YOLOX
Exceeding YOLO Series in 2021

Advanced Computer Vision Meetup

Performance

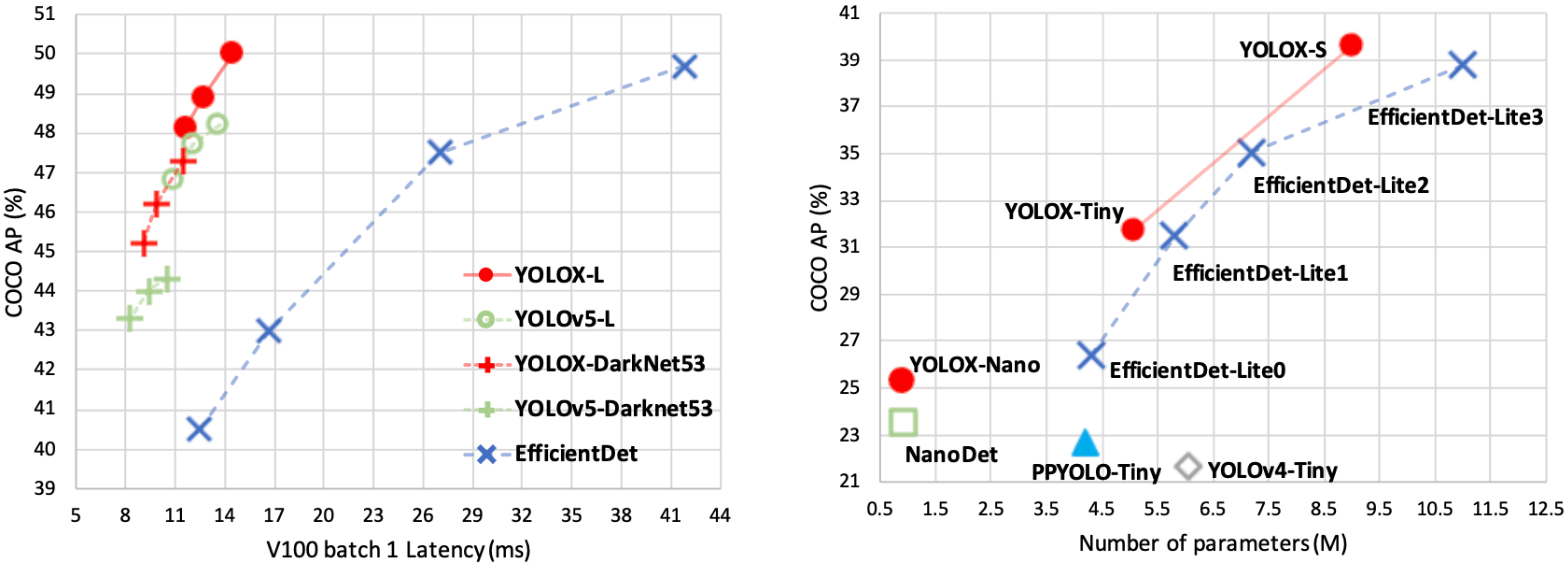
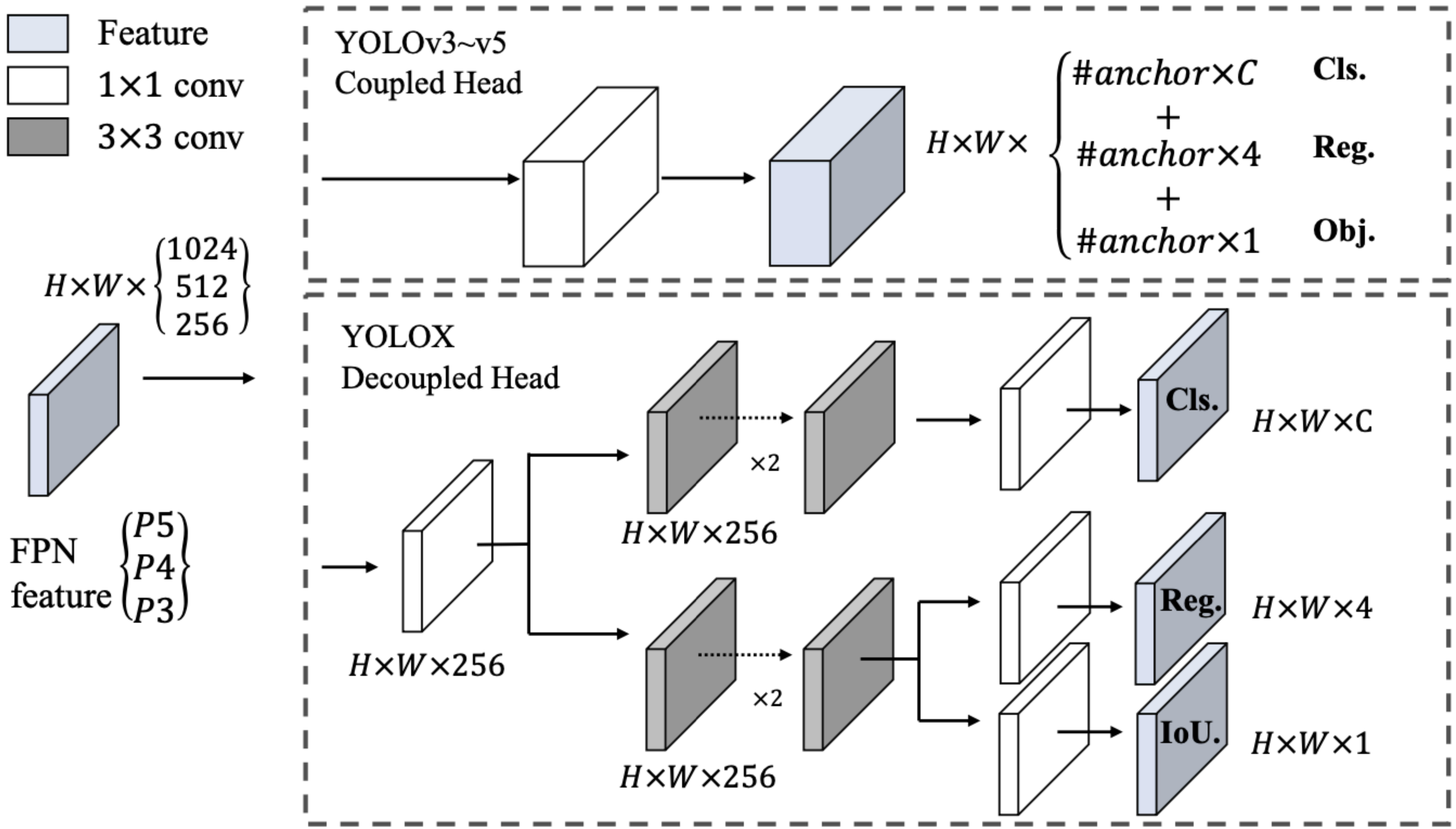


Figure 1: Speed-accuracy trade-off of accurate models (top) and Size-accuracy curve of lite models on mobile devices (bottom) for YOLOX and other state-of-the-art object detectors.

Arch



Feature
1×1 conv
3×3 conv
YOLOv3~v5
Coupled Head
H×W×
#anchor×C
+
#anchor×4
+
#anchor×1
Cls.
Reg.
Obj.
H×W×
1024
512
256
FPN
feature
P5
P4
P3
YOLOX
Decoupled Head
H×W×256
H×W×256
×2
×2
H×W×256
Cls.
Reg.
IoU.
H×W×C
H×W×4
H×W×1

Improvements

- Strong data augmentation
- Decoupled heads
- **End-to-end* (removing NMS)**
- **Anchor free**
- Multiple positives
- Optimal Transport Assignment
- 3x3 Center sampling
- IoU on regression

Strong data augmentation

(a) Crop, Rotation, Flip, Hue, Saturation, Exposure, Aspect.

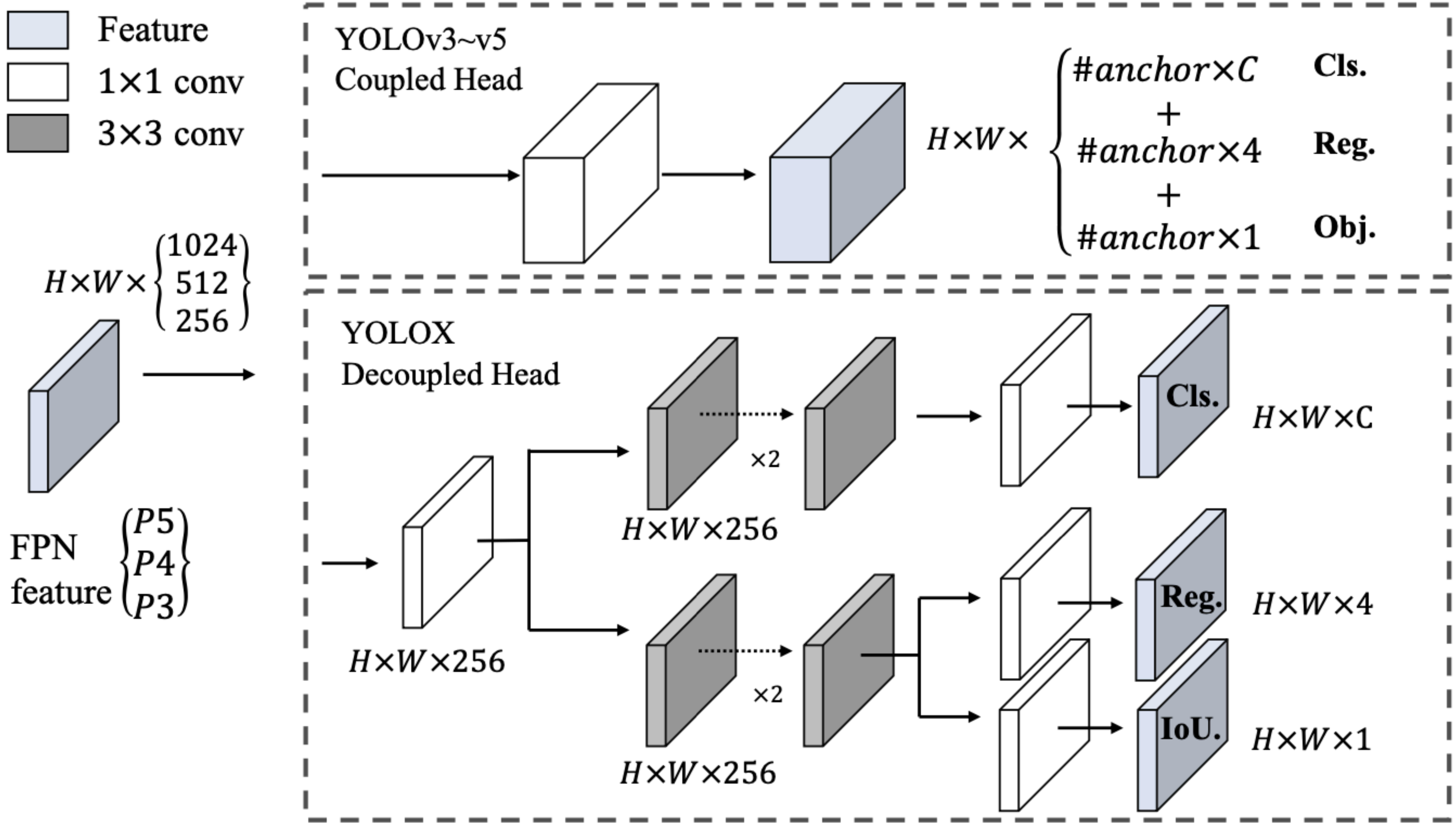
(b) MixUp

(c) CutMix

(d) Mosaic

(e) Blur

Decoupled heads



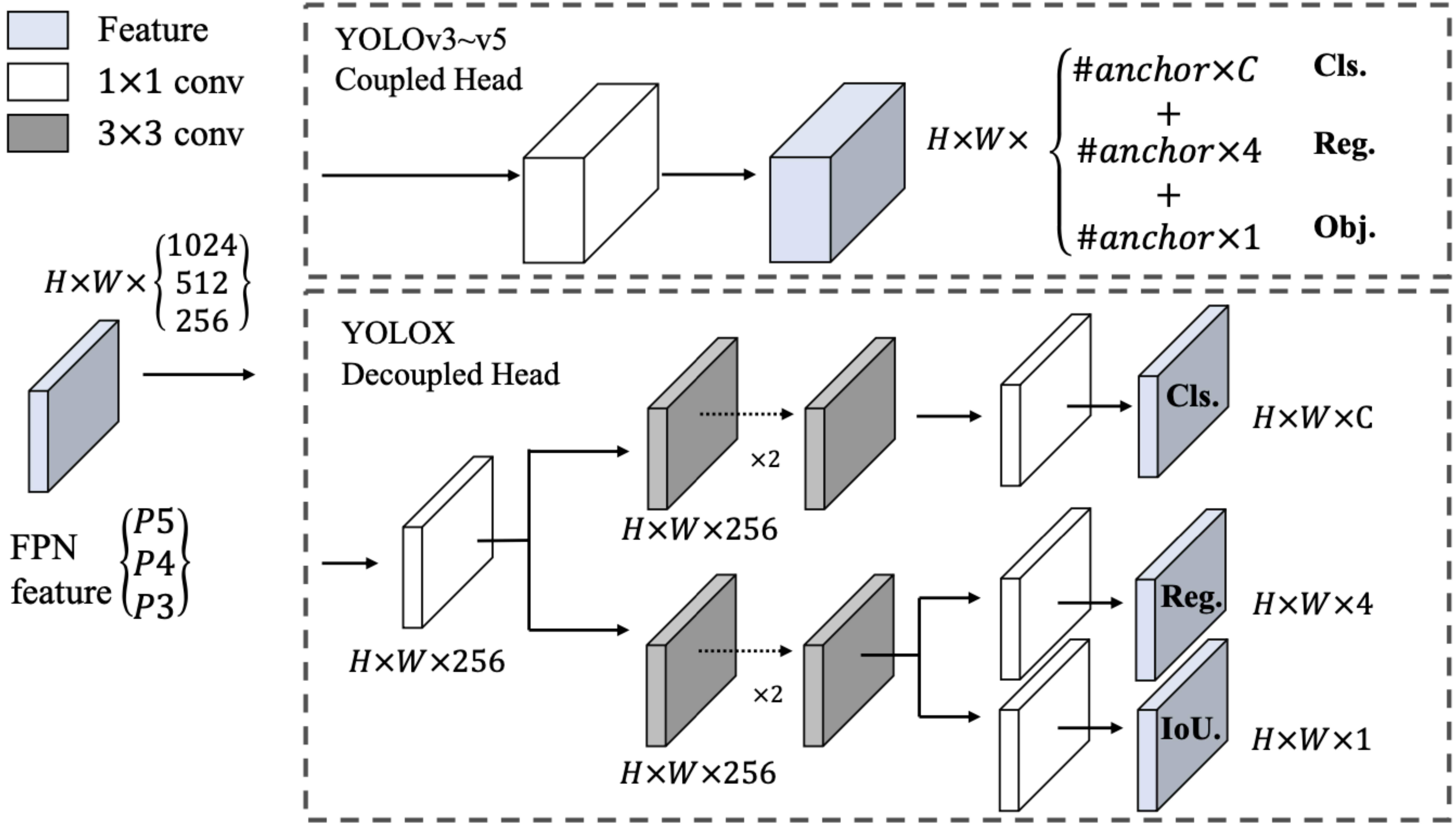
SimOTA

* $c_{ij} = L_{ij}^{cls} + \lambda L_{ij}^{reg}, \qquad (1)$

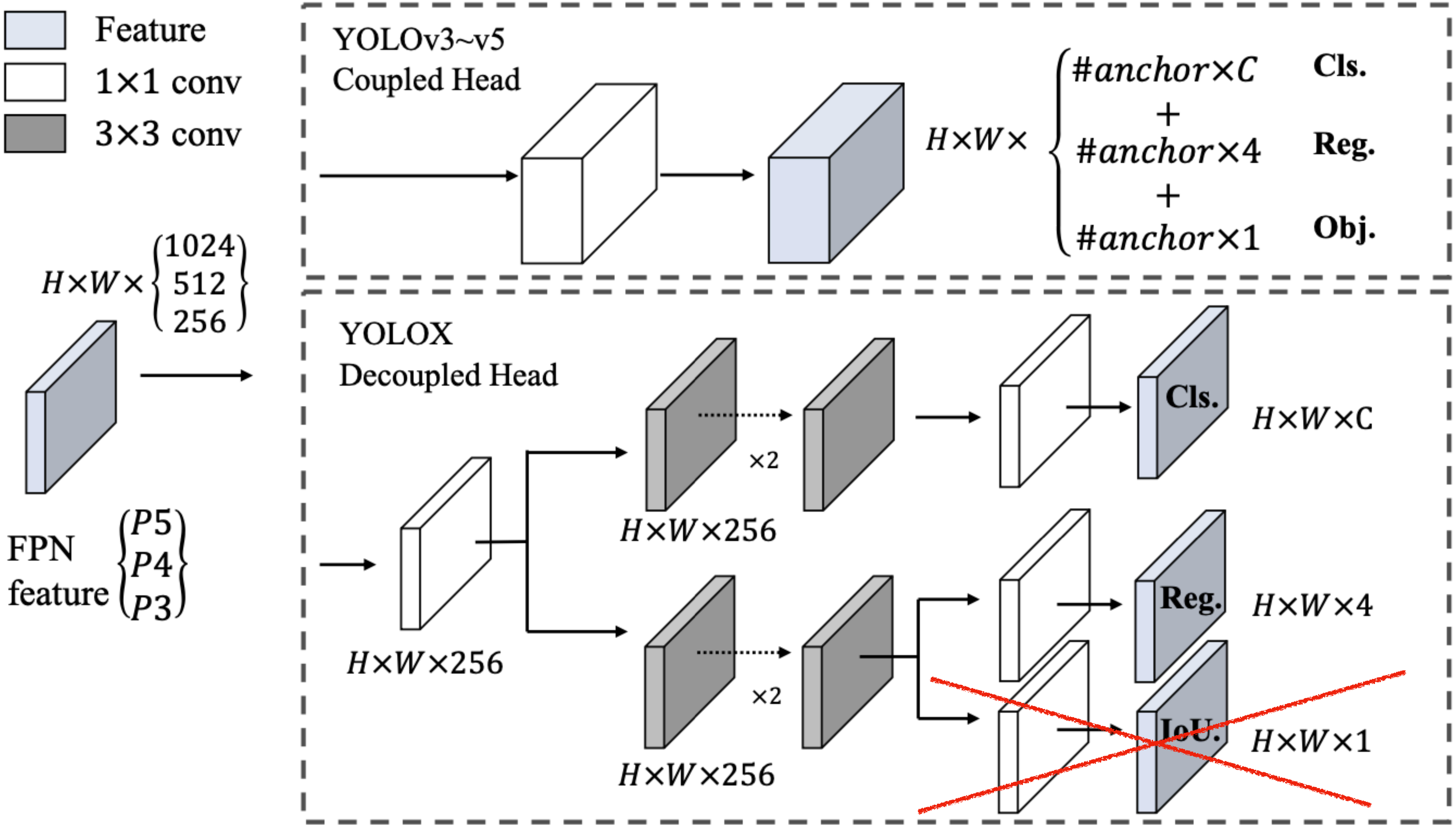
* 3x3 positive samples (center - sampling)

* Smallest area first

IoU Branch


Feature
1×1 conv
3×3 conv
YOLOv3~v5
Coupled Head
H×W×
#anchor×C
+
#anchor×4
+
#anchor×1
Cls.
Reg.
Obj.
H×W×
1024
512
256
FPN
feature
P5
P4
P3
YOLOX
Decoupled Head
H×W×256
H×W×256
×2
H×W×256
×2
Cls.
H×W×C
Reg.
H×W×4
IoU.
H×W×1

IoU Branch


Feature
1×1 conv
3×3 conv
YOLOv3~v5
Coupled Head
H×W×
#anchor×C
+
#anchor×4
+
#anchor×1
Cls.
Reg.
Obj.
H×W×
1024
512
256
FPN
feature
P5
P4
P3
YOLOX
Decoupled Head
H×W×256
×2
H×W×256
×2
H×W×256
Cls.
Reg.
IoU.
H×W×C
H×W×4
H×W×1

IoU Branch

Joker316701882 commented on 16 Aug 2021 Member ...

Yes, YOLOX uses obj branch. For IoU branch, it is merged into cls branch. In the following code, the cls target is multiplied by preded ious.

YOLOX/yolox/models/yolo_head.py
Lines 363 to 365 in `e1052df`

```
363        cls_target = F.one_hot(
364            gt_matched_classes.to(torch.int64), self.num_classes
365        ) * pred_ious_this_matching.unsqueeze(-1)
```

End-to-End*

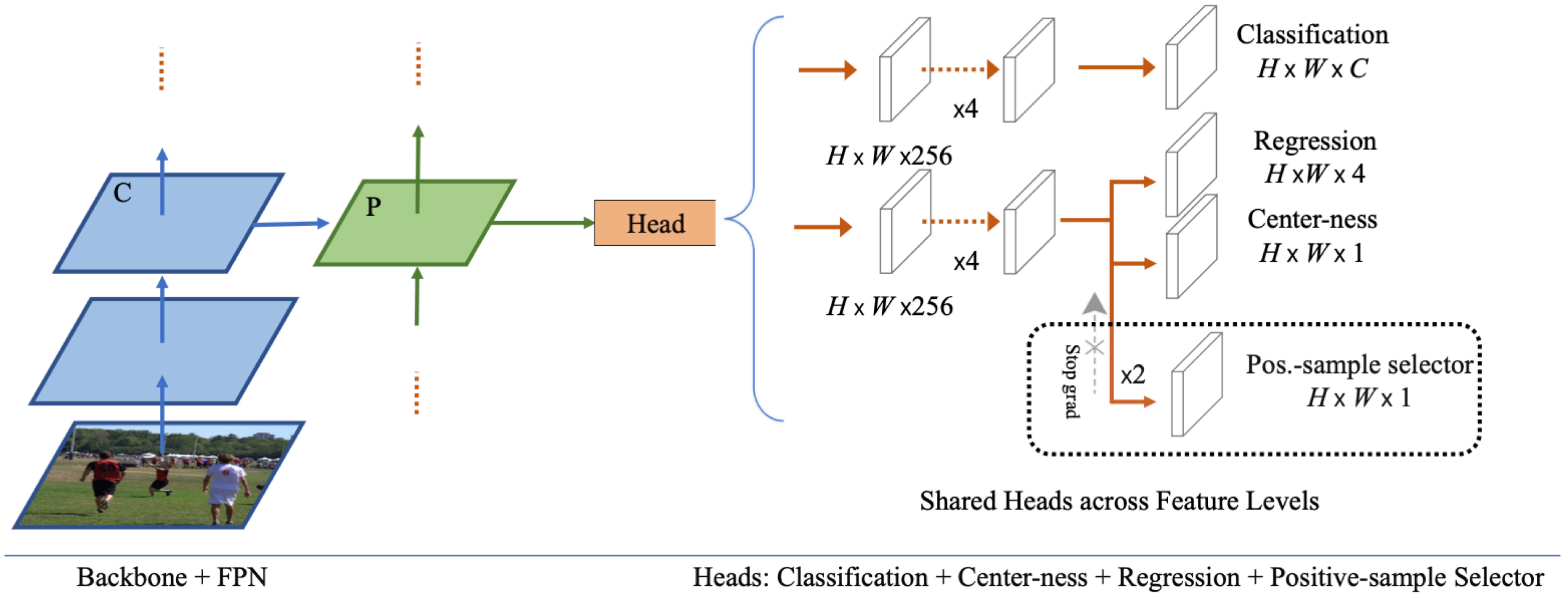


Figure 1 – The proposed detector FCOS_{PSS}, which is NMS free and end-to-end trainable. Compared to the original FCOS detector, the only modification to the network is the introduction of the 'positive sample selector (PSS)' as shown in the dashed box. Because the PSS head consists of only two compact conv. layers, the computation overhead is negligible (~8%). Here the 'Stop-grad' operation plays an important role in training (see details in the text §3.5).

* Drop of 0.8% in AP w.r.t. best model

End-to-End*

Loss contradiction contradiction

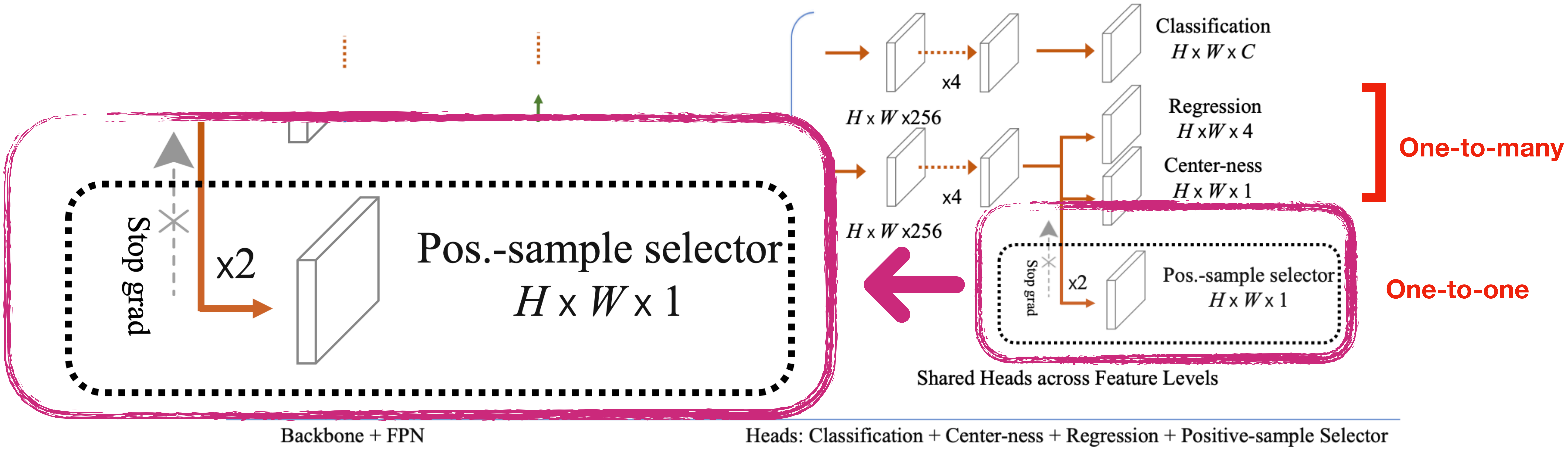


Figure 1 – The proposed detector FCOS$_{PSS}$, which is NMS free and end-to-end trainable. Compared to the original FCOS detector, the only modification to the network is the introduction of the 'positive sample selector (PSS)' as shown in the dashed box. Because the PSS head consists of only two compact conv. layers, the computation overhead is negligible (~8%). Here the 'Stop-grad' operation plays an important role in training (see details in the text §3.5).

* Drop of 0.8% in AP w.r.t. best model

Centre-ness Branch

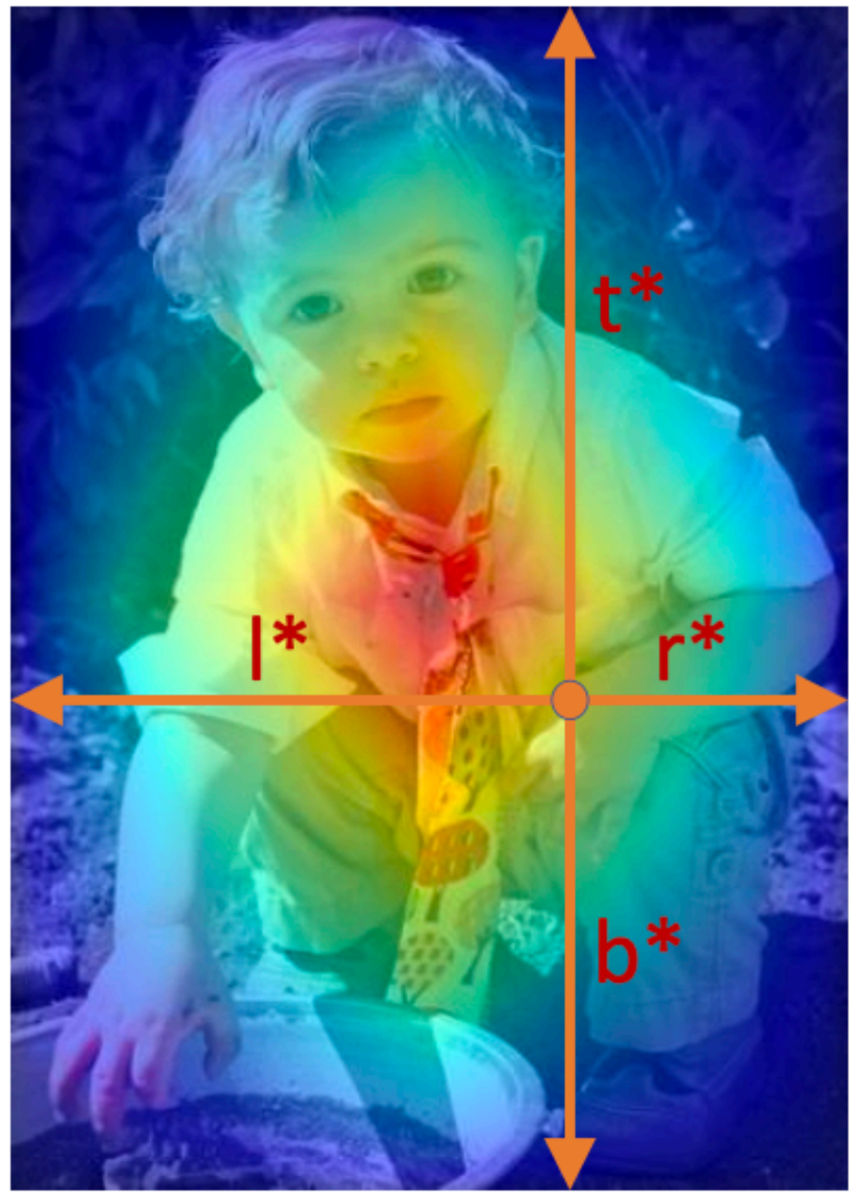


Figure 3 – Center-ness. Red, blue, and other colors denote 1, 0 and the values between them, respectively. Center-ness is computed by Eq. (3) and decays from 1 to 0 as the location deviates from the center of the object. When testing, the center-ness predicted by the network is multiplied with the classification score thus can down-weight the low-quality bounding boxes predicted by a location far from the center of an object.

Centre-ness Effect

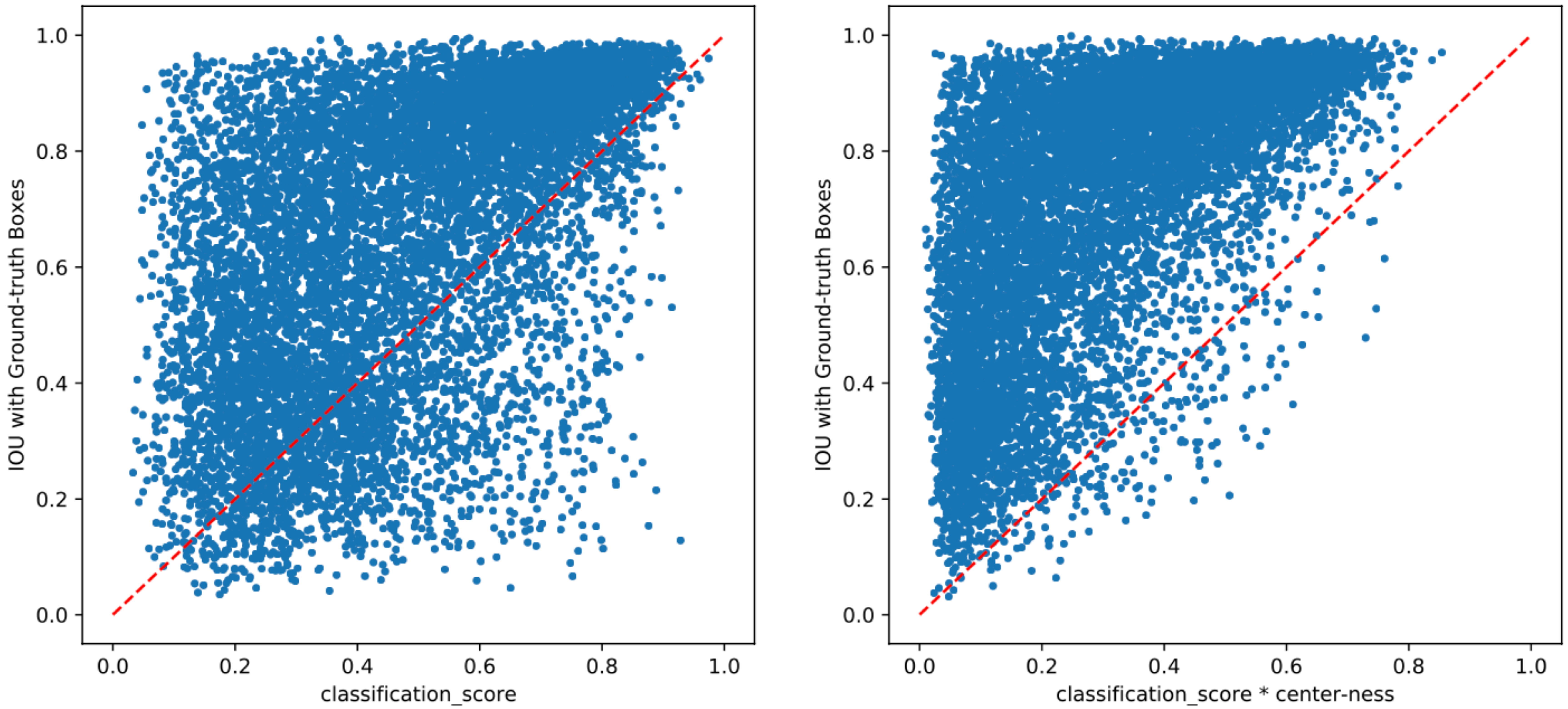


Figure 7 – Without (left) or with (right) the proposed center-ness. A point in the figure denotes a detected bounding box. The dashed line is the line $y = x$. As shown in the figure (right), after multiplying the classification scores with the center-ness scores, the low-quality boxes (under the line $y = x$) are pushed to the left side of the plot. It suggests that the scores of these boxes are reduced substantially.

FCOS

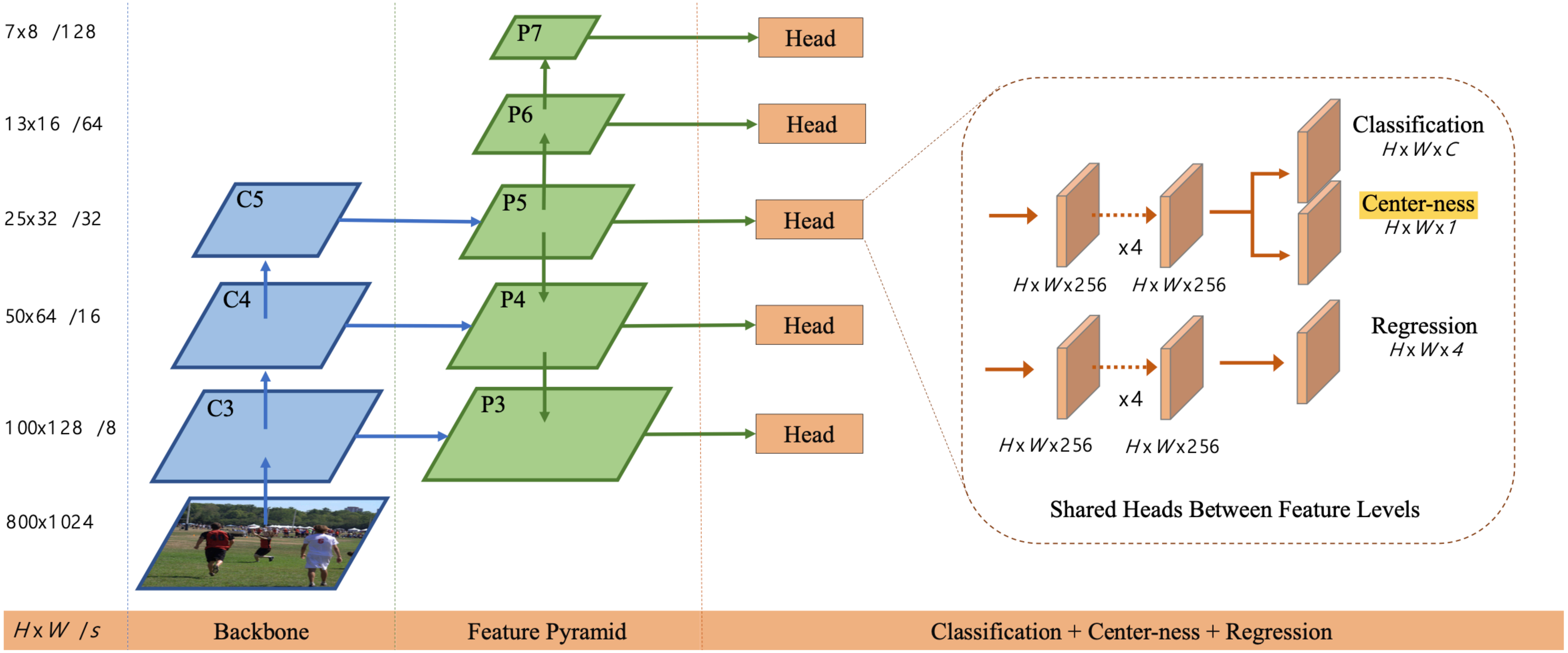
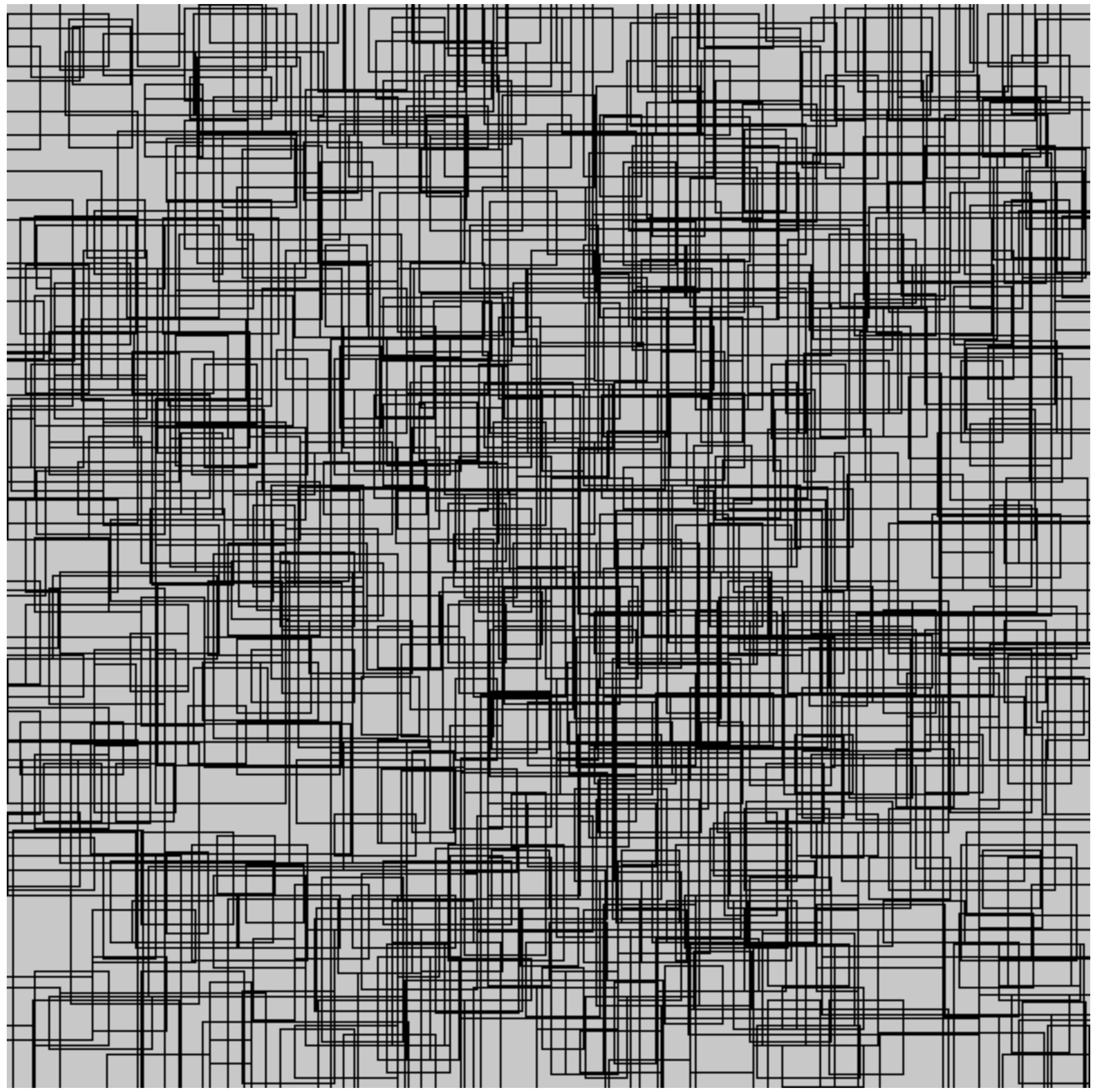


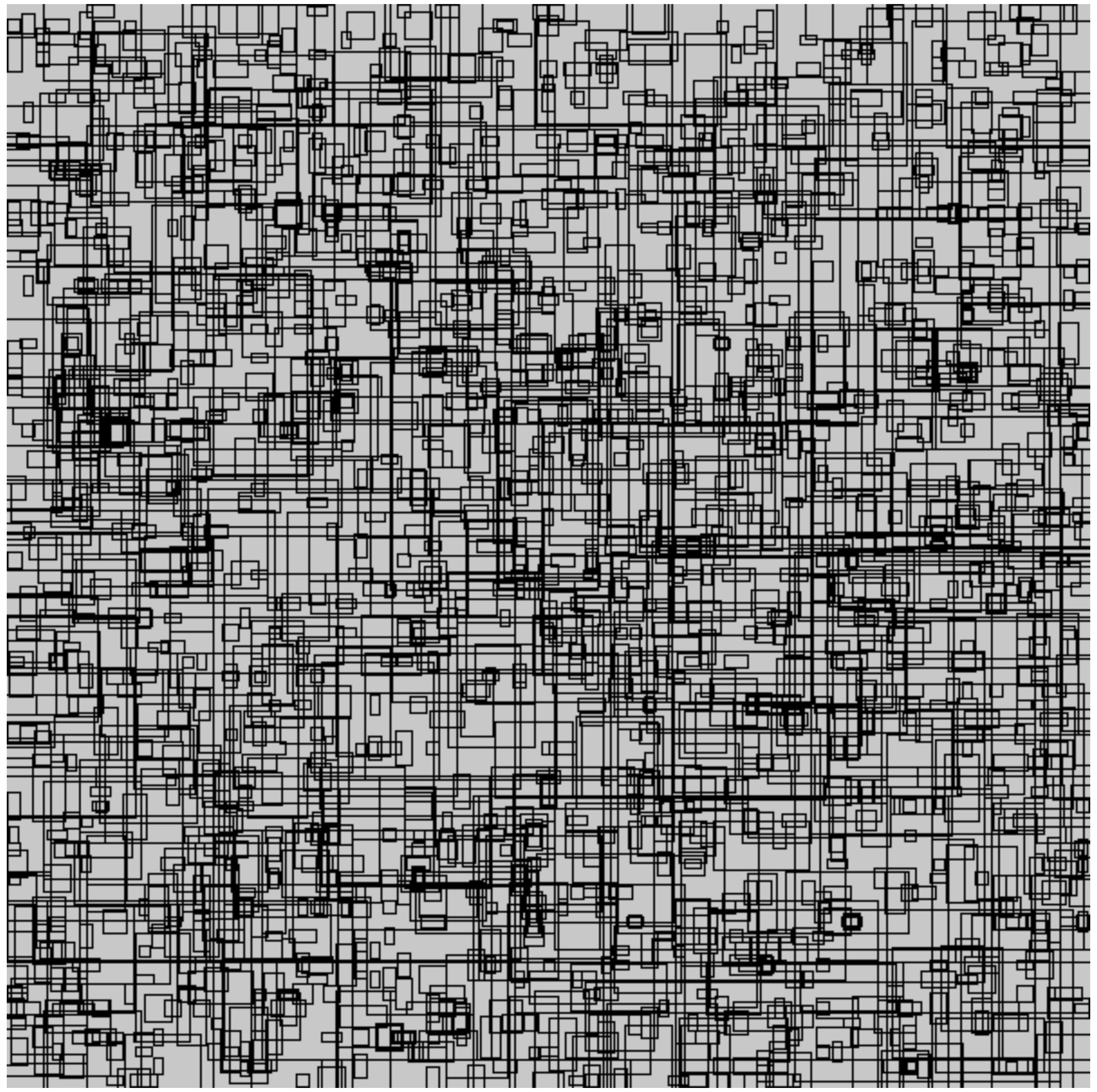
Figure 2 – The network architecture of FCOS, where C3, C4, and C5 denote the feature maps of the backbone network and P3 to P7 are the feature levels used for the final prediction. $H \times W$ is the height and width of feature maps. '/s' ($s = 8, 16, ..., 128$) is the down-sampling ratio of the feature maps at the level to the input image. As an example, all the numbers are computed with an 800×1024 input.

Anchor free

Anchors for RetinaNet (1% of total)



Normal objects



Smaller objects

Anchor free

Some issues with anchors

- Extra hyperparameters
- Specialised in the training data distribution
- Increases complexity of the model (number of params and arch)
- Issues with small objects

Anchor free

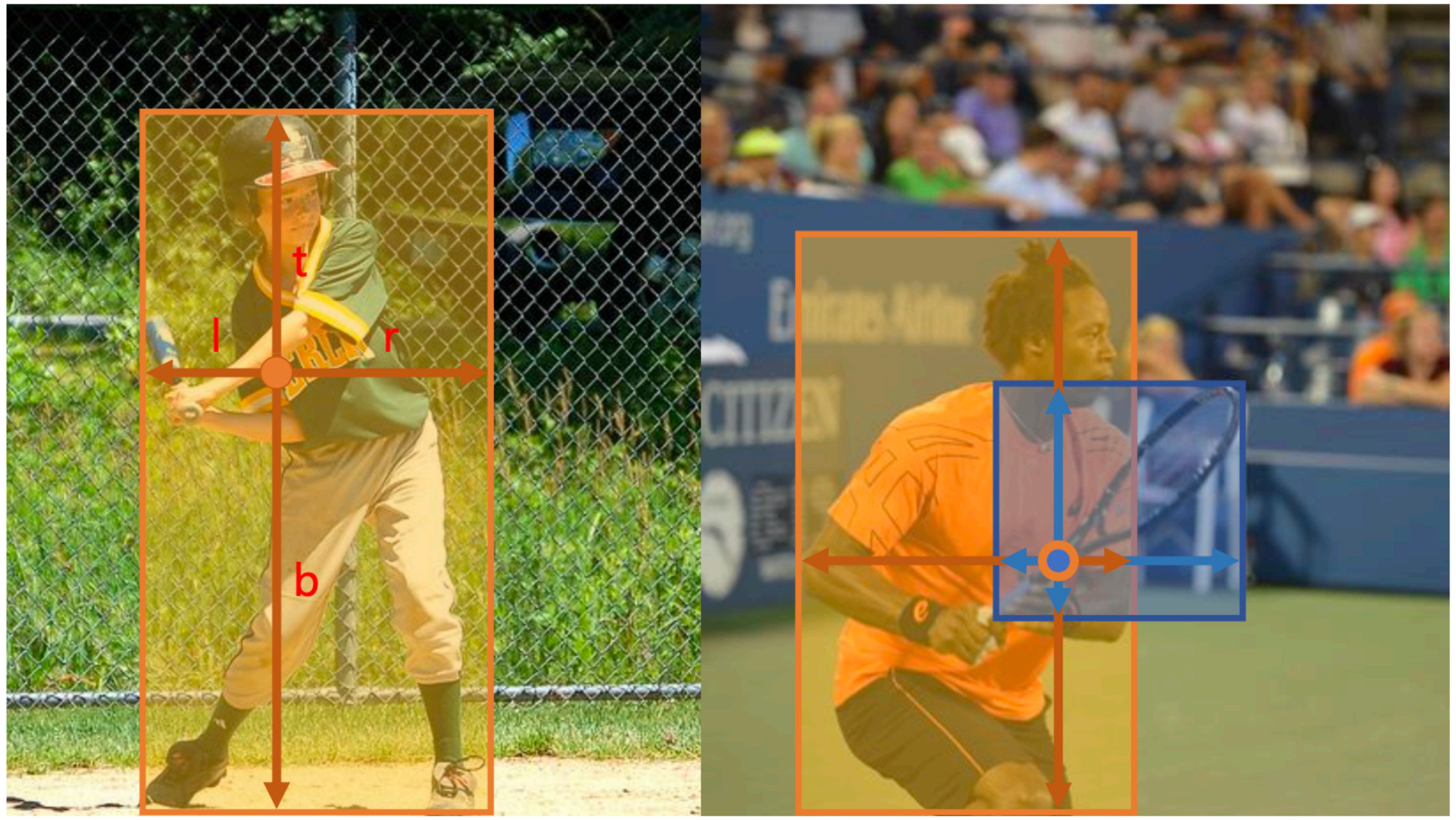


Figure 1 – As shown in the left image, FCOS works by predicting a 4D vector (l, t, r, b) encoding the location of a bounding box at each foreground pixel (supervised by ground-truth bounding box information during training). The right plot shows that when a location residing in multiple bounding boxes, it can be ambiguous in terms of which bounding box this location should regress.