DYNAMIC MULTI-SCALE LOSS BALANCE FOR OBJECT DETECTION

Yihao Luo[1], *Xiang Cao*[1], *Juntao Zhang*[1], *Peng Cheng*[2], *Tianjiang Wang*[1], *Qi Feng*[1,*]

[1]School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
[2]Coolanyp Limited Liability Company, Kirkland, State of Washington, USA

ABSTRACT

It is a common paradigm in object detection frameworks to perform multi-scale detection. However, each scale is treated equally during training. In this paper, we carefully study the objective imbalance of multi-scale detector training. We argue that the loss in each scale is neither equally important nor independent. Different from the existing solutions of setting fixed multi-task weights, we dynamically optimize the loss weight of each scale in the training process. Specifically, we propose an Adaptive Variance Weighting (AVW) to balance multi-scale loss according to the statistical variance. Then we develop a novel Reinforcement Learning Optimization (RLO) to decide the weighting scheme probabilistically during training. The proposed dynamic methods make better utilization of multi-scale training loss without extra computational complexity and learnable parameters for backpropagation. Experiments on Pascal VOC and MS COCO benchmark validate the effectiveness of our proposed methods.

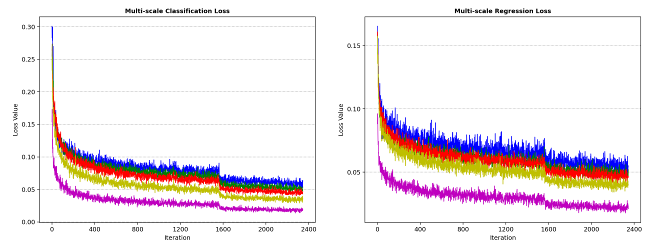
Index Terms— Object detection, Multi-scale Imbalance, Reinforcement learning, Multi-task

1. INTRODUCTION

Benefiting from the advances in deep convolutional neural networks, current a number of deep detectors have achieved remarkable performance [1, 2, 3]. Meanwhile, imbalance problems in object detection have also received significant attention gradually [4].

The most commonly known imbalance problem is the foreground-to-background imbalance. The hard mining methods [4, 5] have alleviated it to a certain extent, where the contribution of a sample is adjusted with weight through a fixed optimization strategy. Recently more sample weighting methods [6, 7] achieve further improvement of detection performance. The weight of samples is trained by fully connected layers or generated by a more complex procedure. As for the scale imbalance, Feature Pyramid Network (FPN) [1] shows outstanding ability in handling the scale diversity of the input bounding boxes. Afterward, extensive FPN-based modules [2, 4] are proposed for enhancing multi-scale feature representation. FPN has become a typical paradigm for multi-scale detection.

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.)



(a) (b)

Fig. 1. An illustration of the imbalance problem in multi-scale detectors training. (a), (b): The training loss of classification and regression through 5 scale predictions. The blue, green, red, yellow, and pink curves represent the prediction loss every 100 iterations of 5 feature scales from large to small. It is trained through RetinaNet [5] with ResNet-50 and 12 epochs on MMDetection [8].

Multi-scale feature maps are created by propagating the high-level semantical information into low-level in FPN. Then predictions (recognition and localization) are independently made on each level while the loss of all scales is summated without distinction for backpropagation [1]. However, we observe that the loss value is uneven and unstable between each scale in Fig 1, which can summarize into three points: (1) The multi-scale loss fluctuates during the training process. (2) Although the total loss of each scale is averaged based on the number of samples, the larger-scale feature map level has a greater loss. (3) The fluctuation of the regression loss value is greater than that of the classification branch.

Based on the illuminating observations, we believe that training multi-scale detectors is also a type of multi-task learning, in addition to classification and regression. The 5-scales can be seen as 5-tasks. In multi-task learning, the gradient norm of each task is different, and the task with a large gradient will occupy a dominant position [9]. If multi-loss is not balanced, the overall performance may be suboptimal [10]. Therefore, we argue that training multi-scale detectors also suffers from objective imbalance. The conventional way to solve it is to weight the tasks [11]. Recent [12] employs the correlation between the classification and regression tasks to assign the weights. However, training the detector of each scale is the same task in multi-scale training. It is difficult to determine the weight of each scale.

In this work, we aim to analyze and alleviate the pointed imbalance problem of multi-scale detectors training. Inspired by the uncertainty weighting [9], we propose Adaptive Variance Weighting (AVW) to balance multi-scale loss. Unlike previous works [6, 7] that estimate the standard deviation through extra networks, AVW calculates the statistical variance of the training loss in each level to measure their importance. The loss contributed by the important (high variance reduction rate) scale is enhanced during training. For further improvement, we develop a novel Reinforcement Learning Optimization (RLO). We regard multi-scale detectors as multi-agents, which seek the optimal decision in different training stages. RLO dynamically decides the optimal scheme for adapting to different phases in model training. In particular, our methods introduce no extra computational complexity and learnable parameters for backpropagation. Unlike other reinforcement-learning-based (RL-based) methods [13, 14, 15] that use DQN for bounding box refinement, RLO approximates a Markov decision process (MDP) and performs actions by probability.

We implement RetinaNet [5], Faster R-CNN [16], and ATSS [17] with ResNet-50 and ResNet-101 as the baselines in our experiments to investigate the impacts of our methods. Experimental results on the MS COCO [18] and Pascal VOC [19] benchmark validate the effectiveness of our proposed approaches.

2. PROPOSED METHOD

2.1. Problem Formulation

The overall pipeline is shown in Fig. 2. Following mmdetection [8], the detector consists of three main components: Backbone, Neck and Head. The multi-scale loss produced by Head is input into our algorithm for optimizing dynamically. Then the adjusted loss of each scale is summed for backpropagation according to the default configuration. We use the FPN as the default configuration for multi-scale detection. Our study focuses on balancing multi-scale loss, so the classification loss $L_{cls,i}$ and regression loss $L_{reg,i}$ are all regarded as L_i. We take the RetinaNet of 5 levels pyramid as an example to illustrate our methods.

To solve this objective imbalance, we adjust the multi-scale loss by weighting factors, which can be mathematically defined as

$$L_{total} = \sum_{i=3}^{7} w_i L_i, \quad (1)$$

where L_{total} is the total loss for backpropagation. The most

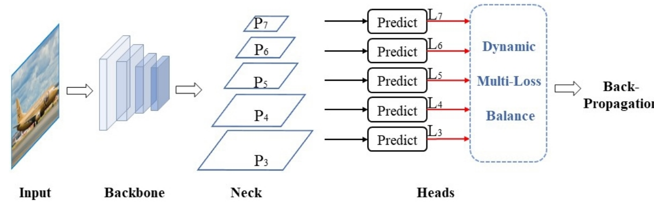


Fig. 2. An overview of our learning strategy. Our proposed Adaptive Variance Weighting (AVW) and Reinforcement Learning Optimization (RLO) are two algorithm configurations for multi-loss optimization.

common approach in multi-task learning is to linearly combine the losses by manual weighting, which is not an optimized approach to address the objective imbalance of the tasks [11]. It is significant how to determine the appropriate loss weight for the current training phase. Hence, we propose Adaptive Variance Weighting (AVW) and Reinforcement Learning Optimization (RLO) to dynamically optimize w_i, which will be described in detail.

2.2. Adaptive Variance Weighting

The multi-task learning method [9] demonstrates the relationship between uncertainty and variance. Different from the 'variance' trained through fully connected layers [7], we calculate the statistical variance of each scale level in the iteration interval. Under different loss values, the decrease in variance means that the training results are more stable and confident. We believe that the scales with high statistical variance reduction rates are more important, whose loss needs to be enhanced during training to prevent other larger gradients from occupying a dominant position. We treat iteration as a time dimension and redefine the interval loss of each scale as:

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

where m is the current iteration, T denotes total iterations, $i \in \{3, 4, 5, 6, 7\}$ indicates the index of the scale, t denotes the time sequence, and α is the iteration interval. Then the variance of each loss value per iteration interval can be calculated as

$$Var_{i,t} = \frac{1}{\alpha - 1} \sum_{m=(t-1)\times\alpha+1}^{t\times\alpha} (L_{i,m} - \frac{\mathcal{L}_{i,t}}{\alpha})^2, \quad (3)$$

$$r_{i,t} = \frac{Var_{i,t-1} - Var_{i,t}}{Var_{i,t-1} + \epsilon}, \quad (4)$$

where $r_{i,t}$ denotes the rate of decrease in variance, and $\epsilon = 0.00001$ is a small value to avoid numerical instability. The initial value of $r_{i,t}$ and $Var_{i,t-1}$ are set to 0, 1 respectively.

We believe that the two scales with the maximal $r_{i,t}$ are more important than the others, whose weights are selected to be increased. We have studied the number of scales that require enhancement through experiments. It is indicated that 2 is the optimal number. Besides, increasing the loss of all scales is similar to enlarging the learning rate, which is not in the scope of our research. w_i of these two scales are adjusted to a larger value as

$$w_j = \begin{cases} 1 + \lambda \frac{\mathcal{L}_{j,t}}{\sum \mathcal{L}_{i,t}} & \text{if j is the selected scales,} \\ 1 & \text{otherwise,} \end{cases} \tag{5}$$

where j denotes the index of the selected two scales, and λ is an amplification factor, set to 1.5 empirically. w_i is initialized as 1.0 and updated at the end of every iteration interval α.

2.3. Reinforcement Learning Optimization

In addition to the viewpoint of variance, there are also many concepts about loss and gradient. Some studies [20, 5] prefer a more significant magnitude of loss and gradient to accelerate training, while some others [6, 12] attach importance to the samples that have performed better. Hence, we raise the following questions: Is it better to enhance the level with a large loss? Could enhancing scales of less loss be conducive to performance improvement? Does modifying the weights will make the performance worse?

These concepts all seem to make sense. Therefore, we introduce a novel Reinforcement Learning Optimization (RLO), which decides the weighting scheme for dynamically adapting to different training phases. RLO approximates a Markov decision process (MDP) and performs actions by probability. The multi-scale loss $\{\mathcal{L}_{i,t}\}$ can be seen as the state of the environment. Other elements are specifically defined as follows:

Action. AVW is essentially a subjective greedy strategy, which may not be suitable for the entire training epochs. So we take AVW as the first action in set $\mathcal{A}$. Then we develop the corresponding actions to the above viewpoints. We denote the action set as $\mathcal{A} = \{a_0, a_1, a_2, a_3\}$. The agent decides an action A_t at time t to execute through the policy. And each action in $\mathcal{A}$ is defined as follows:

a_0 (AVW): Select the two levels with the maximal $r_{i,t}$, and update w_i by Eq 5.

a_1 : Select the two levels with the minimal $\mathcal{L}_{i,t}$, and update w_i by Eq 5.

a_2 : Select the two levels with the maximal $\mathcal{L}_{i,t}$, and update w_i by Eq 5.

a_3 : Each w_i is updated to 1.

Reward. The reward function is critical to agent building, which defines what are the good and bad events for the agent. In the reinforcement learning system, the agent should improve overall performance rather than individual optimization. The intuitive result is that the total loss has decreased in the training process. To this end, we design the reward function as

$$R_{A_{t-1}}(\mathcal{S}_{t-1}, \mathcal{S}_t) = \text{sign}\left(\sum \mathcal{L}_{i,t-1} - \sum \mathcal{L}_{i,t}\right), \tag{6}$$

where $\mathcal{S}_t$ denotes the state at time t, $\mathcal{S}_{t-1}$ is the previous one. $R_{A_{t-1}}$ is obtained at t, which represents the reward signal of the action A_{t-1}. The initial value of the reward is set to 0. A positive reward is returned if the total loss between two adjacent states drops. Otherwise, its value is less than or equal to zero.

Policy. We devise a probabilistic decision policy to guide the agent. Each action in $\mathcal{A}$ is assigned a selection probability. At time t, the probability of performing action a_k is p_k: $P_{\mathcal{A}}(A_t = a_k) = p_{k,t}$, $(k = 0, 1, 2, 3)$. To make a trade-off between exploration and development, we do not introduce any prior knowledge. The initial values of p_k are all 0.25.

According to the reward signal, a positive value will increase the probability of the corresponding action. Otherwise, the probability will decrease. The process is described as

$$\begin{aligned} &p_{k,t} = min(max((p_{k,t-1} \pm \gamma), \beta_{min}), \beta_{max}), \quad \text{if } A_{t-1} = a_k, \\ &p_{q,t} = p_{q,t-1} \frac{1 - p_{k,t}}{1 - p_{k,t-1}}, \quad p_{q,t} \in \mathcal{P}_{\mathcal{A}} - p_{k,t}, \end{aligned} \tag{7}$$

where $p_{k,t}$ is the probability of the previous action A_{t-1}, which requires award or punishment first through a small factor γ. γ is empirically set to 0.01. $\pm$ indicates that if $R_{A_{t-1}} > 0$ performing $+$, otherwise $-$. After updating $p_{k,t}$, the remaining three probabilities $p_{q,t}$ are modified proportionally, which asserts the sum of all probabilities is 1. And $\beta_{min} = 0.1, \beta_{max} = 0.9$ are the boundary value of $p_{k,t}$ if outliers.

RLO is embedded at the end of model backpropagation and executed every α iterations. It has brought a negligible computational burden in the training process because it does not require backpropagation.

3. EXPERIMENTS

We perform our experiments on MS COCO [18] and Pascal VOC [19] benchmark. We train all the models on MS COCO train-2017 and report results on val-2017. The performance metrics follow the standard COCO-style mean Average Precision (mAP). And we merge the VOC2007 trainval and VOC2012 trainval split for training and evaluate on VOC2007 test split. We train the detectors on 4 NVIDIA Quadro P5000 GPUs for 12 epochs. All experiments are implemented based on mmdetection [8]. Settings and hyperparameters follow the default configuration in mmdetection.

3.1. Ablation Study

To evaluate the effectiveness of each action, we conduct ablation experiments on MS COCO val-2017. The baseline is RetinaNet [5] with ResNet-50. We also report the

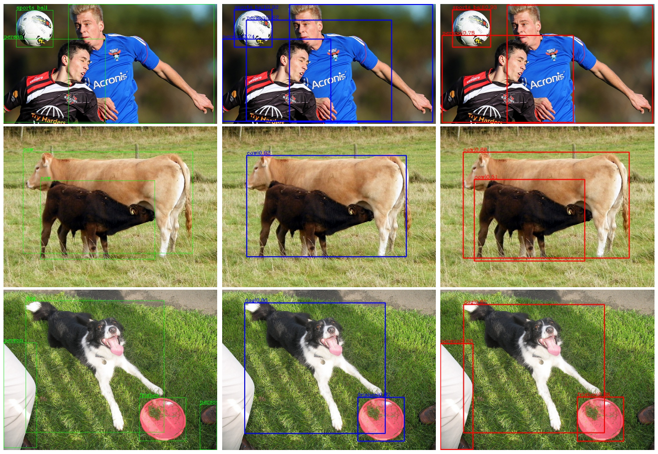


Fig. 3. Qualitative results comparison. The first column is the ground truth with green bounding boxes. The results of the original RetinaNet are listed by the blue bounding boxes, while those of RLO are the red bounding boxes.

Giga Floating-point Operations Per Second (GFLOPS) of the detectors to prove that our methods do not add extra computational burdens. Table 1 demonstrates that performing a fixed scheme throughout the training process may lead to sub-optimal. Optimizing strategies dynamically during different phases is significant to multi-scale detectors training.

Table 1. Effect of each action on COCO val-2017.

action 0 (**AVW**)	action 1	action 2	action 3 (baseline)	AP	GFLOPs
			✓	36.1	250.34
✓				36.6	250.34
	✓			36.3	
		✓		36.0	
	✓	✓		36.4	
✓	✓	✓	✓	36.9	

3.2. Performance and Comparison

We integrate our method into a variety of widely used object detectors (RetinaNet [5], Faster R-CNN [16], and ATSS [17]) with different backbone networks (ResNet-50, ResNet-101) to evaluate the generalization ability. As shown in Table 2, experiments on MS COCO val-2017 validate the effectiveness of our methods. RLO performs better in one-stage detectors than two-stage detectors.

Then we compare our approach with other representative RL-based methods including Tree-RL [13], dlr-RPN [14], and PAT [15]. We implement our RLO with RetinaNet + ResNet-101. As shown in Table 3, experimental results on PASCAL VOC indicate that RLO can obtain superior performance than other RL-based methods.

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

Method	Backbone	AP	AP_S	AP_M	AP_L
YOLOv3 [21]	DarkNet-53	33.7	19.4	36.8	44.3
YOLOF [22]	ResNet-50	37.5	19.0	42.0	53.2
Foveabox [23]	ResNet-50	36.5	20.5	39.9	47.7
GHM [24]	ResNet-101	39.0	22.1	42.8	51.9
RetinaNet [5]	ResNet-50	36.1	19.8	39.7	47.1
[5] + **RLO**		**36.9**	20.8	40.1	48.5
RetinaNet [5]	ResNet-101	38.2	22.2	42.5	49.9
[5] + **RLO**		**38.9**	21.6	43.1	51.2
Faster R-CNN [16]	ResNet-50	37.2	21.3	40.8	47.9
[16] + **RLO**		**37.7**	22.1	41.1	48.2
Faster R-CNN [16]	ResNet-101	39.2	22.5	43.3	51.4
[16] + **RLO**		**39.5**	22.3	43.9	52.0
ATSS [17]	ResNet-50	39.0	23.2	42.7	49.8
[17] + **RLO**		**39.9**	23.4	43.3	52.4
ATSS [17]	ResNet-101	41.2	24.1	45.3	52.7
[17] + **RLO**		**41.7**	24.2	46.1	53.5

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

Method	Data	Epochs	mAP
PAT [15]	07+12	20	75.9
dlr-RPN [14]	07+12	-	76.4
Tree-RL [13]	07+12	25	76.6
RLO	07+12	12	**80.1**

We also show the comparisons of the qualitative results in Fig. 3. Both models are built upon ResNet-50 + FPN. The images are chosen from COCO val-2017. We compare the detection performance with threshold = 0.5. It can be seen that RetinaNet+RLO generates satisfactory results especially in terms of large objects, while the original RetinaNet generates inferior results.

4. CONCLUSION

In this paper, we argue that training multi-scale detectors is also a type of multi-task learning in addition to classification and regression, which suffers from an objective imbalance. Based on our observation, we propose Adaptive Variance Weighting and Reinforcement Learning Optimization to improve multi-scale training. Experiments show that our approaches can consistently boost the performance over various baseline detectors without extra computational complexity. We hope that our viewpoint can inspire further researches on multi-scale training and reinforcement learning.

5. REFERENCES

[1] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017, pp. 2117–2125.

[2] Chaoxu Guo, Bin Fan, Qian Zhang, Shiming Xiang, and Chunhong Pan, "Augfpn: Improving multi-scale feature learning for object detection," in *CVPR*, 2020, pp. 12595–12604.

[3] Hongkai Zhang, Hong Chang, Bingpeng Ma, Naiyan Wang, and Xilin Chen, "Dynamic R-CNN: towards high quality object detection via dynamic training," in *ECCV*, 2020, pp. 260–275.

[4] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin, "Libra r-cnn: Towards balanced learning for object detection," in *CVPR*, 2019, pp. 821–830.

[5] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár, "Focal loss for dense object detection," in *ICCV*, 2017, pp. 2980–2988.

[6] Yihui He, Chenchen Zhu, Jianren Wang, Marios Savvides, and Xiangyu Zhang, "Bounding box regression with uncertainty for accurate object detection," in *CVPR*, 2019, pp. 2888–2897.

[7] Qi Cai, Yingwei Pan, Yu Wang, Jingen Liu, Ting Yao, and Tao Mei, "Learning a unified sample weighting network for object detection," in *CVPR*, 2020, pp. 14161–14170.

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

[9] Alex Kendall, Yarin Gal, and Roberto Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *CVPR*, 2018, pp. 7482–7491.

[10] Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei, "Dynamic task prioritization for multitask learning," in *ECCV*, 2018, pp. 282–299.

[11] 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.

[12] Yuhang Cao, Kai Chen, Chen Change Loy, and Dahua Lin, "Prime sample attention in object detection," in *CVPR*, 2020, pp. 11580–11588.

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

[14] Aleksis Pirinen and Cristian Sminchisescu, "Deep reinforcement learning of region proposal networks for object detection," in *CVPR*, 2018, pp. 6945–6954.

[15] 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.

[16] 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.

[17] 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 *CVPR*, 2020, pp. 9756–9765.

[18] 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 *ECCV*, 2014, pp. 740–755.

[19] Mark Everingham and et al., "The pascal visual object classes (VOC) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[20] Abhinav Shrivastava, Abhinav Gupta, and Ross B. Girshick, "Training region-based object detectors with online hard example mining," in *CVPR*, 2016, pp. 761–769.

[21] Joseph Redmon and Ali Farhadi, "Yolov3: An incremental improvement," *arXiv:1804.02767*, 2018.

[22] Qiang Chen, Yingming Wang, Tong Yang, Xiangyu Zhang, Jian Cheng, and Jian Sun, "You only look one-level feature," in *CVPR*, 2021.

[23] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, Lei Li, and Jianbo Shi, "Foveabox: Beyound anchor-based object detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 7389–7398, 2020.

[24] Buyu Li, Yu Liu, and Xiaogang Wang, "Gradient harmonized single-stage detector," in *AAAI*, 2019, pp. 8577–8584.