MULTI-SCALE REINFORCEMENT LEARNING STRATEGY FOR OBJECT DETECTION

*Yihao Luo, Xiang Cao, Juntao Zhang, Leixilan Pan, Tianjiang Wang, Qi Feng**

School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China

ABSTRACT

Feature Pyramid Network (FPN) has become a common detection paradigm by improving multi-scale features with strong semantics. However, most FPN-based methods typically treat each feature map equally and sum the loss without distinction, which might lead to suboptimal overall performance. In this paper, we propose a Multi-scale Reinforcement Learning Strategy (MRLS) for balanced multi-scale training. First, we design Dynamic Feature Fusion (DFF) to dynamically magnify the impact of more important feature maps in FPN. Second, we introduce Compensatory Scale Training (CST) to enhance the supervision of the under-training scale. We regard the whole detector as a reinforcement learning system while the state bases on multi-scale loss. And we develop the corresponding action, reward, and policy. Compared with adding more rich model architectures, MRLS would not add any extra modules and computational burdens on the baselines. Experiments on MS COCO and PASCAL VOC benchmark demonstrate that our method significantly improves the performance of commonly used object detectors.

***Index Terms*—** Object detection, Reinforcement learning, Multi-scale, Balanced training

1. INTRODUCTION

Object detection is a fundamental task in computer vision. With the advances in deep learning, a number of deep detectors have achieved remarkable performance and speed [1, 2, 3]. These state-of-the-art research has shown that making better use of multi-scale features is significant to object detection. FPN [1] builds multi-scale feature representation by propagating the semantical information from high levels into low levels. It has become a common multi-scale detection paradigm. Based on it, many researchers have attempted to design various modules for further improvement. AugFPN [4] systematically demonstrates the defects in feature fusion and proposes residual feature augmentation. Cao *et al.* [5] introduce an attention-guided context extraction model to explore large contextual information.

This work was supported in part by the National Natural Science Foundation of China under Grant 61572214 and Seed Foundation of Huazhong University of Science and Technology (2020kfyXGYJ114). (Corresponding author: Qi Feng.)

Although these effective modules have improved the performance of multi-scale detectors, the space for further improvement of the training process is still large. We notice that two default operations are unreasonable, which might deteriorate the detection performance.

Firstly, most FPN-based methods treat each feature map equally in feature fusion. However, deep convolutional networks learn hierarchical features that capture different scale information [6]. Specifically, shallow spatial-rich features have high resolution and small receptive fields that are suitable for small objects. And deep semantic-rich features with large receptive fields for large objects. Therefore, each feature map contributes to the overall performance unequally.

Secondly, the loss function only distinguishes classification and regression, while the loss of each scale is summed without distinction. The gradient of each scale is inconsistent because that they have different numbers and resolutions of training samples. However, the scale with a large gradient will occupy a dominant position [7]. There is an imbalanced problem that the supervised training is not sufficient for scales with few gradients. The common solution is to balance the loss by weighting factors [8]. But empirical and fixed weights are not the optimal mechanism.

In this paper, we aim to balance multi-scale training without adding extra modules. We propose a Multi-scale Reinforcement Learning Strategy (MRLS) to optimize the weights of different scales dynamically. To address the first issue, we design Dynamic Feature Fusion (DFF) to magnify the impact of more important feature maps in FPN. Then we introduce Compensatory Scale Training (CST) to enhance the supervision of the under-training scale. We regard the whole detector as a reinforcement learning system while the state bases on multi-scale loss. And we develop the corresponding action, reward, and policy. Unlike other reinforcement-learning-based (RL-based) methods [9, 10, 11] that use DQN for bounding box refinement, our MRLS approximates a Markov decision process (MDP) and adopts a policy to perform actions. Compared with adding more rich model architectures, MRLS would not add any extra modules and computational burdens on the baselines.

We integrate MRLS into widely used object detectors to evaluate its effectiveness. Experiments on MS COCO [12] demonstrate that our method significantly improves the performance of baselines. And experimental results on PASCAL

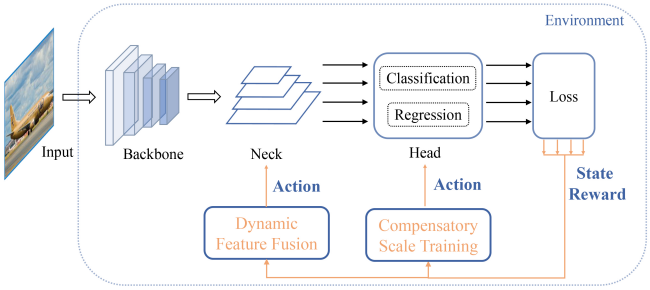

Fig. 1. An overview of our Multi-scale Reinforcement Learning Strategy (MRLS). We omit Region Proposal Network (RPN) for simplicity, and the detector architecture consists of three main components: Backbone, Neck and Head. MRLS can be easily embedded into the training process of commonly used detectors. The whole detector is regarded as a reinforcement learning system. Dynamic Feature Fusion (DFF) and Compensatory Scale Training (CST) optimize the scale weights through the reward function and policy.

VOC [13] benchmark indicate that MRLS can obtain superior performance than previous RL-based methods.

2. PROPOSED METHOD

2.1. Multi-scale Reinforcement Learning Strategy

The overall pipeline of our Multi-scale Reinforcement Learning Strategy (MRLS) is shown in Fig. 1. We take 4-level feature pyramid as an example to illustrate our methods. The multi-scale loss is defined as

$$L_{total} = \sum_{i=2}^{5} L_i, \tag{1}$$

where L_{total} is the total loss for backpropagation, and i denotes the index of multi-scale loss $\{L_2, L_3, L_4, L_5\}$. L_i consists of classification and localization loss, which follows the principle of baseline detectors. The corresponding multi-scale feature maps are denoted as $\{P_2, P_3, P_4, P_5\}$ produced by the FPN.

Our MRLS approximates a Markov decision process (MDP) and adopts a policy to perform actions. As formally defined in [14], a Markov decision process (MDP) can be described by a four-tuple (S, A, P, R). S represents a set of environment states, A is a set of actions, P represents the probability from one state translating to the other one, and R is a reward function that defines what are the good and bad events for the agent. Each element can be specifically defined as follows.

State. The first step is to cast the training process as a sequential state. We sum the loss of each scale in a certain iteration interval during the total iterations:

$$\mathcal{L}_{i,t} = \frac{1}{\sigma} \sum_{k=(t-1)\times\sigma+1}^{t\times\sigma} L_{i,k}, \quad t = 1, 2, 3, ... \quad \text{s.t.} \quad k \in (0, T], \tag{2}$$

where k is the current iteration, T denotes total iterations (e.g. training for 90K iterations), i indicates the index of the scale level, and σ is the interval step.

Then we define the current state as:

$$s_t = [\mathcal{L}_{2,t}, \quad \mathcal{L}_{3,t}, \quad \mathcal{L}_{4,t}, \quad \mathcal{L}_{5,t}]. \tag{3}$$

Unlike a general MDP where the current state is from a definite finite set $\mathcal{S}$, our s_t is a group of uncertain multi-loss values. Therefore, actions do not affect the state transition, which means the transition probability $P(s_{t+1} \mid s_t, a_t) = 1$.

Reward. The reward function returns a signal to award or punish the action. A classical reward [14] is proposed based on Intersection over Union (IoU). In our framework, the agent is expected to improve overall performance. The decrease in the total loss indicates that the detection performance is improved. With this different purpose, we propose a new loss-based reward function:

$$R_{a_{t-1}}(s_t, s_{t-1}) = \text{ReLU}\left(\frac{\sum_j \mathcal{L}_{j,t-1} - \sum_j \mathcal{L}_{j,t}}{\sum_j \mathcal{L}_{j,t-1}}\right), \tag{4}$$

where s_t denotes the state at time t, s_{t-1} is the previous one. $R_{a_{t-1}}$ is obtained at t, which indicates the reward value of the action at $t-1$. The initial value of $R_{a_{t-1}}$ is set to 0, and $\mathcal{L}_{i,t=0} = 0$. In Eq 4, we calculate the change rate of the total loss. A positive reward is given if the loss between two adjacent states drops, which indicates the degree of improvement. Otherwise, the reward value is equal to zero.

Below, we describe the corresponding action and policy of Dynamic Feature Fusion (DFF) and Compensatory Scale Training (CST).

2.2. Dynamic Feature Fusion

We expect that the features with relatively poor performance (larger loss value) can benefit from others. So we introduce a group of weighting factors for feature fusion in FPN. The process can be mathematically defined as

$$\mathcal{P}_i = \varphi_i(C_i) + \mathcal{W}_{i,t}\mathcal{P}_{i+1} \uparrow_{2\times}, \tag{5}$$

where $\mathcal{P}_i$ is the pyramid feature $\{P_2, P_3, P_4, P_5\}$, φ_i denotes 1x1 convolution, and $\uparrow_{2\times}$ means upsampling 2 times by bilinear interpolation. C_i is the output feature from the backbone. $\mathcal{W}_{i,t}$ is the weight of each scale. Different from the weight trained by backpropagation in EfficientDet [15], our $\mathcal{W}_t$ is generated according to the action and reward function. The action is an assignment operation to $\mathcal{W}_{i,t}$. The initial value of $\mathcal{W}_{i,t}$ at $t = 0$ is set to 1.

Action. We develop three actions $\{A_1, A_2, A_3\}$ for DFF. We expect that the feature map with a small loss value has great weight in feature fusion. Hence, A_1 is defined as:

$$\mathcal{W}_{i,t} = min(max(\frac{\mathcal{L}_{i,t}}{\mathcal{L}_{i+1,t}}, \beta_{min}), \beta_{max}), \tag{6}$$

where β_{min} and β_{max} are the boundary value for outliers. They are empirically set to 0.2 and 5.0.

According to the reward signal, the agent tends to choose the action with a higher reward. At time t, we can obtain the reward signal $R_{a_{t-1}}$ corresponding to $\mathcal{W}_{t-1}$. If $R_{a_{t-1}} > 0$, we can continue to adopt this assignment. Therefore, the second action A_2 is: $\mathcal{W}_t = \mathcal{W}_{t-1}$.

Besides, we introduce the best action corresponding to the highest reward $\{\mathcal{W}_{best}, R_{best}\}$, which is updated if the highest reward signal is obtained ($R_{a_{t-1}} > R_{best}$). This action is denoted as A_3: $\mathcal{W}_t = \mathcal{W}_{best}$. The difficulty of training is different in each epoch during model convergence [6] so that the significance of reward is changing over time. Hence, we adopt a drop probability to replace R_{best} with a positive $R_{a_{t-1}}$, which prevents R_{best} equals to a large value constantly. It can be mathematically defined as

$$P_{drop}\left(R_{best} = R_{a_{t-1}} \mid R_{a_{t-1}} > 0\right) = p_d, \tag{7}$$

where p_d is a relatively small value. We expect the agent to occasionally forget the best action so that p_d is set to 0.1 by experience. $\{\mathcal{W}_{best}, R_{best}\}$ are initialized to $\{1, 0\}$.

Policy. A typical reinforcement Learning system consists of exploring and exploiting [14]. On this basis, we expect DFF to explore more selections at the beginning of training and exploit for obtaining rewards later. The current action a_t is selected from the set of $\{A_1, A_2, A_3\}$ by probability P_a:

$$\begin{aligned}
&P_a\left(a_t = A_1\right) = p_1 = 0.5\gamma(t), \\
&P_a\left(a_t = A_2\right) = p_2 = (1 - p_1) \times \frac{R_{a_{t-1}} + \epsilon}{R_{a_{t-1}} + R_{best} + 2\epsilon}, \\
&P_a\left(a_t = A_3\right) = p_3 = (1 - p_1) \times \frac{R_{a_{best}} + \epsilon}{R_{a_{t-1}} + R_{best} + 2\epsilon},
\end{aligned} \tag{8}$$

$$\gamma(t) = \frac{T - t \times \sigma}{T}, \tag{9}$$

where $\gamma(t)$ is the discount factor about the time sequence and ϵ is a small value (1e-5) to avoid division by zero. At first $\gamma(t) = 1$, and $\gamma(t) \to 0$ when the training process ends.

2.3. Compensatory Scale Training

Similar to DFF, we introduce a group of weighting factors w_i in the loss computation based on Eq 1. Backpropagation cannot train w_i for loss function because w_i has no gradients. Thus, we adopt MDP to dynamically update w_i. The weighted loss function is defined as:

$$L_{total} = \sum_{i=2}^{5} w_i L_i. \tag{10}$$

Algorithm 1 Multi-scale Reinforcement Learning Strategy

Input:
$\mathcal{A}$: The action set for DFF.
$\mathcal{A}'$: The action set for CST.
σ: Update interval factor.
Output:
Trained object detector $\mathcal{D}$.

Begin
Initialize counter t, reward signal R, weighting factors $\mathcal{W}_i$ and w_i, probabilities P_a and $P_{a'}$
for $k = 0$ to max_iter **do**
  Train object detector $\mathcal{D}$ by Eq 10, and obtain the output multi-loss for s_t
  if $k\%\sigma == 0$ **then** #Execute every σ iteration
    $t+ = 1$
    Calculate reward R by Eq 4
    Determine R_{best} and trigger P_{drop}
    Execute $a_t \sim P_a$ and $a'_t \sim P_{a'}$
End

Action. We develop four actions $\{A'_1, A'_2, A'_3, A'_4\}$ for CST. As mentioned in Sec 1, the supervised training is not sufficient for the scale with less gradient. Hence, we expect to strengthen the two scales with the smallest proportion of the loss. The first action A'_1 is defined as:

$$rate_{i,t} = \frac{\mathcal{L}_{j,t}}{\sum \mathcal{L}_{i,t}}, \tag{11}$$

$$w_{i,t} = \begin{cases} 2 - rate_{i,t} & \text{if } rate_{i,t} \text{ is the smallest two} \\ 1 & \text{otherwise.} \end{cases} \tag{12}$$

Similar to the definitions of A_2 and A_3 in DFF, A'_2 is described as: $w_t = w_{t-1}$, and A'_3 is $w_t = w_{best}$.

Moreover, we observe that frequently modifying weighting factors in the later stage of training will affect the convergence of the model. In this case, it may be a good choice to make the multi-scale loss without weight. Hence, we define A'_4 as: $w_{i,t} = 1$.

Policy. The current action a'_t is selected from the set of $\{A'_1, A'_2, A'_3, A'_4\}$ by probability $P_{a'}$. Unlike P_a in DFF, we assign an increasing probability to A'_4. $P_{a'}$ is described as:

$$\begin{aligned}
&P_{a'}\left(a'_t = A'_1\right) = p'_1 = 0.5\gamma(t), \\
&P_{a'}\left(a'_t = A'_4\right) = p'_4 = 0.5(1 - \gamma(t)), \\
&P_{a'}\left(a'_t = A'_2\right) = p'_2 = 0.5 \times \frac{R_{a'_{t-1}} + \epsilon}{R_{a'_{t-1}} + R_{best} + 2\epsilon}, \\
&P_{a'}\left(a'_t = A'_3\right) = p'_3 = 1 - p'_1 - p'_2 - p'_4.
\end{aligned} \tag{13}$$

We summarize the whole procedure of MRLS (DFF+CST) in Algorithm 1. It is embedded at the end of model backpropagation and executed every σ iterations. MRLS has brought a negligible computational burden in the training process because it does not require backpropagation.

3. EXPERIMENTS

In this section, we conduct experiments on the MS COCO [12] and PASCAL VOC [13] benchmark. In MS COCO, we train all the models on train-2017 (115k images) and report results on val-2017 (5k images). The performance metrics follow the standard COCO-style mean Average Precision (mAP) under different IoU thresholds, ranging from 0.5 to 0.95 with an interval of 0.05. AP_S, AP_M, and AP_L are the score of small, medium, and large objects. As for Pascal VOC, we merge the VOC2007 trainval and VOC2012 trainval split for training and evaluate on VOC2007 test split. The mAP for VOC is under 0.5 IoU. We train the detectors on 2 NVIDIA Quadro P5000 GPUs (2 images per GPU) for 12 epochs. All experiments are implemented based on mmdetection [16]. Settings and hyperparameters follow the default configuration.

3.1. Ablation Study

To evaluate the effectiveness of each component for MRLS, we conduct ablation experiments on MS COCO val-2017. The baseline is RetinaNet [17] with ResNet-101 [18]. We also report the inference time of the detectors to prove that our methods do not add extra computational burdens.

Table 1. Ablation study of MRLS. The baseline is RetinaNet with ResNet-101.

Method	mAP	FPS
baseline	38.2	8.2
baseline + DFF w fixed weights	38.3	8.2
baseline + **DFF**	**38.8**	
baseline + CST w fixed weights	38.6	8.2
baseline + **CST**	**39.1**	
baseline + **MRLS**	**39.6**	8.2

We assemble DFF into the baseline and the mAP is boosted to 38.8 as illustrated in Table 1. Then we integrate CST into the baseline and obtain 39.1 mAP. When implementing MRLS (DFF+CST) the detection performance is improved by 1.4 points. Experimental results validate the effectiveness of our proposed methods.

We also implement fixed weights (non-reinforcement-learning) for DFF and CST. The fixed weights are generated by MRLS at the end of training. Table 1 demonstrates that the mAP has only been improved by 0.1 and 0.4 respectively. It illustrates that the dynamic reinforcement learning strategy performs better than the stationary method.

3.2. Performance and Comparison

We integrate MRLS into a variety of widely used object detectors (RetinaNet [17], FCOS [19], and ATSS [3]) with different backbone networks (ResNet-50, ResNet-101, and ResNet-101-DCN [20]) to evaluate the generalization ability. We train ATSS with ResNet-101-DCN to obtain higher performance. As shown in Table 2, experiments on MS COCO val-2017 demonstrate that our method significantly improves the performance of baselines.

Table 2. Detection performance of different detectors on MS COCO val-2017.

Method	Backbone	AP	AP_S	AP_M	AP_L
RetinaNet [17]	ResNet-50	36.1	19.8	39.7	47.1
[17] + **MRLS**		**37.7**	21.2	41.1	49.4
RetinaNet [17]	ResNet-101	38.2	22.2	42.5	49.9
[17] + **MRLS**		**39.6**	22.8	43.5	51.2
FCOS [19]	ResNet-50	36.6	21.0	40.6	47.0
[19] + **MRLS**		**37.6**	21.4	41.2	49.1
FCOS [19]	ResNet-101	39.0	22.7	43.3	50.3
[19] + **MRLS**		**39.9**	23.0	43.9	51.5
ATSS [3]	ResNet-50	39.0	23.2	42.7	49.8
[3] + **MRLS**		**40.1**	23.5	43.5	52.3
ATSS [3]	ResNet-	44.2	25.5	48.6	58.3
[3] + **MRLS**	101-DCN	**45.0**	26.2	49.3	59.2

We also compare our approach with other representative RL-based methods including Tree-RL [9], dlr-RPN [10], and PAT [11]. We implement our MRLS with RetinaNet + ResNet-101. As shown in Table 3, experimental results on PASCAL VOC indicate that MRLS can obtain superior performance than other RL-based methods.

Table 3. Comparisons with the other RL-based methods on PASCAL VOC 2007 test.

Method	Data	Epochs	mAP
PAT [11]	07+12	20	75.9
dlr-RPN [10]	07+12	-	76.4
Tree-RL [9]	07+12	25	76.6
MRLS	07+12	12	**80.4**

4. CONCLUSION

In this paper, we propose a Multi-scale Reinforcement Learning Strategy (MRLS) for balanced multi-scale training. The training approach generates weighting factors for future fusion and loss function through an approximate Markov decision process (MDP). Experimental results validate the effectiveness of the proposed methods on MS COCO and PASCAL VOC benchmark. In future work, we will study the generalization of our methods on more multi-scale vision tasks.

5. REFERENCES

[1] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.

[2] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[3] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9756–9765.

[4] Chaoxu Guo, Bin Fan, Qian Zhang, Shiming Xiang, and Chunhong Pan, "Augfpn: Improving multi-scale feature learning for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12595–12604.

[5] Junxu Cao, Qi Chen, Jun Guo, and Ruichao Shi, "Attention-guided context feature pyramid network for object detection," *arXiv preprint arXiv:2005.11475*, 2020.

[6] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin, "Libra r-cnn: Towards balanced learning for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 821–830.

[7] Alex Kendall, Yarin Gal, and Roberto Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7482–7491.

[8] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas, "Imbalance problems in object detection: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 10, pp. 3388–3415, 2021.

[9] Zequn Jie, Xiaodan Liang, Jiashi Feng, Xiaojie Jin, Wen Feng Lu, and Shuicheng Yan, "Tree-structured reinforcement learning for sequential object localization," in *Advances in Neural Information Processing Systems*, 2016, pp. 127–135.

[10] Aleksis Pirinen and Cristian Sminchisescu, "Deep reinforcement learning of region proposal networks for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6945–6954.

[11] Songtao Liu, Di Huang, and Yunhong Wang, "Pay attention to them: deep reinforcement learning-based cascade object detection," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 7, pp. 2544–2556, 2019.

[12] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick, "Microsoft coco: Common objects in context," in *Proceedings of the European conference on computer vision (ECCV)*, 2014, pp. 740–755.

[13] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John M. Winn, and Andrew Zisserman, "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[14] Ngan Le, Vidhiwar Singh Rathour, Kashu Yamazaki, Khoa Luu, and Marios Savvides, "Deep reinforcement learning in computer vision: A comprehensive survey," *arXiv preprint arXiv:2108.11510*, 2020.

[15] Mingxing Tan, Ruoming Pang, and Quoc V Le, "Efficientdet: Scalable and efficient object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10781–10790.

[16] Kai Chen, Jiaqi Wang, Jiangmiao Pang, and et al., "MMDetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2017, pp. 2980–2988.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[19] 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*, 2019, pp. 9627–9636.

[20] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai, "Deformable convnets V2: more deformable, better results," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9308–9316.