Instance Segmentation

韩坤洋

Papers

- Mask R-CNN
- PointRend: Image Segmentation as Rendering
- YOLACT: Real-time Instance Segmentation
- PolarMask: single shot instance segmentation with polar representation
- SOLO: segmenting objects by locations

Mask R-CNN

- Based on faster rcnn
- Output a binary mask for each RoI

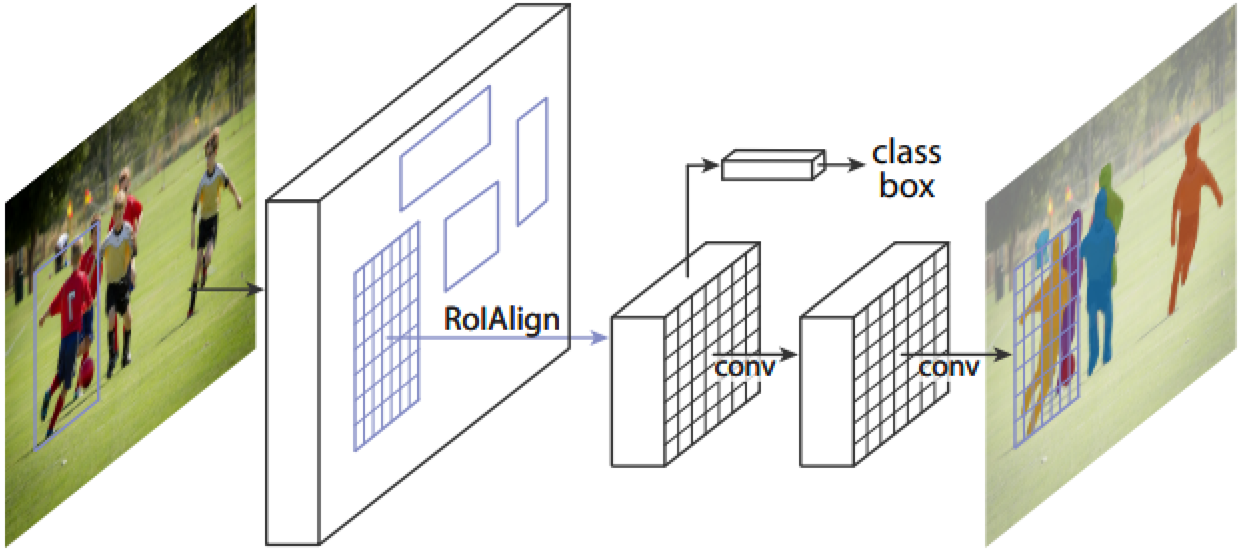
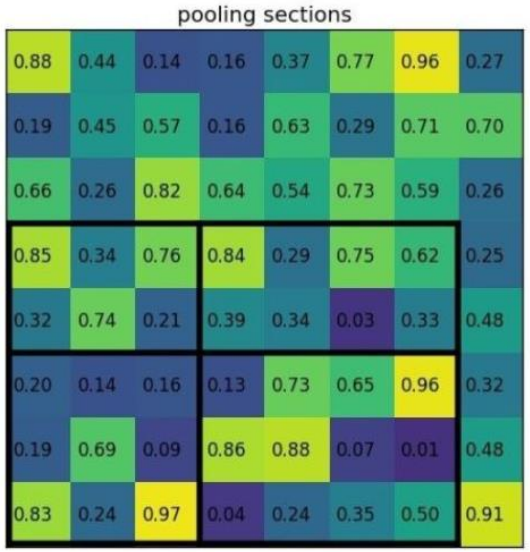


Figure 1. The **Mask R-CNN** framework for instance segmentation.

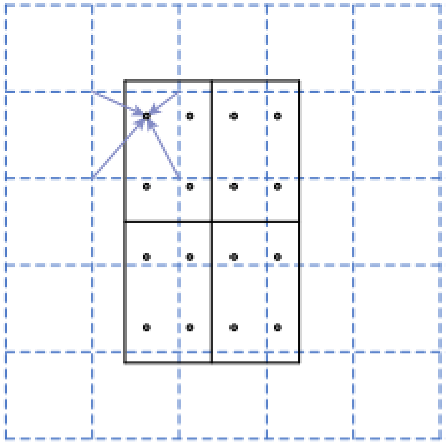
Mask R-CNN RoIAlign

- RoIPool first quantizes a floating-number RoI to the discrete granularity of the feature map
- Then subdivided into spatial bins which are themselves quantized
- Cause misalignments



pooling sections

0.88	0.44	0.14	0.16	0.37	0.77	0.96	0.27
0.19	0.45	0.57	0.16	0.63	0.29	0.71	0.70
0.66	0.26	0.82	0.64	0.54	0.73	0.59	0.26
0.85	0.34	0.76	0.84	0.29	0.75	0.62	0.25
0.32	0.74	0.21	0.39	0.34	0.03	0.33	0.48
0.20	0.14	0.16	0.13	0.73	0.65	0.96	0.32
0.19	0.69	0.09	0.86	0.88	0.07	0.01	0.48
0.83	0.24	0.97	0.04	0.24	0.35	0.50	0.91



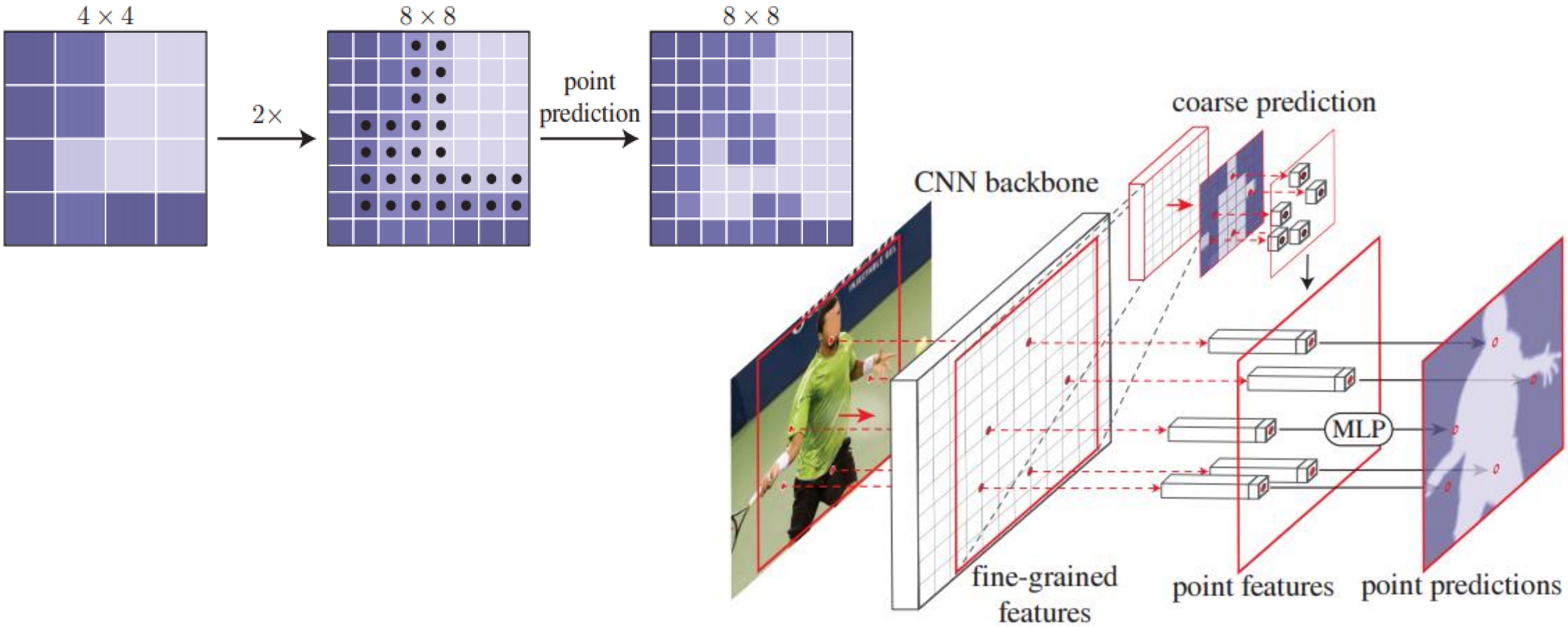
Mask R-CNN Loss

- Define a multi-task loss on each sampled RoI

$$L = L_{cls} + L_{box} + L_{mask}$$

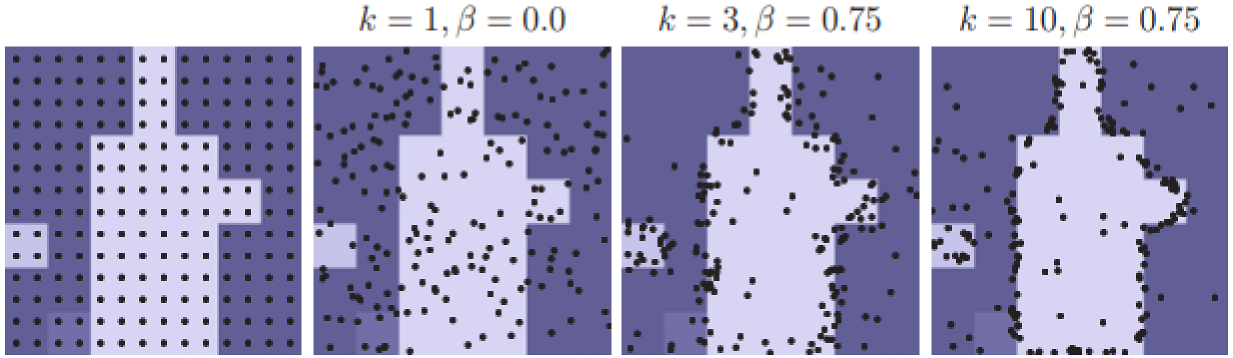
- Mask branch has a K x m x m output for each RoI
    - m*m resolution, k classes
    - per-pixel sigmoid
    - average binary cross-entropy loss
    - for RoI with ground-truth class k, L_{mask} is only defined on the k-th mask

PointRend: Image Segmentation as Rendering



PointRend Point Selection

- Inference: adaptive subdivision (in rendering
- Train: to select N points
  - Over generation: select kN points randomly
  - Importance sampling: select most uncertain βN points (β ∈ [0, 1])
  - Coverage: (1 - β)N points sampled from a uniform distribution



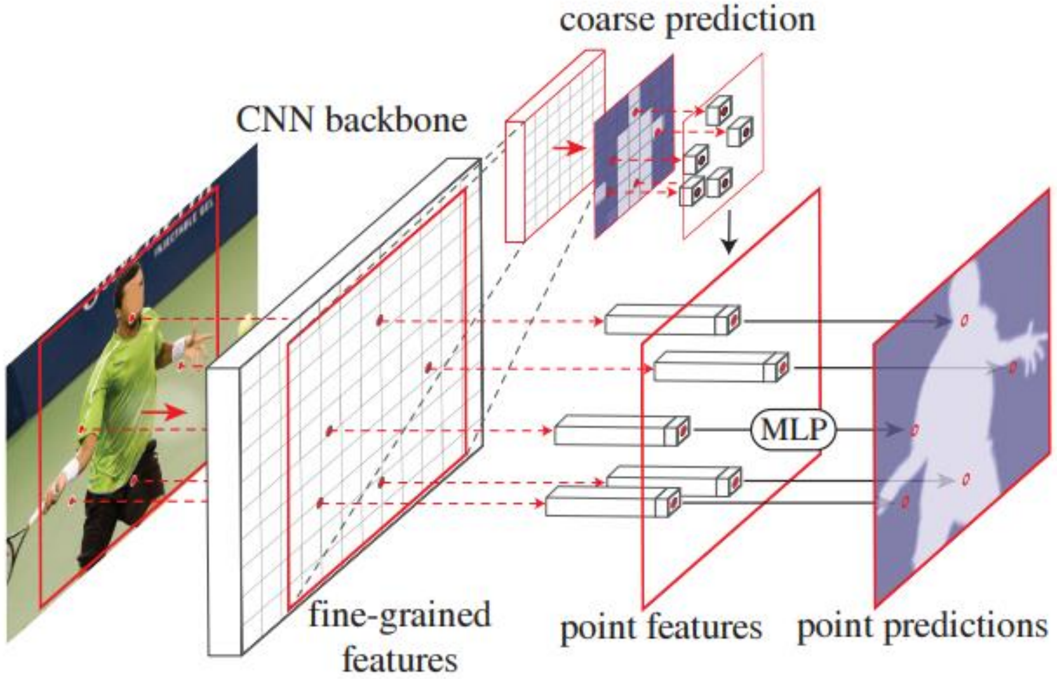
$k = 1, \beta = 0.0$ $k = 3, \beta = 0.75$ $k = 10, \beta = 0.75$

a) regular grid b) uniform c) mildly biased d) heavily biased

We use the distance between 0.5 and the probability of the ground truth class interpolated from the coarse prediction as the point-wise uncertainty measure.

PointRend Point-wise Representation

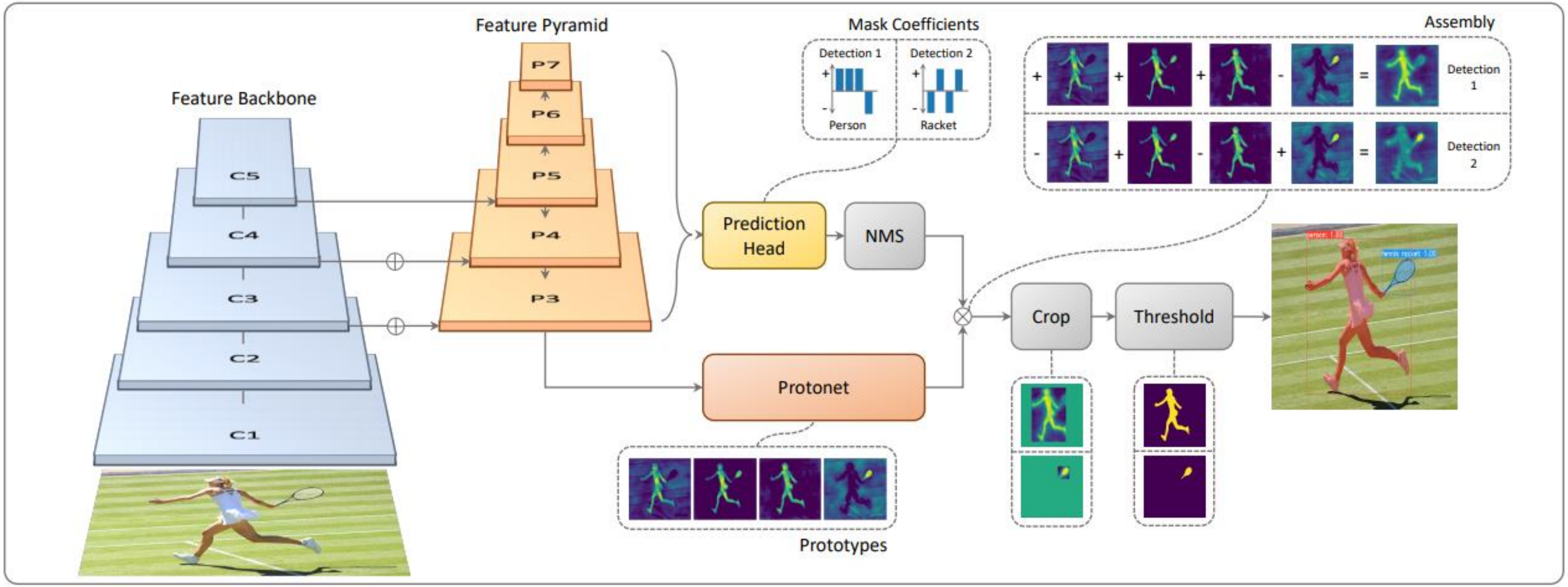
- Fine-grained features: CNN feature maps
  - fine segmentation details
  - bilinear interpolation
  - single or multiple feature map (res2 or res2 to res5 in a ResNet
- Coarse prediction features
  - region-specific information
  - contextual and semantic information



PointRend

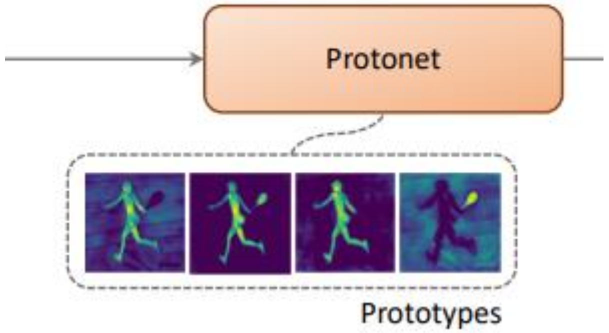
- Point head: MLP
  - shares weights across all points
  - binary cross-entropy
- Train:
  - sample 14^2 points (ROI_MASK_HEAD.POOLER_RESOLUTION)
  - One time (does not improve the baseline Mask R-CNN
  - Bilinear GT
- Inference
  - Top-k
  - Cascade

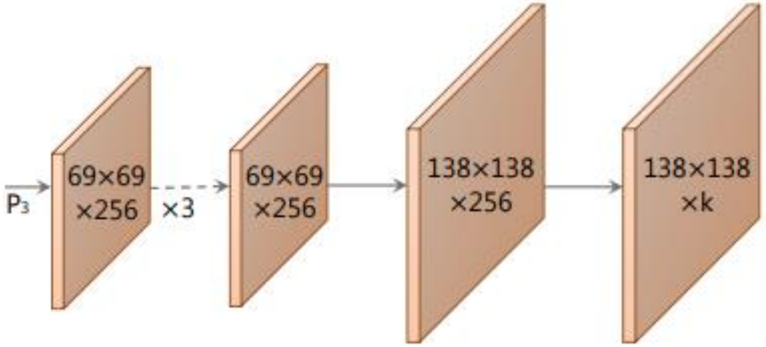
YOLACT


Feature Pyramid
Mask Coefficients
Assembly
Feature Backbone
C5
C4
C3
C2
C1
P7
P6
P5
P4
P3
Detection 1
Detection 2
Person
Racket
Detection 1
Detection 2
Prediction Head
NMS
Crop
Threshold
Protonet
Prototypes

YOLACT Prototype Generation (protonet

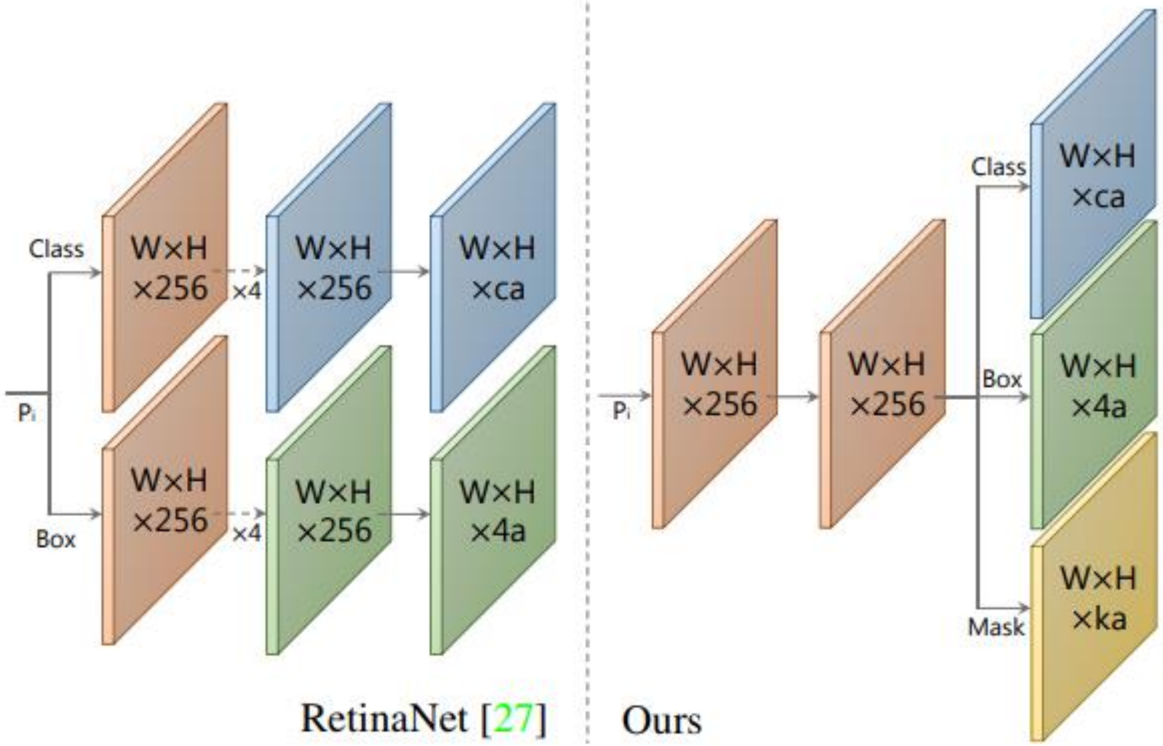
- FCN structure,
  - k channel out (k prototypes
  - No extra loss
- Attached to p3 in FPN
  - deeper backbone features, more robust
  - higher resolution prototypes, higher quality masks





YOLACT Mask Coefficients

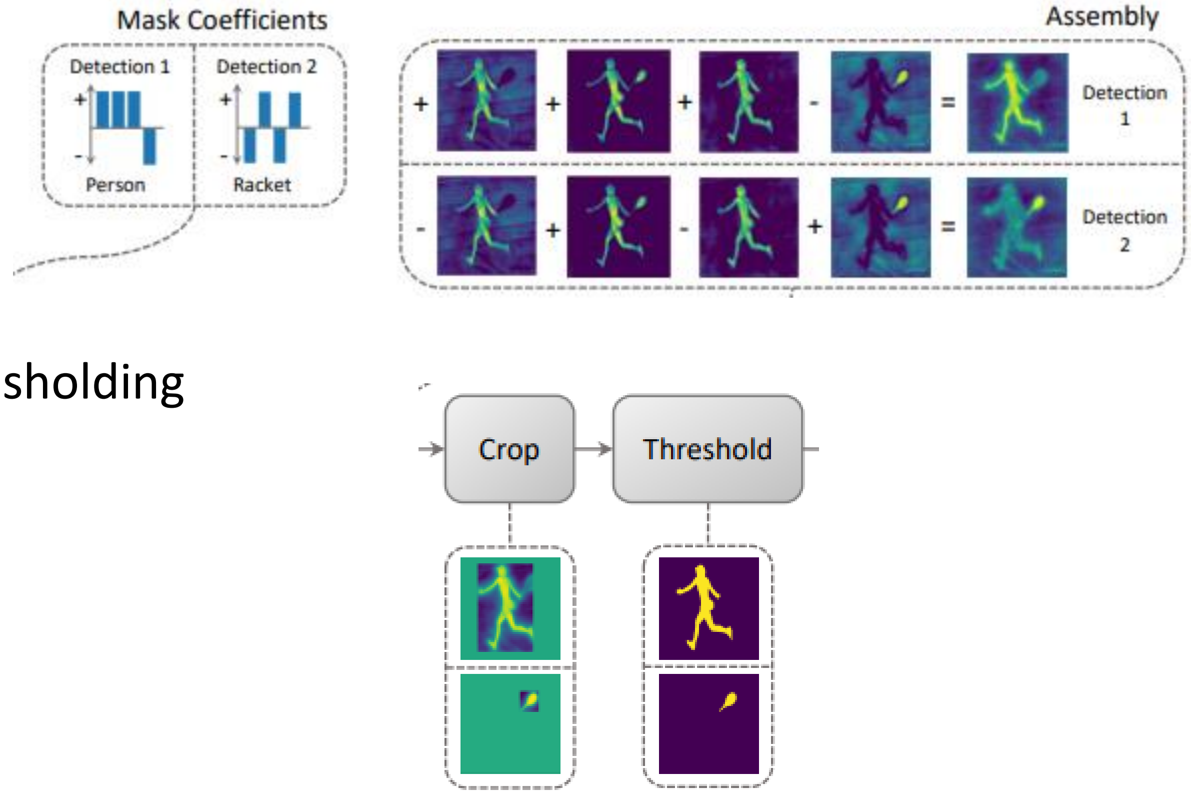
- Three branch
  - c class confidences
  - 4 bounding box regressors
  - k mask coefficients (k prototypes
- Nonlinearity
  - tanh to the k mask coefficients



YOLACT Mask Assembly

- Linear combination (matrix multiplication and sigmoid

$$M = \sigma(PC^T)$$



  - P , prototype masks, h × w × k
  - C , mask coefficients, n × k
    - n instances, surviving NMS and score thresholding
- Cropping Masks
  - Evaluation, predicted bounding box
  - Training, ground truth bounding box

YOLACT Prototype Behavior

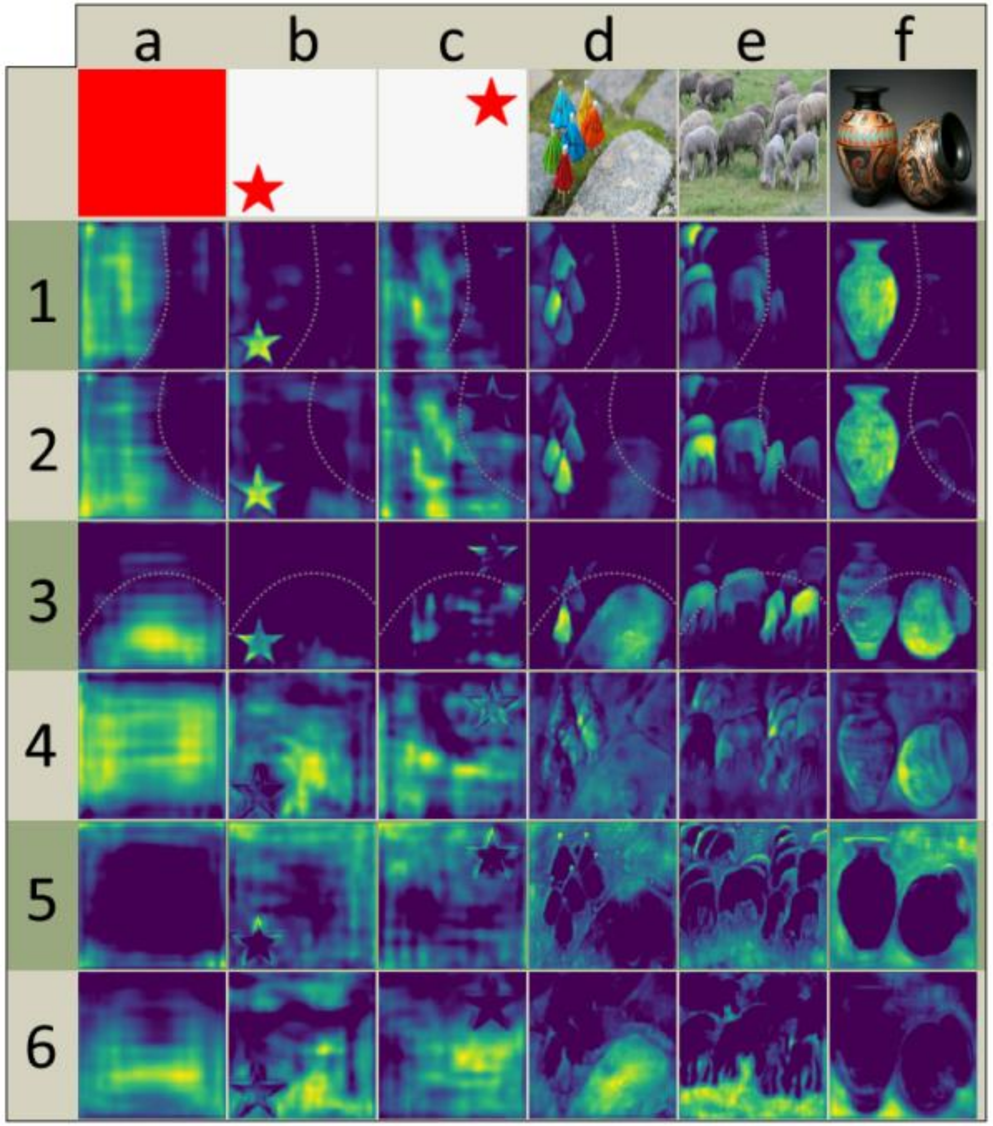


Figure 5: **Prototype Behavior** The activations of the same six prototypes (y axis) across different images (x axis). Prototypes 1-3 respond to objects to one side of a soft, implicit boundary (marked with a dotted line). Prototype 4 activates on the bottom-left of objects (for instance, the bottom left of the umbrellas in image d); prototype 5 activates on the background and on the edges between objects; and prototype 6 segments what the network perceives to be the ground in the image. These last 3 patterns are most clear in images d-f.

PolarMask : Single Shot Instance Segmentation with Polar Representation

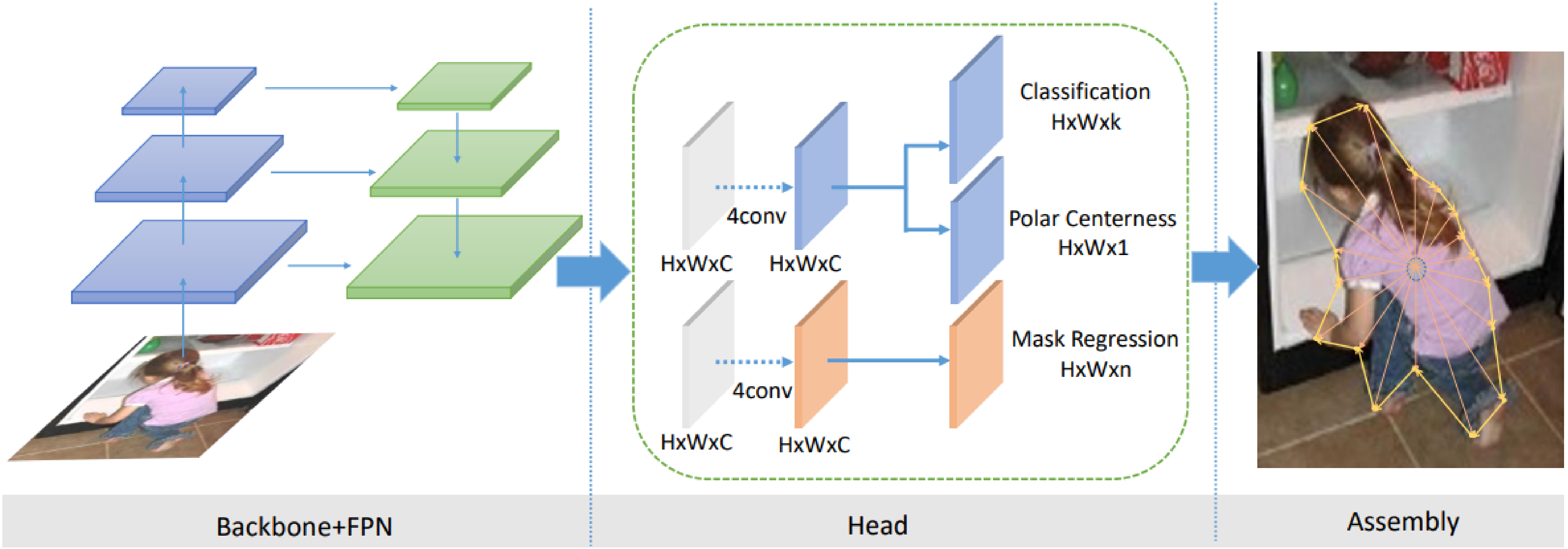
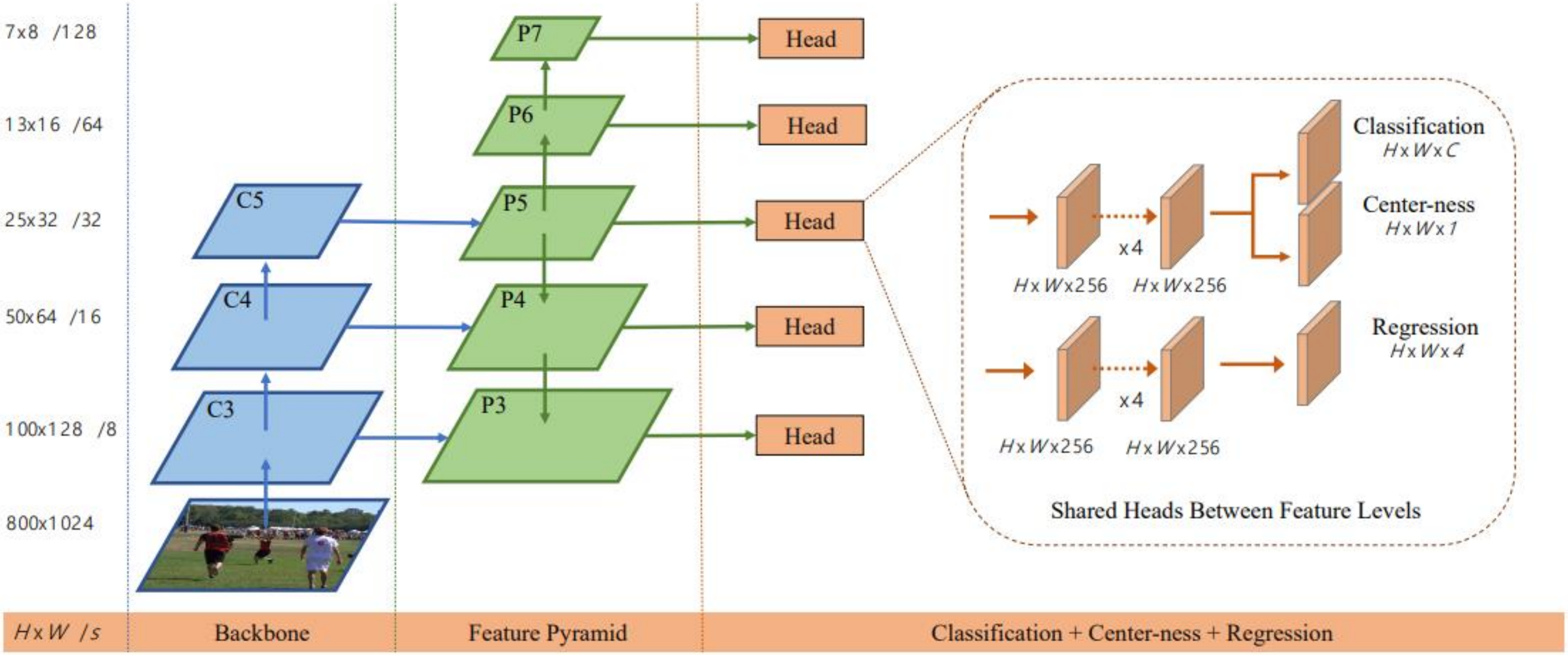
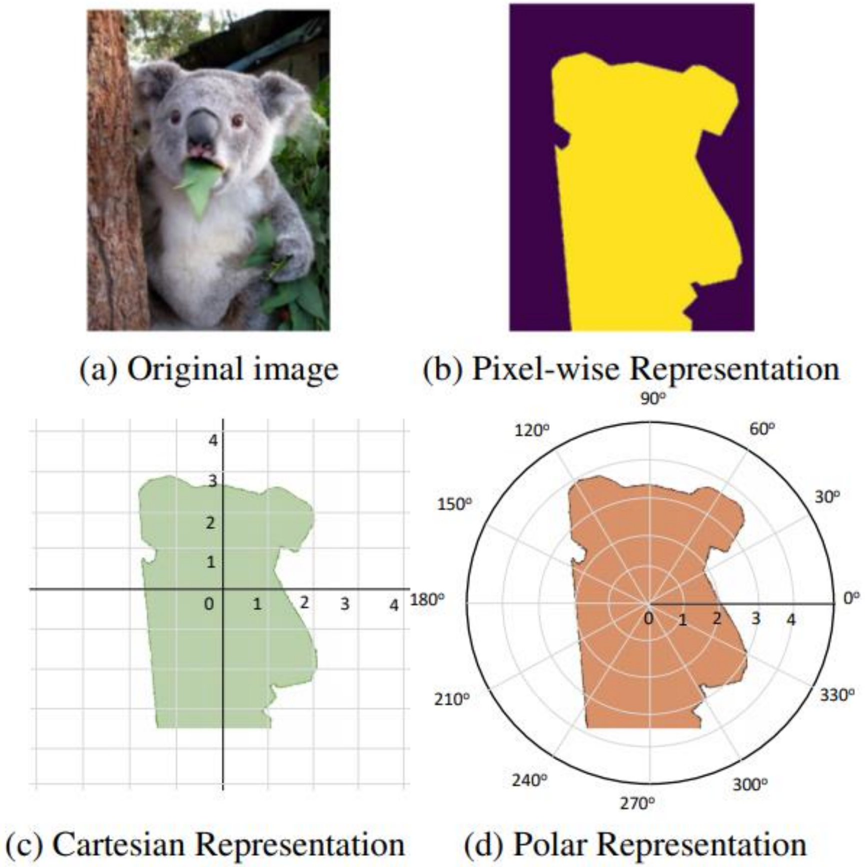


Figure 2 – The overall pipeline of PolarMask. The left part contains the backbone and feature pyramid to extract features of different levels. The middle part is the two heads for classification and polar mask regression. H, W, C are the height, width, channels of feature maps, respectively, and k is the number of categories (e.g., $k = 80$ on the COCO dataset), n is the number of rays (e.g., $n = 36$)

FCOS


7x8 /128
13x16 /64
25x32 /32
50x64 /16
100x128 /8
800x1024
C5
C4
C3
P7
P6
P5
P4
P3
Head
Head
Head
Head
Head
Classification
$H \times W \times C$
Center-ness
$H \times W \times 1$
Regression
$H \times W \times 4$
x4
x4
$H \times W \times 256$
$H \times W \times 256$
$H \times W \times 256$
$H \times W \times 256$
Shared Heads Between Feature Levels
$H \times W$ /s
Backbone
Feature Pyramid
Classification + Center-ness + Regression

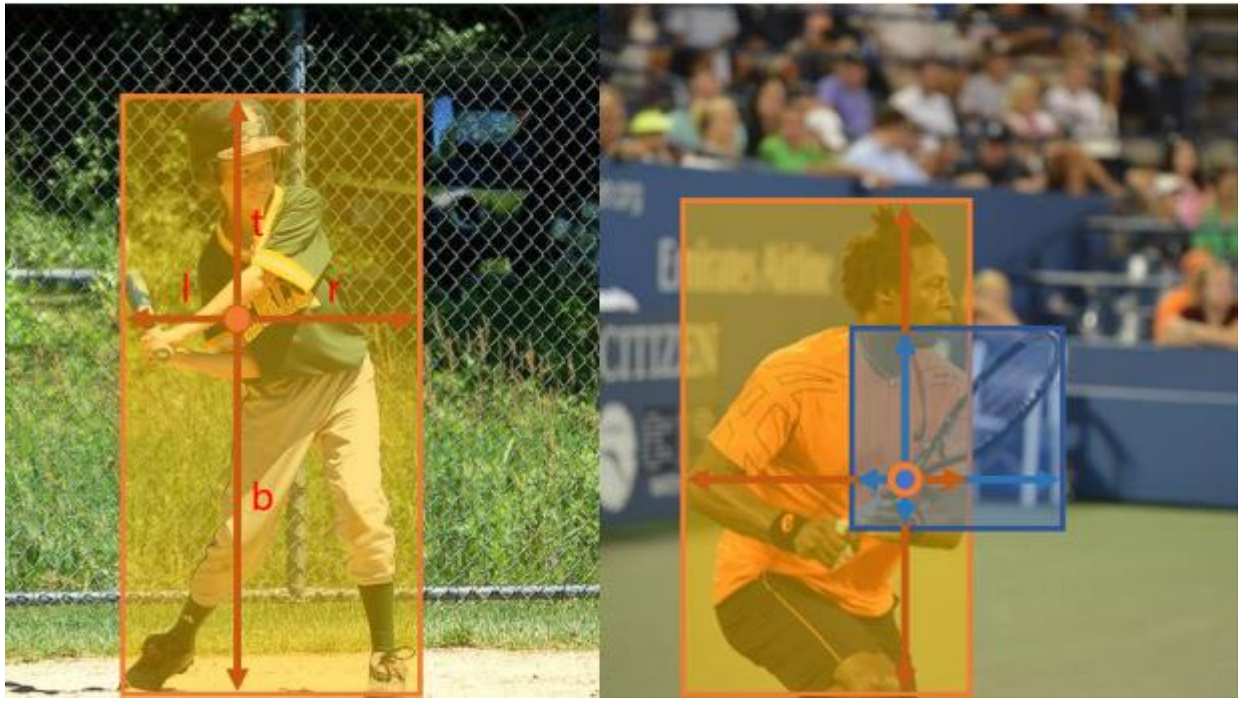
PolarMask


(a) Original image

(b) Pixel-wise Representation

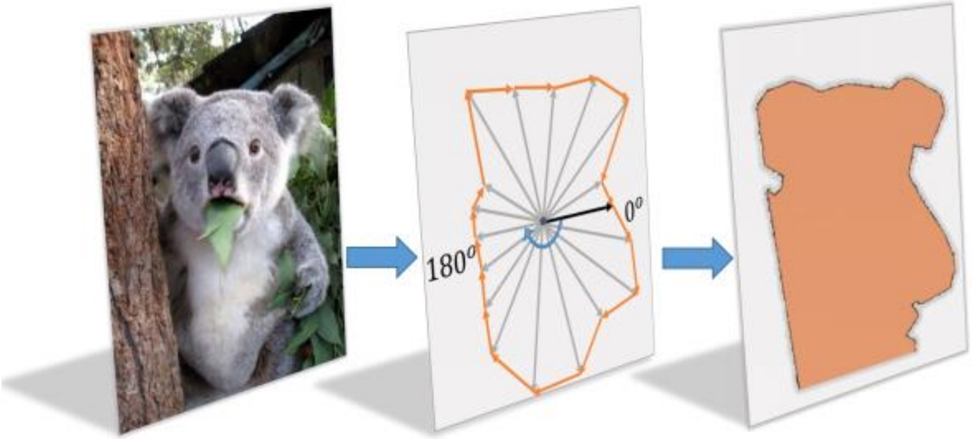
(c) Cartesian Representation

(d) Polar Representation



FCOS

PolarMask



- Polar Representation
  - pre-define angle interval, predict length of the ray
- Center Samples
  - (x, y), falls into areas around the mass-center, about 9∼16 pixels
  - Avoid imbalance ..., Mass-center may not be the best center
- Distance Regression
  - multiple intersection, one with the maximum length
  - does not have intersection, set to 1e-6
- Mask Assembling
  - Top-k, threshold, NMS
  - Calculate point position, connect them one by one

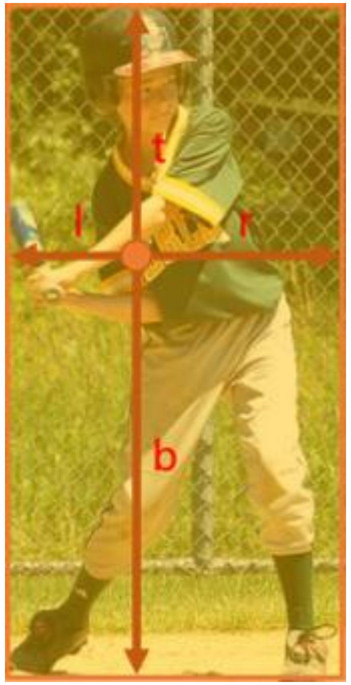
PolarMask Center-ness

- Poor performance,
  - low-quality bounding boxes, far away from the center of an object
- Center-ness branch
  - Single-layer branch

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

  - Range from 0 to 1, use BCE loss
- Polar Center-ness

$$\text{Polar Centerness} = \sqrt{\frac{\min(\{d_1, d_2, \ldots, d_n\})}{\max(\{d_1, d_2, \ldots, d_n\})}}$$



PolarMask Polar IoU Loss

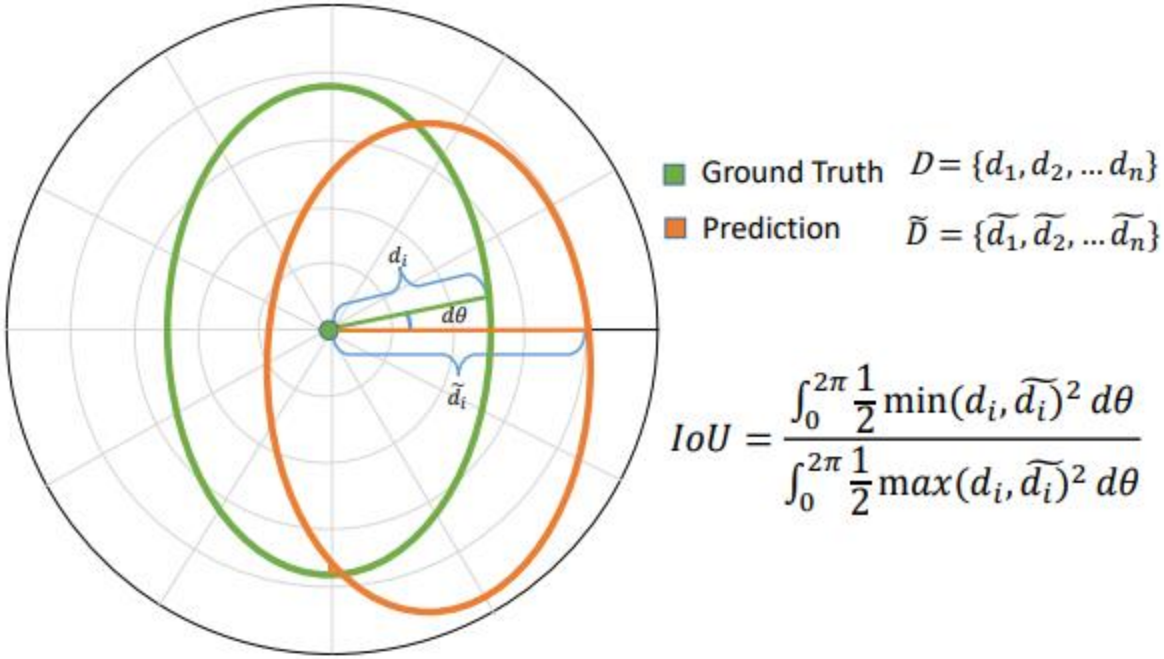
- Converts segmentation into regression

- Polar IoU

$$\text{Polar IoU} = \frac{\sum_{i=1}^{n} d_{\min}}{\sum_{i=1}^{n} d_{\max}}$$

- Polar IOU Loss

$$\text{Polar IoU Loss} = \log \frac{\sum_{i=1}^{n} d_{\max}}{\sum_{i=1}^{n} d_{\min}}$$



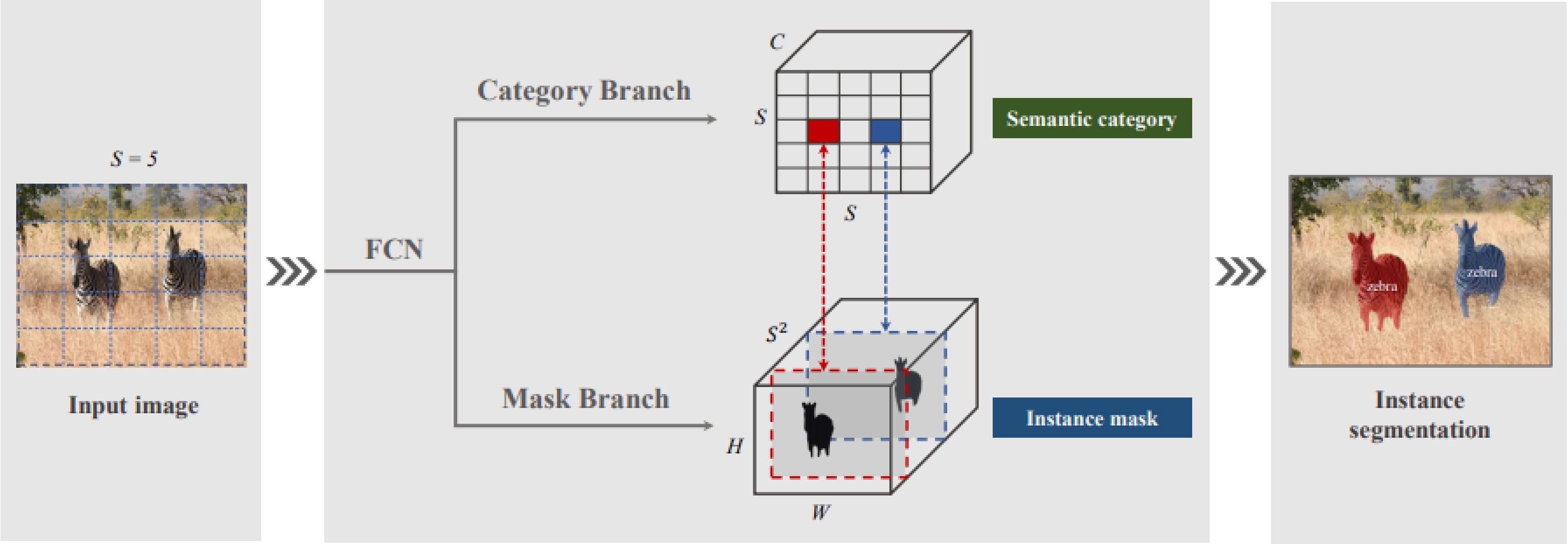
d_i
$d\theta$
$\tilde{d_i}$
🟩 Ground Truth $D = \{d_1, d_2, \dots d_n\}$
🟧 Prediction $\tilde{D} = \{\tilde{d_1}, \tilde{d_2}, \dots \tilde{d_n}\}$

$$IoU = \frac{\int_0^{2\pi} \frac{1}{2} \min(d_i, \tilde{d_i})^2 \, d\theta}{\int_0^{2\pi} \frac{1}{2} max(d_i, \tilde{d_i})^2 \, d\theta}$$

Figure 5 – Mask IoU in Polar Representation. Mask IoU (interaction area over union area) in the polar coordinate can be calculated by integrating the differential IoU area in terms of differential angles.

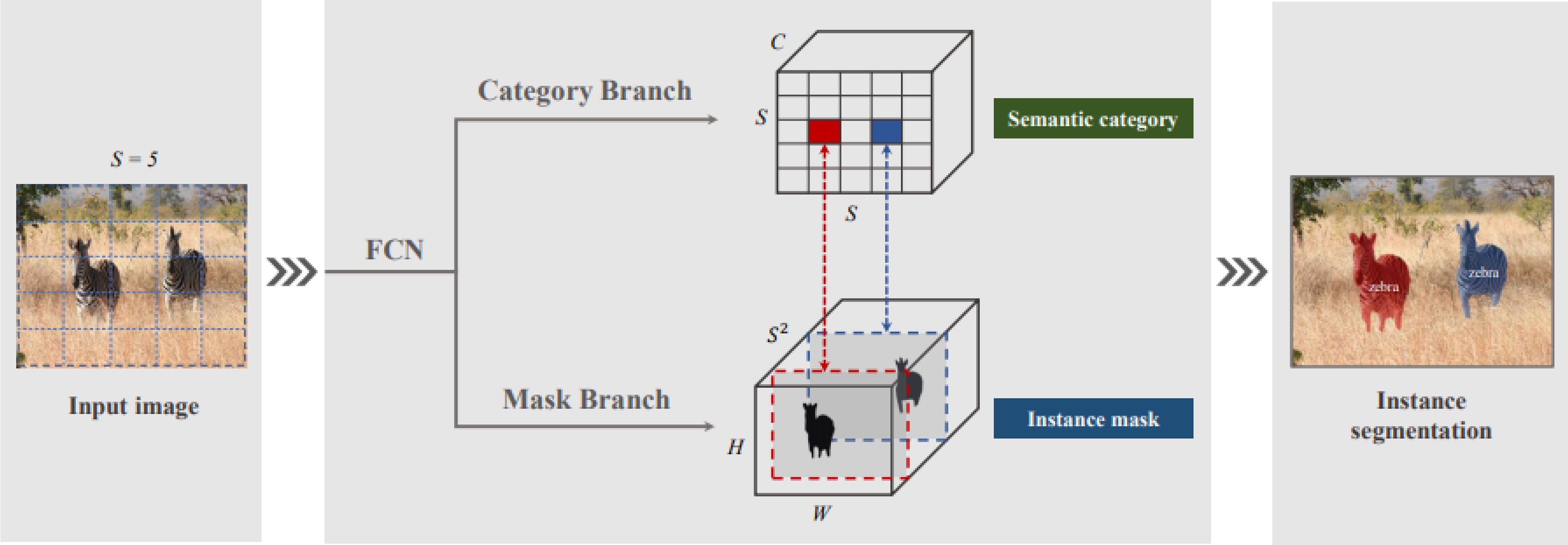
SOLO: Segmenting Objects by Locations

- Divides input image into a uniform grids (S×S
  - predicting the semantic category
  - segmenting that object instance



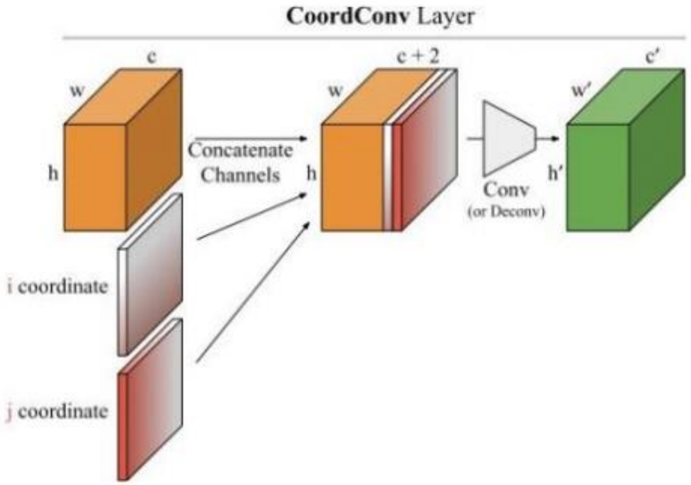
SOLO

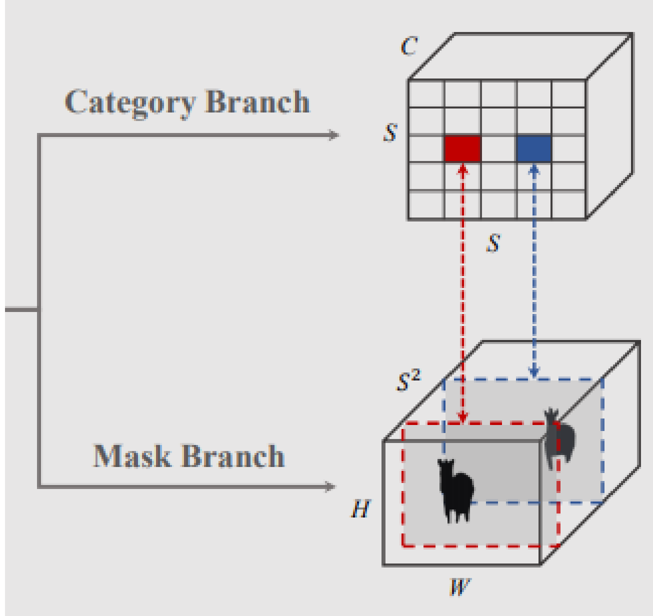
- Semantic Category
  - Output S x S x C, C classes



SOLO Instance Mask

- One-to-one correspondence
  - Channel k, grid (i, j), where k = i · S + j
- Spatially variant, position sensitive
  - CoordConv


CoordConv Layer
c
w
h
Concatenate Channels
i coordinate
j coordinate
c + 2
w
h
Conv (or Deconv)
c'
w'
h'


Category Branch
Mask Branch
C
S
S
S^2
H
W