Appendix for AABO: Adaptive Anchor Box Optimization for Object Detection via Bayesian Sub-sampling

Optimal FPN Hyper-parameters in Preliminary Analysis

In preliminary analysis, we search for better RPN head architecture in FPN [6] as well as anchor settings simultaneously, to compare the respective contributions of changing RPN head architecture and anchor settings. The optimal hyper-parameters are shown in Table 1. Since the search space for RPN head architecture and anchor settings are both relatively small, the performance increase is not very significant.

Table 1. The optimal hyper-parameters of FPN [6] on COCO determined by BOHB [3] in preliminary analysis. Both RPN head architecture and anchor boxes are different from default settings in FPN. Note that the anchor scales here are the basic anchor scales without multiplying the strides of the feature maps.

Hyper-parameter	Conv Layers	Kernel Size	Dilation Size
Optimal Configuration	2	5x5	1
Hyper-parameter	Location of ReLU	Anchor Scales	Anchor Ratios
Optimal Configuration	Behind 5x5 Conv	2.5, 5.7, 13.4	2:5, 1:2, 1, 2:1, 5:2

Optimal Anchor Configurations

In this section, we record the best configurations searched out by AABO and analyze the difference between the results of default Faster-RCNN [8] and optimized Faster-RCNN.

As we design an adaptive feature-map-wised search space for anchor optimization, anchor configurations distribute variously in different layers of FPN [6]. Table 2 shows the optimal anchor configurations in FPN for COCO [7] dataset. We can observe that anchor scales and anchor ratios are larger and more diverse in shallower layers of FPN, while anchors tend to be smaller and more square in deeper layers.

Table 2. The optimal anchor configurations of FPN [6] for COCO [7] searched out by AABO. There are different anchor boxes in different layers of FPN.

FPN-Layer	Anchor Number	Anchor Scales	Anchor Ratios
Layer-1	9	{5.2, 6.1, 3.4, 4.9, 5.8, 4.8, 14.6, 7.4, 10.3}	{6.0, 0.3, 0.5, 1.6, 1.7, 2.6, 0.5, 0.5, 0.6}
Layer-2	6	{11.0, 7.6, 11.8, 4.7, 5.7, 3.8}	{0.2, 2.0, 2.3, 2.8, 1.0, 0.5}
Layer-3	7	{10.5, 11.1, 8.5, 6.7, 12.4, 4.6, 3.5}	{0.4, 4.2, 2.2, 1.6, 2.3, 0.5, 1.1}
Layer-4	6	{5.7, 8.5, 7.0, 11.2, 15.2, 15.5}	{0.3, 0.4, 0.8, 0.7, 2.9, 2.8}
Layer-5	5	{5.0, 4.2, 14.0, 10.1, 7.8}	{1.1, 1.4, 0.8, 0.7, 2.5}

Improvements on SOTA Detectors over COCO Test-Dev

After searching out optimal anchor configurations via AABO, we apply them on several SOTA detectors to study the generalization property of the anchor settings. In this section, we report the performance of these optimized detectors on COCO *test-dev* split. The results are shown in Table 3.

It can be seen that the optimal anchors can consistently boost the performance of SOTA detectors on both COCO *val* split and COCO *test-dev*. Actually, the mAP of the optimized detectors on *test-dev* is even higher than the mAP on *val*, which illustrates that the optimized anchor settings could bring consistent performance improvements on *val* split and *test-dev*.

Table 3. Benefit of the optimal anchor settings on some SOTA methods evaluated on both COCO *val* and *test-dev*. Here HTC* means 2x training of HTC. The results indicate that the optimal anchors can consistently boost the performance of SOTA detectors, whether on COCO *val* or *test-dev*.

Model	Anchor Setting	Eval on	mAP
Mask RCNN[4] w r101	Default	*val*	40.3
	Searched via AABO	*val*	$42.3^{+2.0}$
	Searched via AABO	*test-dev*	$\mathbf{42.6}^{+2.3}$
DCNv2[9] w x101	Default	*val*	43.4
	Searched via AABO	*val*	$45.8^{+2.4}$
	Searched via AABO	*test-dev*	$\mathbf{46.1}^{+2.7}$
Cascade Mask RCNN[1] w x101	Default	*val*	44.3
	Searched via AABO	*val*	$46.8^{+2.5}$
	Searched via AABO	*test-dev*	$\mathbf{47.2}^{+2.9}$
HTC*[2] w x101	Default	*val*	47.5
	Searched via AABO	*val*	$50.1^{+2.6}$
	Searched via AABO	*test-dev*	$\mathbf{50.6}^{+3.1}$

Additional Qualitative Results

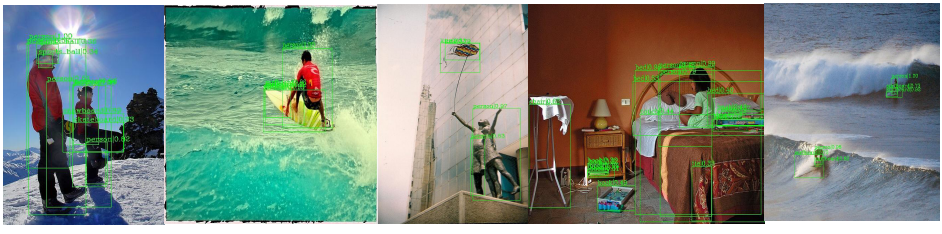
In this section, Figure 1 and Figure 2 give some qualitative result comparisons of Faster-RCNN [8] with default anchors and optimal anchors.

As illustrated in Figure 1, more larger and smaller objects can be detected using our optimal anchors, which demonstrates that our optimal anchors are more diverse and suitable for a certain dataset. And there are some other differences shown in Figure 2: Using optimal anchor settings, the predictions of Faster-RCNN are usually tighter, more precise and concise. While the predictions are more inaccurate and messy when using pre-defined anchors. Specifically, there exist many bounding boxes in a certain position, which usually are different parts of one same object and overlap a lot.

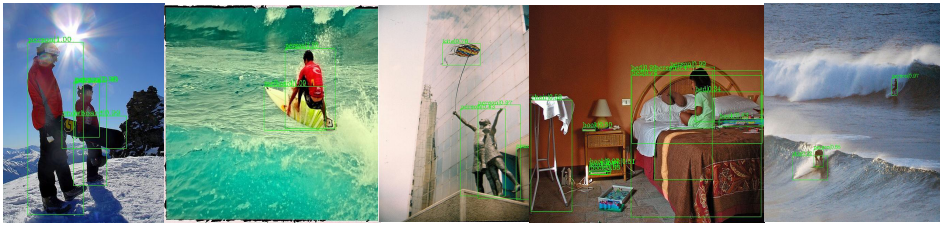
(a) Faster-RCNN with Default Anchors

(b) Faster-RCNN with Optimized Anchors

Fig. 1. Some qualitative result comparison on COCO [7] dataset. Using optimized anchor configurations, more large and small objects are detected. We use ResNet-50 [5] as backbones.



(a) Faster-RCNN with Default Anchors



(b) Faster-RCNN with Optimized Anchors

Fig. 2. Some qualitative result comparisons on COCO [7] dataset. The bounding boxes given by Faster-RCNN [8] with optimized anchor configurations are much tighter and clearer, while bounding boxes given by default Faster-RCNN are more messy and overlap a lot. We use ResNet-50 [5] as backbones.

References

1. Cai, Z., Vasconcelos, N.: Cascade r-cnn: High quality object detection and instance segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2019)
2. Chen, K., Pang, J., Wang, J., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: Hybrid task cascade for instance segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (2019)
3. Falkner, S., Klein, A., Hutter, F.: Bohb: Robust and efficient hyperparameter optimization at scale. arXiv preprint arXiv:1807.01774 (2018)
4. He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask r-cnn. In: 2017 IEEE International Conference on Computer Vision (ICCV) (2017)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
6. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)
7. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
8. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NIPS (2015)
9. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable convnets v2: More deformable, better results. arXiv preprint arXiv:1811.11168 (2018)