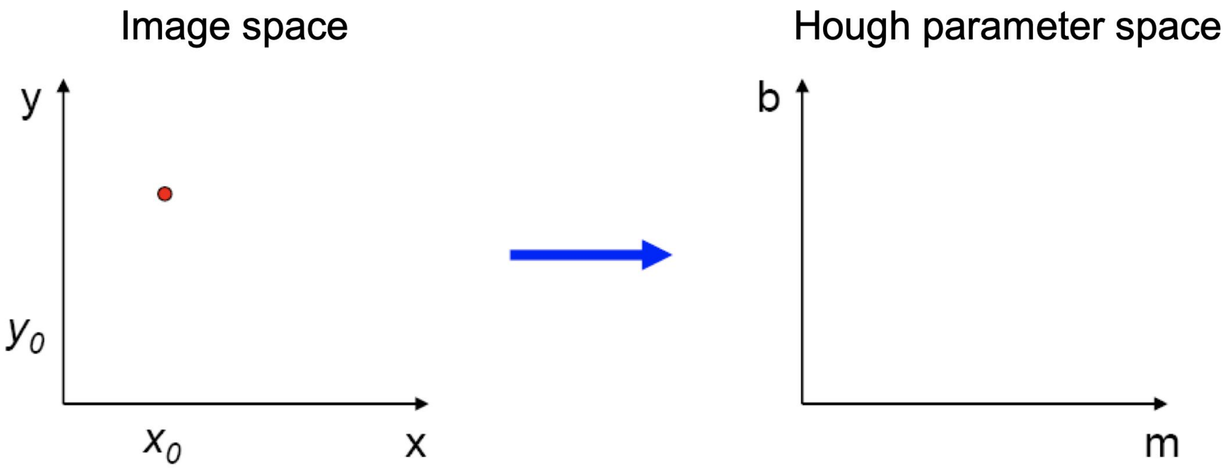
Before DPM, RCNN dominance: detection-as-voting

Hough Voting

- Detect analytical shapes (e.g., lines) as peaks in the dual parametric space
- Each pixel casts a vote in this dual space
- Detect peaks and 'back-project' them to the image space

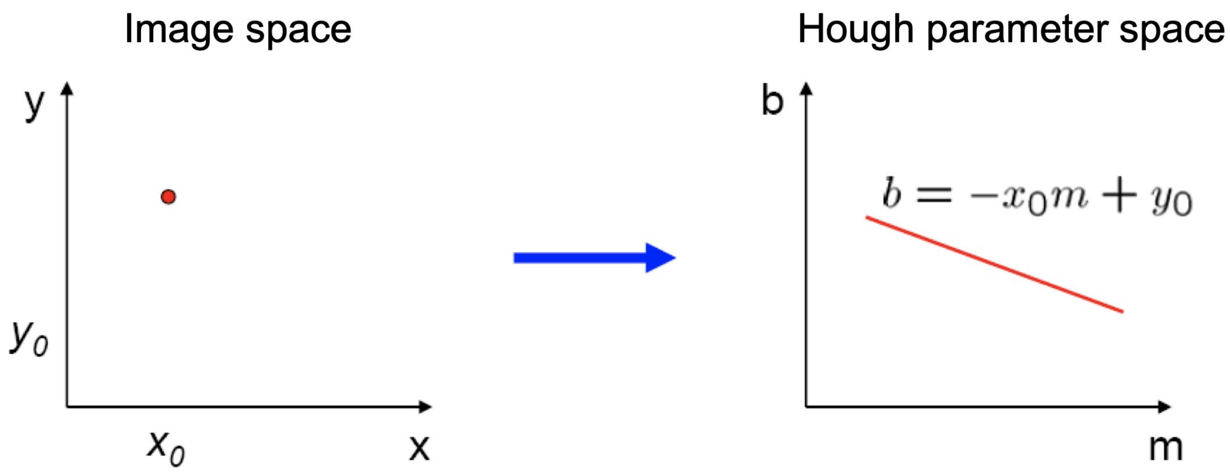
Example: Line Detection

- Each edge point in image space casts a vote



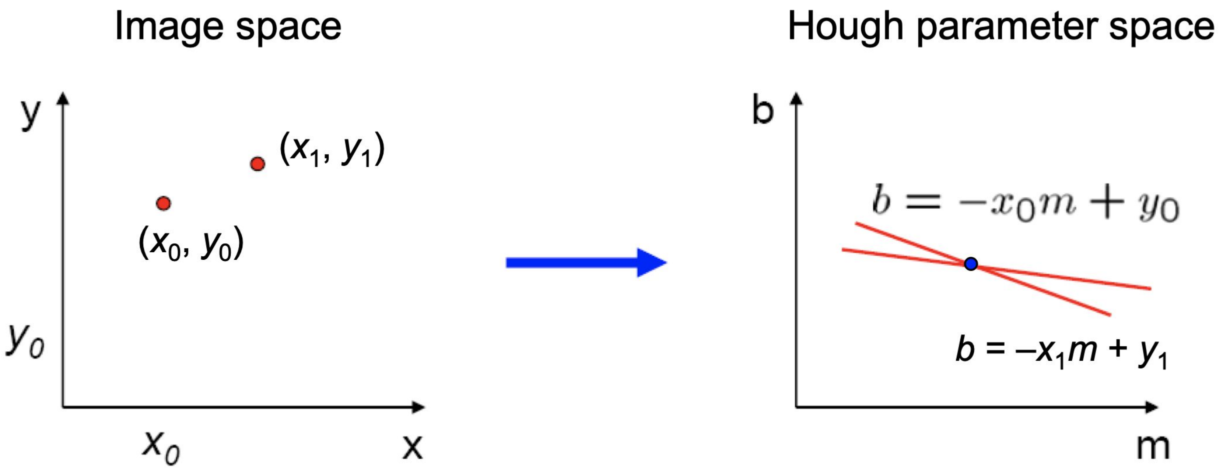
Example: Line Detection

- Each edge point in image space casts a vote
- The vote is in the form of a line that crosses the point



Example: Line Detection

- Accumulate votes from different points in (discretized) parameter space
- Read-out maxima (peaks) from the accumulator



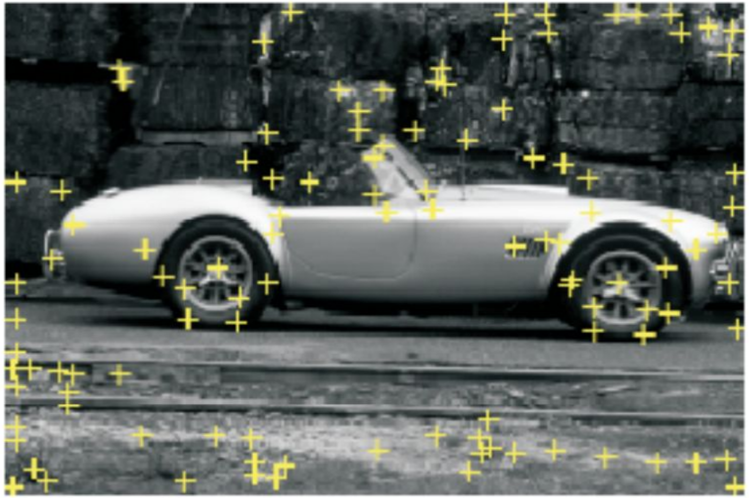
Object Detection as Voting

- Idea: Objects are detected as consistent configurations of the observed parts (visual words)

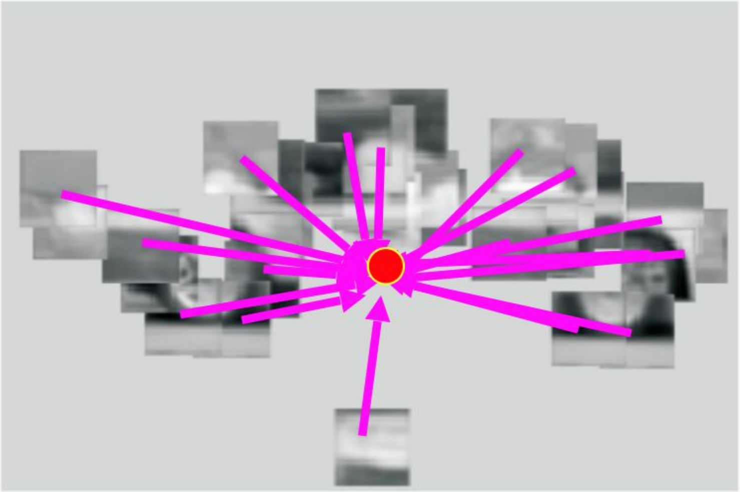
Leibe et al., Robust Object Detection with Interleaved Categorization and Segmentation, IJCV'08

Object Detection

- Training



Interest point detection
(SIFT, SURF)

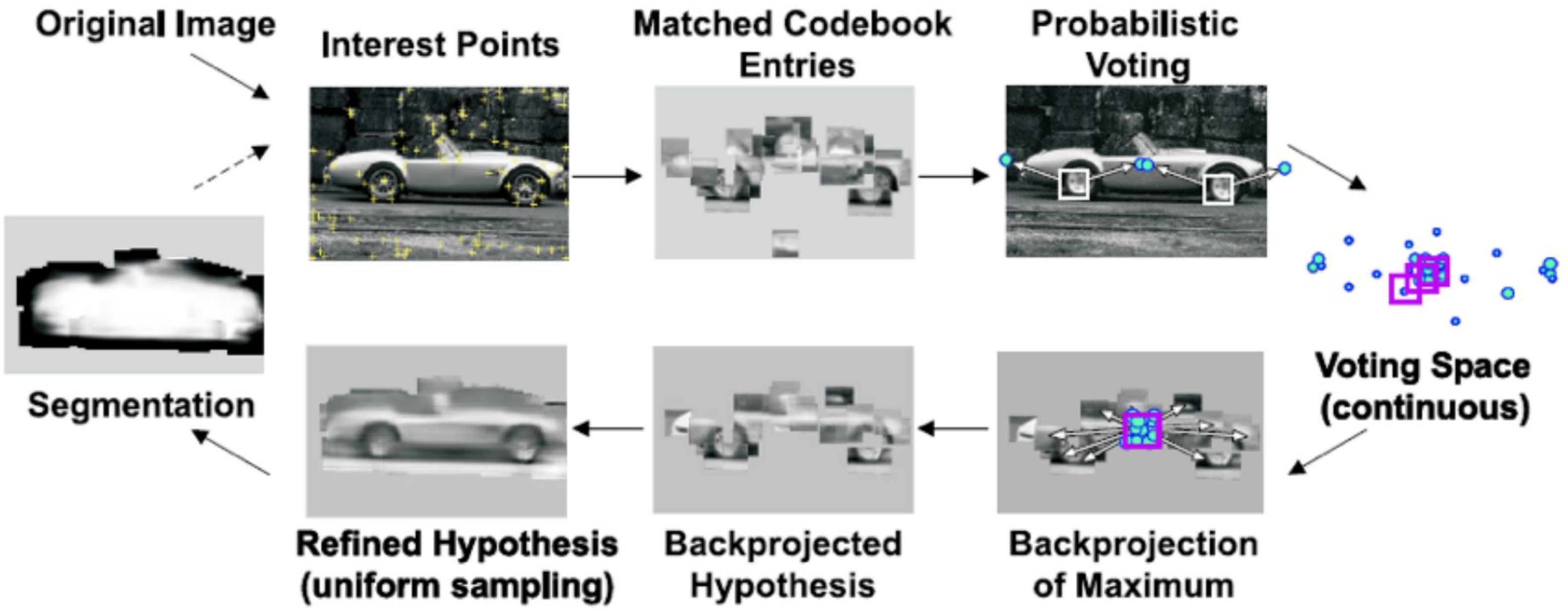


Center point voting

Leibe et al., Robust Object Detection with Interleaved Categorization and Segmentation, IJCV'08

Object Detection

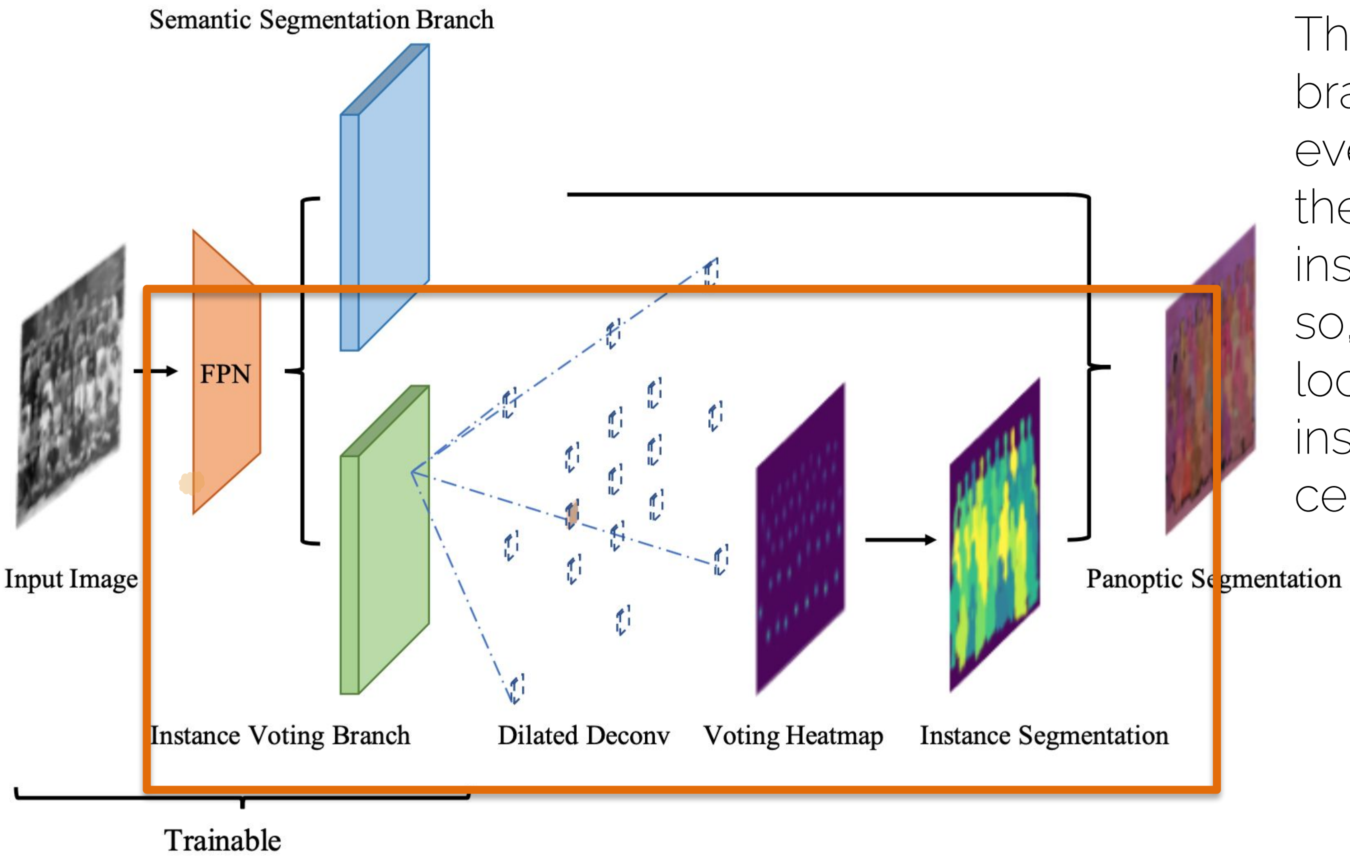
- Inference (test time)



Back to the future

- Back to 2020…
- We can use pixel consensus voting for panoptic segmentation (CVPR 20)

Overview



The instance voting branch predicts for every pixel whether the pixel is part of an instance mask, and if so, the relative location of the instance mask centroid.

H. Wang et al. "Pixel Consensus Voting for Panoptic Segmentation" CVPR 2020

In a Nutshell

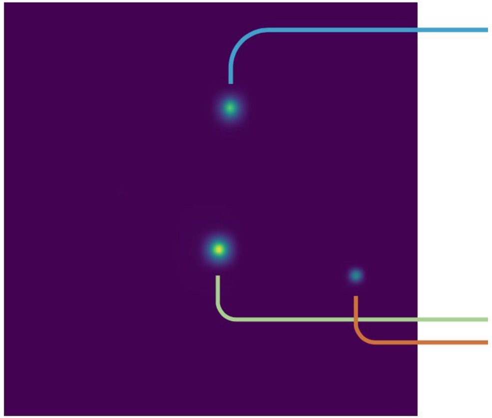
1. Discretize regions around each pixel.
2. Every pixel votes for a centroid (or no centroid for "stuff") over a set of grid cells.

Discretization & Classification

In a Nutshell

3. Vote aggregation probabilities at each pixel are cast to accumulator space via (dilated) transposed convolutions
4. Detect objects as 'peaks' in the accumulator space

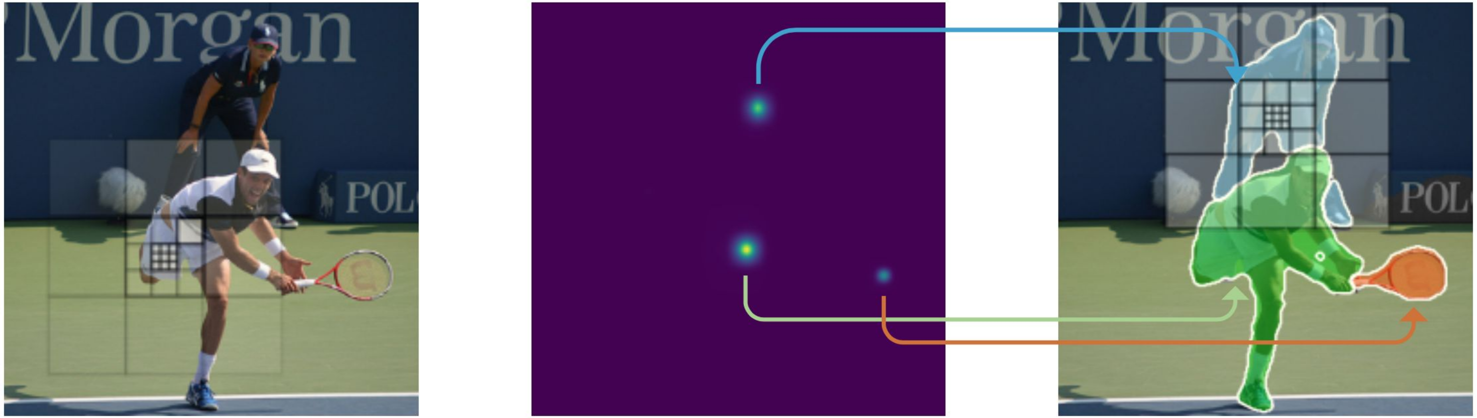
Discretization & Classification



Voting as Transposed Convolution

In a Nutshell

5. Back-projection of 'peaks' back to the image to get an instance masks
6. Category information provided by the parallel semantic segmentation head



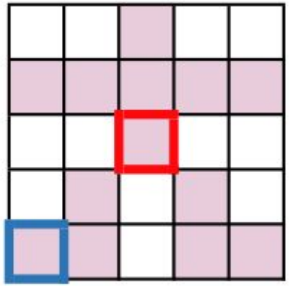
Discretization & Classification   Voting as Transposed Convolution   Backprojection as Filtering

Voting Lookup Table

- Discretize region around the pixel: M×M cells converted into K=17 indices.

10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
11	11	11	2	1	8	15	15	15
11	11	11	3	0	7	15	15	15
11	11	11	4	5	6	15	15	15
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14

Voting filter



Instance Mask

10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
11	11	11	2	1	8	15	15	15
11	11	11	3	0	7	15	15	15
11	11	11	4	5	6	15	15	15
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14

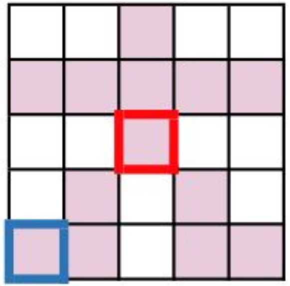
Ground Truth Assignment

Voting Lookup Table

- The vote should be cast to the center, which is the red pixel, which corresponds to position 16.

10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
11	11	11	2	1	8	15	15	15
11	11	11	3	0	7	15	15	15
11	11	11	4	5	6	15	15	15
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14

Voting filter



Instance Mask

10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
11	11	11	2	1	8	15	15	15
11	11	11	3	0	7	15	15	15
11	11	11	4	5	6	15	15	15
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14

Ground Truth Assignment

Voting

- At inference, instance voting branch provides tensor of size [H,W,K+1]
- Softly accumulate votes in the voting accumulator. **How?**

10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
11	11	11	2	1	8	15	15	15
11	11	11	3	0	7	15	15	15
11	11	11	4	5	6	15	15	15
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14

Example: for the blue pixel, we get a vote for index 16 with 0.9 probability (softmax output)

- Transfer 0.9 to cell 16 -- (dilated) **transposed convolution**
- Evenly distribute among pixels, each gets 0.1 -- **average pooling**

Transposed Convolutions

- Take a single value in the input
- Multiply with a kernel and *distribute* in the output map
  - Kernel *defines* the amount of the input value that is being distributed to each of the output cells
- For the purpose of vote aggregation, however, we fix the kernel parameters to 1-hot across each channel that marks the target location.

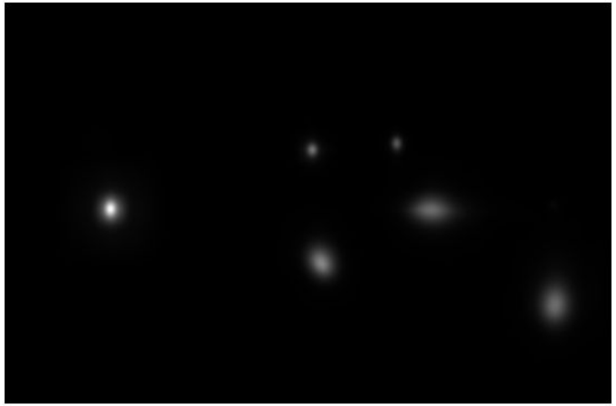
Voting - Implementation

- Output tensor: [H,W,K+1]
- Example: 9 inner, 8 outer bins, K=17
- Split the output tensor to two tensors: [H,W,9],[H,W,8]
  - Apply two transposed convolutions, with kernel of size [3,3,9], stride=1 and [3,3,8], stride=3
  - Pre-fixed kernel parameters; 1-hot across each channel that marks the target location
  - Dilation => spread votes to the outer ring
- Smooth votes evenly via average pooling

10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
10	10	10	9	9	9	16	16	16
11	11	11	2	1	8	15	15	15
11	11	11	3	0	7	15	15	15
11	11	11	4	5	6	15	15	15
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14
12	12	12	13	13	13	14	14	14

Object Detection

- Peaks in the heatmap -- consensus detections
- Thresholding + connected components

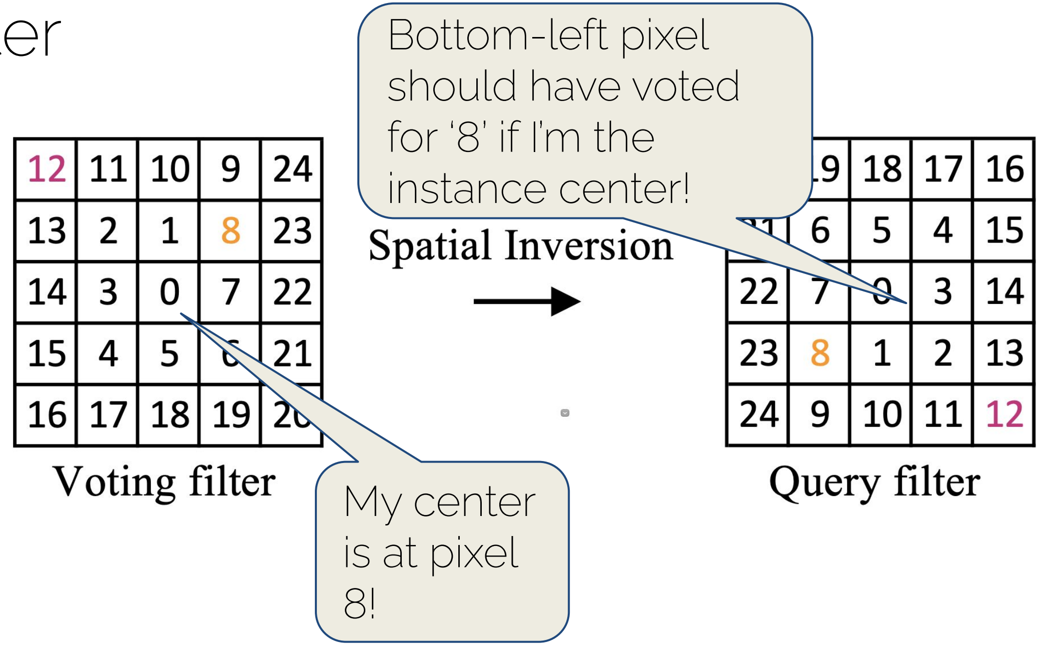


Object Localization

- Vote back-projection
  - For every peak, determine pixels that favor this region above all others

Object Localization

- Idea: determine which pixels could have voted for a specific object center
  - Query filter
- Examine votes
  - Vote argmax
- Find ``consensus``
  - Equality test

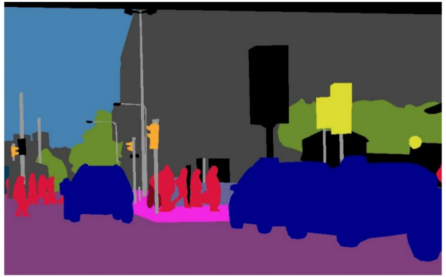


Qualitative Results

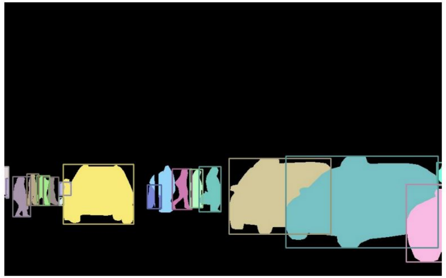
Fine-grained Scene Interpretation

- Individual objects, surfaces (things and stuff)
- Mobile robots
  - Reason about the drivability of surfaces.
  - The type of objects and obstacles.
  - The intent of other agents in the vicinity.

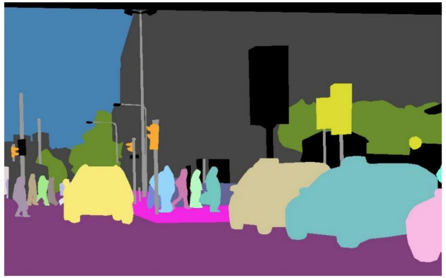
(a) image



(b) semantic segmentation



(c) instance segmentation



(d) panoptic segmentation

Instance segmentation