Appendix

A. Further Illustration of Position-aware Dynamic Convs

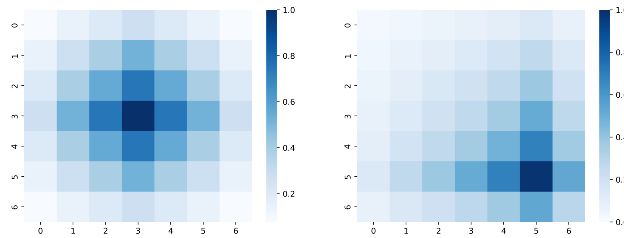
We have described the bounding box and centerness-based positional encoding in the paper. Here, we provide the pseudo-code of dynamic convs with the two kinds of PE in Algorithm 1.

Algorithm 1 Position-aware Dynamic Convs (PyTorch)

```
# q: proposal featrues, (N, c)
# r: roi features, (N, 7 * 7, c)
# b: proposal boxes(x, y, w, h), (N, 4)
# m: centerness, (7 * 7)
# pe_i: PE of image coordinates, (N, 7 * 7, c)

def PositionAwareDynamicConvs(q, r, b, pe_i, m):
   # kernels of two 1x1 conv layers
   params = dynamic_layer(q)
   k = params[:, :c*d].view(N, c, d)
   v = params[:, c*d:].view(N, d, c)

   # encode box center (x, y) into PE vector
   center, w, h = b[:, :2], b[:, 2], b[:, 3]
   qc = mlp_c(q) # center transformation
   qs = mlp_s(q) # w_ref, h_ref
   pe = sinusoidal(center)
   pe = pe * qc
   pe[:, :c/2] = qs[:, 0] / w * pe[:, :c/2]
   pe[:, c/2:] = qs[:, 1] / h * pe[:, c/2:]

   # modulate r and PE by centerness
   pe_i = pe_i * m.flatten()[None, :, None]
   pe = pe.view(N, 1, c) * m.flatten()[None, :,
       None]

   # concate (r, pe_img) and (k, pe)
   r = torch.cat([r, pe_i], dim=-1)
   k = k.unsqueeze(1).repeat(1, 7*7, 1, 1)
   pe = pe.unsqueeze(-1).repeat(1, 1, 1, d)
   k = torch.cat([k, pe], dim=2)

   # interaction between r and q
   r = relu(norm(torch.matmul(r.unsqueeze(-2), k)
       .squeeze(-2)))
   r = relu(norm(bmm(r, v)))

   # reduce spatial dimension to obtain object
       features o
   r = r.flatten(1)
   o = out_layer(r)

   return o
```

Two conv kernels are generated from the proposal feature first. Then we map the proposal feature to two vectors in geometry space. One of them is $q_c \in \mathbb{R}^c$, the another is $q_s \in \mathbb{R}^2$, consisting of two scaler w_{ref} and h_{ref}. Then they work together to modulate the sinusoidal embedding from box center (x, y) as Eq.(5).

The centerness-based PE is built on the bounding box PE to vary local positions within a proposal box. We also list it in Algorithm 1. The single channel mask is generated as Eq.(6), and it is shown in Fig. 1(a). It is the same for all proposal boxes. We try two strategies to enhance it. First, we make the centerness-based mask trainable, initialized with Fig. 1(a). Second, as different objects have different shapes and semantic centers, we predict the semantic center coordinate. The largest value of centerness mask is at the predicted semantic center as Fig. 1(b). However, the static one in Fig. 1(a) gives the best results.



(a) Static (b) Adjust

Figure 1. Different forms of centerness-based PE.

B. Effect of Number of Stages

We provide the effectiveness of the number of stages on Sparse R-CNN and RecursiveDet in Fig.4 in the paper. It shows our model gains better results on any number of stages. We further present the comparisons on AdaMixer and DiffusionDet in Fig. 2 and Fig. 3. Except for the first stage in DiffusionDet, ReursiveDet behaves better than its counterpart.

Fig. 4 gives the visualization of predicted boxes from cascade structure Sparse R-CNN and recursive structure RecursiveDet. It shows that both methods detect objects progressively, and the recursive structure even gets a better final result than Sparse R-CNN. Similar phenomenon is illustrated in Fig. 5 and Fig. 6.

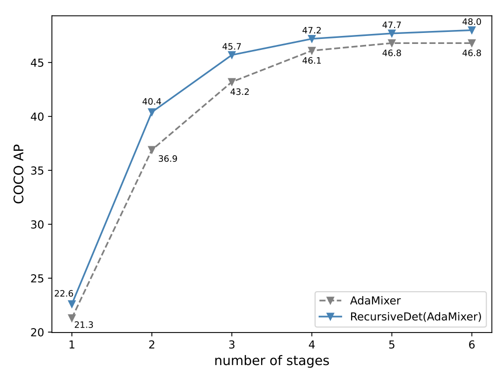


Figure 2. Effect of the number of stages in AdaMixer and RecursiveDet.

C. Results on CrowdHuman

CrowdHuman [7] dataset is a highly crowd pedestrian benchmark with only one class. There are about 23 persons per image on average and many overlaps. We only use

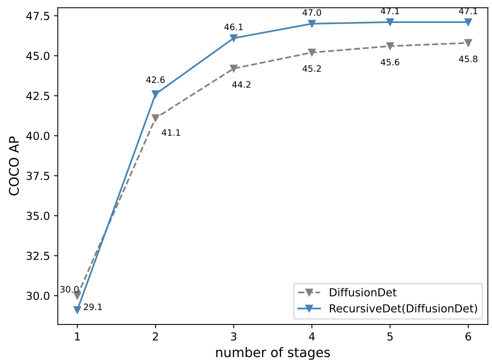


Figure 3. Effect of the number of stages in DiffusionDet and RecursiveDet.

the full-body bounding box to train models. The standard Caltech[2] evaluation metric, Average Precision (AP) and Recall are reported. We train the model for 50 epochs, and dividing the learning rate by 10 at 40th epoch. The results are listed in Tab. 1. It shows that Sparse R-CNN is already better than other well established detectors, like RetinaNet and AdaptiveNMS. It also behaves better than DETR-series. Our enhanced version achieves 91.9 AP, gains 1.1 AP from Sparse R-CNN. The other two metric mMR and Recall are also better than it.

Method	Queries	NMS	AP ↑	mMR ↓	Recall ↑
Faster R-CNN [6]	-	✓	85.0	50.4	90.2
RetinaNet [4]	-	✓	81.7	57.6	88.6
FCOS [9]	-	✓	86.1	55.2	94.3
AdaptiveNMS [5]	-	✓	84.7	49.7	91.3
DETR [1]	100	○	66.1	80.6	-
Deformable DETR [10]	400	○	86.7	54.0	92.5
PETR [3]	1000	○	89.5	45.6	94.0
Sparse R-CNN[8]	500	○	90.7	44.7	81.4
RecursiveDet	500	○	**91.8**	**43.3**	**96.7**

Table 1. Performance comparisons with other detectors on CrowdHuman. Our RecursiveDet is built on Sparse R-CNN.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 213–229. Springer, 2020.

[2] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE transactions on pattern analysis and machine intelligence*, 34(4):743–761, 2011.

[3] Matthieu Lin, Chuming Li, Xingyuan Bu, Ming Sun, Chen Lin, Junjie Yan, Wanli Ouyang, and Zhidong Deng. Detr for crowd pedestrian detection. *arXiv preprint arXiv:2012.06785*, 2020.

[4] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[5] Songtao Liu, Di Huang, and Yunhong Wang. Adaptive nms: Refining pedestrian detection in a crowd. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6459–6468, 2019.

[6] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[7] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

[8] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14454–14463, 2021.

[9] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019.

[10] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

Figure 4. Predictions of each stage in Sparse R-CNN and RecursiveDet. Boxes with classification score above 0.5 are showed.

Figure 5. Predictions of each stage in Sparse R-CNN and RecursiveDet. Boxes with classification score above 0.5 are showed.

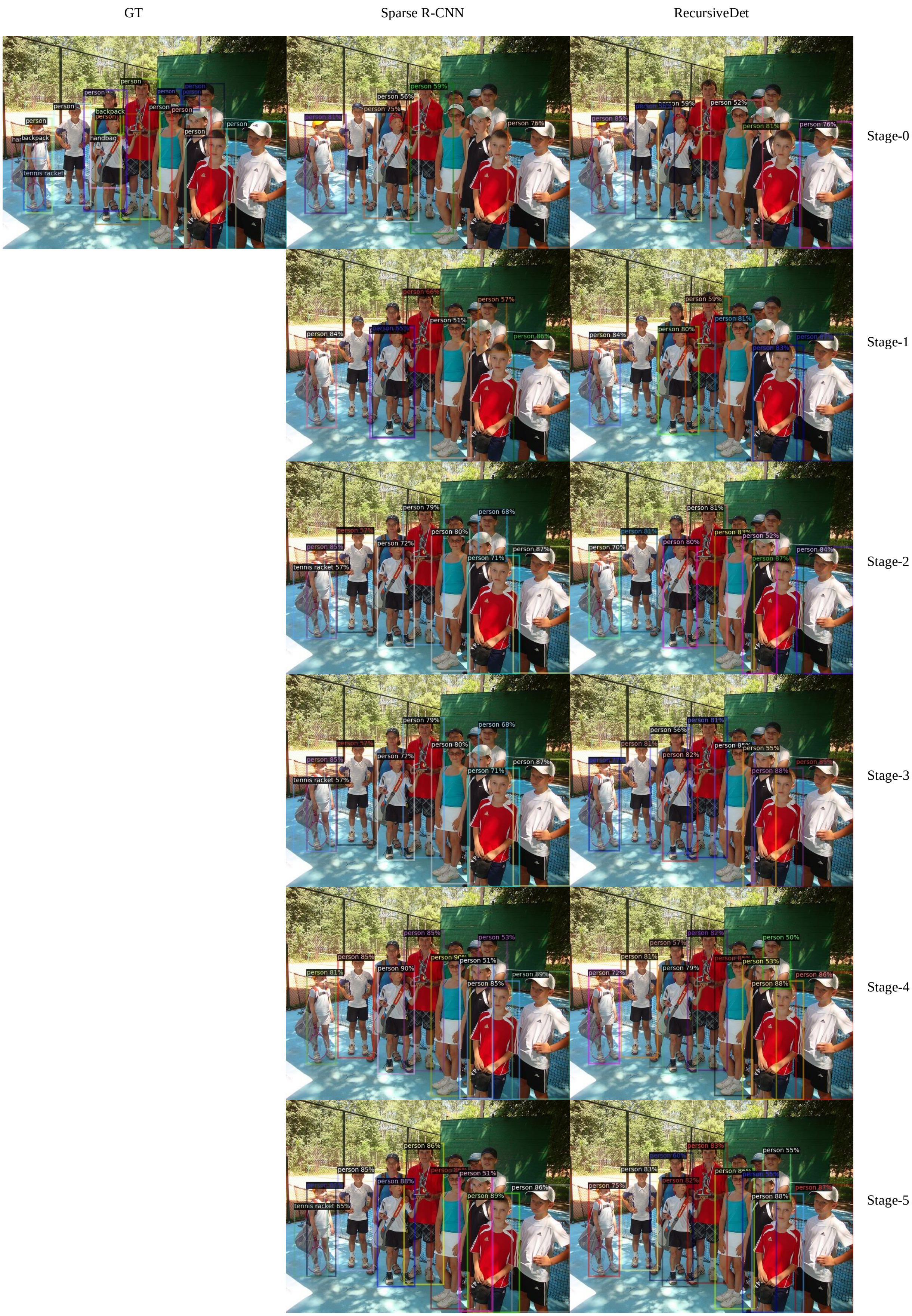


Figure 6. Predictions of each stage in Sparse R-CNN and RecursiveDet. Boxes with classification score above 0.5 are showed.