Consistent Optimization for Single-Shot Object Detection

Tao Kong[1]† Fuchun Sun[1] Huaping Liu[1] Yuning Jiang[2] Jianbo Shi[3]
[1]Department of Computer Science and Technology, Tsinghua University,
Beijing National Research Center for Information Science and Technology (BNRist)
[2]ByteDance AI Lab [3]University of Pennsylvania
taokongcn@gmail.com, {fcsun,hpliu}@tsinghua.edu.cn,
jiangyuning@bytedance.com, jshi@seas.upenn.edu

Abstract

We present consistent optimization for single stage object detection. Previous works of single stage object detectors usually rely on the regular, dense sampled anchors to generate hypothesis for the optimization of the model. Through an examination of the behavior of the detector, we observe that the misalignment between the optimization target and inference configurations has hindered the performance improvement. We propose to bride this gap by consistent optimization, which is an extension of the traditional single stage detector's optimization strategy. Consistent optimization focuses on matching the training hypotheses and the inference quality by utilizing of the refined anchors during training.

To evaluate its effectiveness, we conduct various design choices based on the state-of-the-art RetinaNet detector. We demonstrate it is the consistent optimization, not the architecture design, that yields the performance boosts. Consistent optimization is nearly cost-free, and achieves stable performance gains independent of the model capacities or input scales. Specifically, utilizing consistent optimization improves RetinaNet from 39.1 AP to 40.1 AP on COCO dataset without any bells or whistles, which surpasses the accuracy of all existing state-of-the-art one-stage detectors when adopting ResNet-101 as backbone. The code will be made available.

1. Introduction

We are witnessing the remarkable progress for object detection by exploring the powerful deep leaning technology. Current state-of-the-art deep learning based object detection frameworks can be generally divided into two major groups: two-stage, proposal-driven methods [34][11][23] and one-stage, proposal-free methods [26][31]. Compared with the two-stage framework, one-stage object detector is more difficult, since it relies on a powerful model head to predict the regular, dense sampled anchors at different locations, scales, and aspect ratios. The main advantage of one-stage detector is its high computational efficiency. However, its detection accuracy is usually behind that of the two-stage approach, one of the main reasons being due to the class imbalance problem [24][20]. The most recent work of RetinaNet [24], is able to match the accuracy of the existing state-of-the-art two-stage detectors, by utilizing Focal Loss to address the *foreground-background class imbalance* challenge.

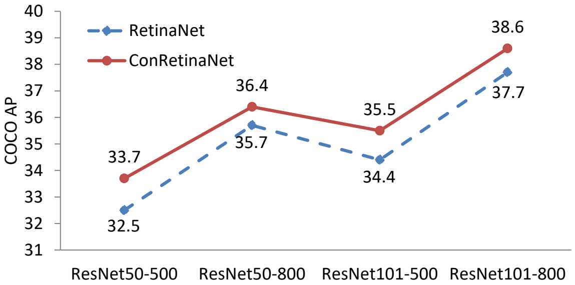
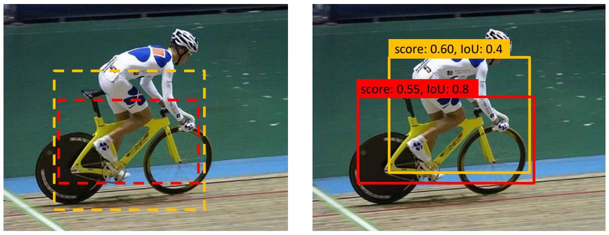


Figure 1: RetinaNet *v.s.* ConRetinaNet on different model capacities and input resolutions. An improved variant of ConRetinaNet achieves 40.1 AP with ResNet-101 backbone, which is not shown in this figure. Details are given in §6.

In this work, we observe that in addition to the foreground-background class imbalance challenge, current single shot object detectors also face another challenge: *the misalignment between the training targets and inference configurations.* Given the candidate anchors, the one-stage object detector combines both object *classification* and *localization*. At training phase, the goal of classification subnetworks is to assign the candidate anchor to one of $M+$ 1 classes, where class 0 contains background and the re-

†Part of the work was done when Tao Kong was a visiting scholar at University of Pennsylvania.

第一个问题
a两个框在分类的时候都分成车，并且黄色框的score更高
b回归的时候，黄色的框与人的gt回归（类间遮挡严重）
导致：红色的框在NMS的时候被干掉



(a) (b)

Figure 2: Demonstrative case of the misalignment between training target and predicted results. (a) There are two anchors matching the groundtruth *bicycle*. (b) The resulting scores and IoUs of the corresponding refined anchors for class *bicycle*. The red bounding box will be suppressed during NMS procedure.

maining the objects to detect. The localization module finds the optimal transformation for the anchor to best fit the groundtruth. At inference, for each anchor, the location is refined by the regression sub-networks. The class probability of the refined anchor is given by the classification sub-networks. The misalignment is: *the training target of the classification is to classify the default, regular anchor, while the predicted probability is assigned to the corresponding regressed anchor which is generated by the localization branch.*

Such training-inference configuration works well when the original anchor and refined anchor share the same groundtruth target. However, it fails in two circumstances. (a) When two objects are occluded to each other, the regression is apt to get confused about the moving direction. In Figure 2, there are two anchors that match the bicycle. So the detector treats these anchors as class *bicycle*, then tries to optimize the classification loss and offsets between the anchors and groundtruth. However since the *person* and the *bicycle* are with very significant inter-class occlusion, the anchor in yellow is incorrectly regressed to the *person*. The misalignment may lead to accurately located anchors being suppressed by misleading ones in the NMS procedure. (b) Some anchors assigned as negative samples may also match the groundtruth well after regression. Unfortunately these samples are eliminated due to low class scores. In Figure 3, we plot the IoUs of the anchors with the nearest groundtruth before and after regression. Some anchors with low IoUs also match the groundtruth well after being regressed. Detection model that is optimized only based on the original anchors is loose and inaccurate. Given the above analysis, a natural question to ask is: *could we use the input hypothesis of the more accurate, regressed anchors for the optimization of the detector*? Utilizing the regressed anchor hypothesis at training phase could bridge the gap between optimization and prediction for each detection.

第二个问题
在做reg之前，anchor与gt的iou可能比较低，因此在分类的时候分成
了负样本，但是回归之后可能与gt有比较高的iou，此时虽然iou高了
但是由于使用的是reg之前的anchor分类，因此score很低，被干掉

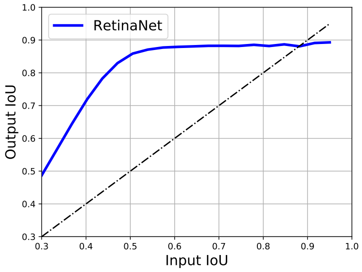


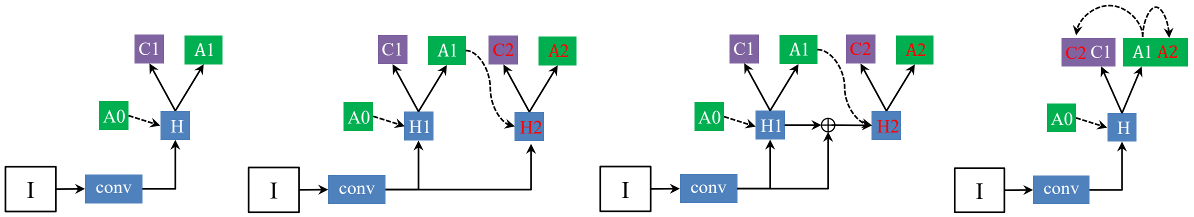
Figure 3: The localization performance. The plots are based on RetinaNet with ResNet-50 backbone [24].

There are several design choices to utilize the refined anchor for training, as shown in Figure 4. The direct way is to adopt the cascade manner, motivated from Cascade R-CNN [4], as shown in Figure 4(b) and Figure 4(c). Through comparison of several typical design choices, we find that it is the final optimization target, not the architecture designing tricks that plays the key role to boost the detector's performance. In this paper, we propose to use the refined anchor hypothesis for training the detector to keep the consistency, and to use the design architecture of Figure 4(d) to implement upon the detector. In the proposed implementation, the refined anchor is used both for classification and regression. At training phase given the regular, dense sampled candidate anchors, the object detector not only adopts the original anchors to perform classification and regression, but also utilizes the refined anchor to further optimize the same model. We name the proposed solution as consistent optimization, since it's goal is to resolve the inconsistency of training-inference for single stage object detector.

To validate it's effectiveness, we add consistent optimization on the state-of-the-art RetinaNet [24]. Our model, named as ConRetinaNet, is quite simple to implement and trained end-to-end. The results show that a vanilla implementation outperforms RetinaNet with different model capacities (ResNet-50/ResNet-101), input resolutions (short-size from 500 to 800), and localization qualities on challenging MS COCO dataset [25] (Figure 1). The improvements are consistent with almost no additional computation or parameters. In particular, ConRetinaNet is able to achieve 40.1 AP on the MS COCO dataset, which is the first ResNet-101 based single stage object detector to achieve such performance without any testing time bells or whistles. We believe that this simple and effective solution can be of interest for many object detection research efforts.

2. Related work

Classic Object Detectors: Prior to the widely development of deep convolutional networks, the DPM [9] and its variants [8][3] have been the dominating methods for years. These methods use image descriptors such as HOG



(a) RetinaNet (b) Cascade (c) Cascade with Gated Feature (d) Ours

Figure 4: The architectures of single shot detection frameworks. "I" is input image, "conv" backbone convolutions, "H" convolutional network head, "A" anchor box, "C" classification, "$A0$" the original anchor, and "$A1$" the refined anchor. In (b)-(d), the refined anchor hypothesis is utilized to further optimize the model.[1]

[7], SIFT [27], and LBP [41] as features and sweep through the entire image to find regions with a class-specific maximum response. There are also many efforts of region proposal generation. These methods usually adopt cues like superpixels [40], edges [46], saliency [1] and shapes [2] as features to generate category-independent region proposals to reduce the searching space of the target objects.

Two-stage Detectors: After the remarkable success of applying deep convolutional neural networks (CNNs or ConvNets) on image classification tasks, deep learning based approaches have been actively explored for object detection, especially for the region-based convolutional neural networks (R-CNN) and its variants. The SPP-Net [14] and Fast R-CNN [11] speed up the R-CNN approach with RoI-Pooling that allows the classification layers to reuse the CNN feature maps. Since then, Faster R-CNN [34] and R-FCN [5] replace the region proposal step with lightweight networks to deliver a complete end-to-end system. Several attempts have been performed to boost the performance of the detector, including feature pyramid [23][21], multi-scale [38][28], and object relation [16]. The most recent related works are Cascade R-CNN [4] and IoU-Net [18]. Cascade R-CNN gradually increases qualified proposal towards high quality detection with cascade sub-networks. IoU-Net learns to predict the IoU between each detected bounding box and the matched ground-truth as the localization confidence.

One-stage Detectors: OverFeat [35] is one of the first modern one-stage object detector based on deep networks. SSD [26] and YOLO [31] have renewed interest in one-stage methods. The methods skip the region proposal generation step and predict bounding boxes and detection confidences of multiple categories directly. One stage detectors are applied over a regular, dense sampled locations, scales, and aspect ratios, and rely on the fully ConvNets to predict the objects on each localization. The main advantage of this is its high computational efficiency [32]. Due to the foreground-background class imbalance problem, previous one stage object detectors trailed in small scale object detection and larger compute budget. Recently, RetinaNet [24] is able to match the accuracy of the two-stage object detectors, with the proposed Focal Loss and dense prediction.

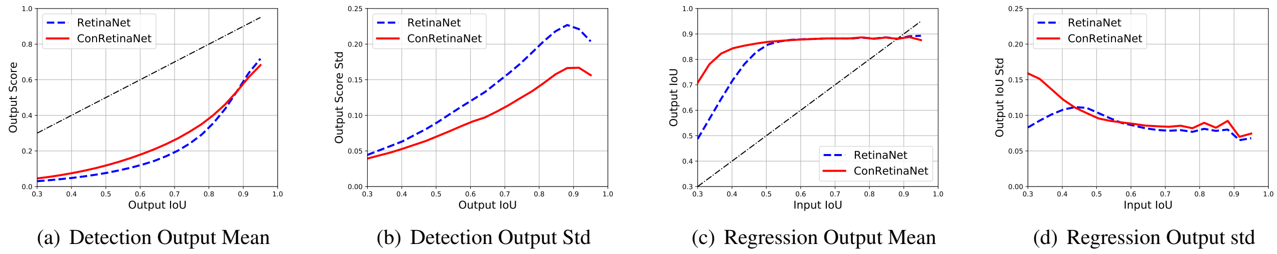
Improving One-stage Detectors: There are many works trying to improve the one stage object detectors, including better feature pyramid construction [10][19], multi-stage refinement [44][42], adaptive anchors [43] and usage of corner keypoints [22]. The most related works are BPN [42] and RefineDet [44]. Both of them try to refine the detection box with new branches of predictions. In this work, we find that given the feature pyramid representations of an image, the key bottleneck for the performance is the train-inference misalignment, and the misalignment could be directly ameliorated by the consistent optimization. Some prior works share similarities with our work, and we will discuss them in more detail in §5.

3. Single Shot Object Detection

In this section, we first review the single shot object detection. Then, we investigate the misalignment in the detector. The consistent optimization solution will be described in §4.

The key idea of the single shot object detector is to associate the pre-defined anchor which is centered at each feature map location with the convolutional operation and results, as shown in Figure 4(a). The single stage detector is composed of a *backbone* network and two *task-specific* sub-networks. The backbone is responsible for computing a convolutional feature map over an entire input image and is an off-the-shelf convolutional network. The first subnet performs convolutional object classification on the backbone's output; the second subnet performs convolutional anchor box regression. In the box sub-network, the optimization goal is to minimize the offsets between the regular anchors and the nearby ground-truth boxes, if one exists (usually utilizing the Smooth L_1 loss). For each of the anchors per spatial location, these 4 outputs predict the relative offset

[1] Current single stage detectors usually utilize feature pyramid heads to detect multi-scale objects. For clarity, we only show one head.



(a) Detection Output Mean (b) Detection Output Std (c) Regression Output Mean (d) Regression Output std

Figure 5: The detection and localization performance of RetinaNet detector with different IoUs.

between the anchor and the ground-truth box. The classification sub-networks predict the probability of object presence at each spatial position for each of the anchors and object classes.

During inference, each pre-defined anchor is regressed based on the box sub-networks. The score of the the refined box is given by the classification sub-networks. Finally, non-maximum-suppression (NMS) is applied to remove duplicated bounding boxes. In this paper, we take RetinaNet [24] to do case study since it gets promising performance both on efficiency and accuracy.

3.1. The Misalignments

There are two misalignments between training and inference in the current single stage detector. (a) The localization qualities between original anchors and the refined anchors are very different, as shown in Figure 3. Such difference will harm the reliability of the class output scores, since the classification subnet is trained based on the original anchor. Moreover, the inter-class confusion (Figure 2) may lead to the well located box being eliminated, causing inter-class error when scoring a final bounding box. (b) Anchors whose IoU overlap with the groundtruth lower than 0.5 are treated as negative samples. However after regression, some refined anchors also have high overlaps with the groundtruth. Such samples will be eliminated due to the small class scores, causing foreground-background classification error.

Figure 5 shows the detection and localization performance of RetinaNet [24] with different IoUs. As shown in Figure 5(a), the output IoUs and the prediction confidences have a significant positive correlation. However, the variations are increasing dramatically as the output IoU increases (Figure 5(b)). That means the detector is more robust for the negative samples than the positive samples, thanks to the utilization of Focal Loss. However, the whole detection performance is evaluated as a function of IoU threshold, as in MS COCO [25]. The robustness of the positive samples is also import for object detection, since the score confidence order of the samples plays an decisive role for an accurate detector, not just filtering the negative samples. We believe the high variations are caused by the training-inference inconsistency as analyzed above.

We also visualize the localization performance of RetinaNet. The localization performance is evaluated as a function of the input IoUs. Figure 5(c) shows the average output IoUs while Figure 5(d) visualizes the variations. What surprises us is that the regressor performs pretty well when the input IoUs are larger (for example, input IoU ≥ 0.5). One can infer that RetinaNet seems to produce more tight boxes, same as that suggested by [29]. Utilizing the regressed anchors are more accurate for the training of a high quality object detector.

4. Consistent Optimization

Consistent Detection: The loose training signal for the classification sub-networks has hindered the accuracy of the detector, since the behaviors between the training anchors and refined anchors at prediction phase are different. In this work, the solution is simple: attaching subsequent classification targets for the regressed anchors. The classification target becomes

$$L_{cls} = \frac{1}{N_{cls}} \sum_i [L_{cls}(c_i, c_i^*) + \alpha L_{cls}(c_i, c_i^\dagger)]. \quad (1)$$

Here, i is the index of an anchor in a mini-batch and c_i is the predicted probability of the (refined) anchor i being an object. The ground-truth label c_i^* is the label for the original anchor i, while $c_i^\dagger$ is the label for the refined one. N_{cls} is the mini-batch size for the classification branch. α balances the weights between two terms. We find that directly training the model with the refined anchors already gets superior performance. Combining the two loss together makes the training process more stable, and does not harm the performance.

Consistent Localization As shown in Figure 5(c), RetinaNet seems to produce tight boxes, which is different from the observations of two-stage object detectors [4][29]. To keep consistency with the classification branch, we also add the subsequent regressions. The localization loss function becomes

$$L_{reg} = \frac{1}{N_{reg}^0} \sum_i L_{reg}^0(t_i^0, t_i^*) + \frac{1}{N_{reg}^1} \sum_i L_{reg}^1(t_i^1, t_i^\dagger), \tag{2}$$

where t_i^0 is the predicted offset of the original anchor i, t_i^1 the predicted offset of the refined one. t^* and $t^\dagger$ are corresponding groundtruch offsets for the original and refined anchor. N_{reg} is the mini-batch size for the regression branch. More details about the localization loss are referred to [34][12].

Implementation Choices: Given the consistent optimization functions, there are also several implementation choices. Two typical implementations are shown in Figure 4(b) and Figure 4(c). In Figure 4(b), we add new sibling branches for the subsequent localization and classification, denoted as cascade version. The design of the new subnet is identical to the existing subnet, and the parameters are not shared, motivated from Cascade R-CNN [4]. In Figure 4(c), the feature head in the previous stage is also combined to enrich the feature for the subsequent prediction, denoted as gated cascade version, which is similar to some recent works that encode more semantic features for further refinement of the detector [42][44]. Finally, the proposed version is shown in Figure 4(d), which does not require more computational branches. The implementation is simple: in parallel with the original classification and localization loss, we add the consistent terms of Equation 1 and 2. The input features are shared between different terms[2]. During inference, the confidence score are generated exactly the same way as the traditional detector.

In the experiments, we show that the implementation of Figure 4(d) get comparable results compared with other variants. More importantly, it requires almost no additional computation during training and inference, and is easier to converge.

5. Discussion and Comparison to Prior Works

Here, we compare the consistent optimization with several existing object detectors, and point out key similarities and differences between them.

Cascade R-CNN Cascade R-CNN [4] consists of a sequence of detectors trained with increasing IoU thresholds, to be sequentially more selective against close false positives. It's straightforward to extend the cascade idea into one stage detector. However, the main challenge is that the one stage detectors only rely on the convolutional kernels to associate the anchors and final predictions. In contrast, ROI-Pooling in region based detectors makes the feature extraction of the subsequent detector heads easier. The sample distributions between two-stage and one-stage detectors are also different.

RefineDet There are same previous works trying to improve the performance of single stage detectors by utilizing several stages. In RefineDet [44], the authors introduce the anchor refinement module to (1) filter out negative anchors to reduce search space for the classifier, and (2) coarsely adjust the locations. The idea is very like the the cascade manner. There are two main differences: (a) The anchor refinement module in RefineDet plays the role of the first stage (or RPN stage) in Faster R-CNN [34]. It predicts a coarse location for each anchor. Our solution is to make the final prediction more reliable by utilizing consistent optimization. (b) The RefineDet relies on a transfer connection block to transfer the features in the anchor refinement module to the object detection module. From our observation, adding more parameters is not necessary under standard feature pyramid networks [23]. The main performance bottleneck is the misalignment between optimization and prediction. We also conduct experiments to verify this assumption.

IoU-Net Recently, IoU-Net [18] proposes to learn the IoU between the predicted bounding box and the ground truth bounding box. IoU-NMS is then applied to the detection boxes, guided by the learned IoU. The goal of IoU-Net is to make the confidence score of the box to be consistent with the localization performance. IoU-Net shows its effectiveness on the two-stage detectors via jittered RoI training. We believe it could also be utilized in the one-stage detectors, which is beyond the scope of this paper.

6. Experiments

We present experimental results on the bounding box detection track of the challenging MS COCO benchmark [25]. For training, we follow common practice [12] and use the MS COCO `trainval35k` split (union of 80k images from `train` and a random 35k subset of images from the 40k image `val` split). If not specified, we report studies by evaluating on the `minival5k` split. The COCO-style Average Precision (AP) averages AP across IoU thresholds from 0.5 to 0.95 with an interval of 0.05. These metrics measure the detection performance of various qualities. Final results are also reported on the `test-dev` set.

6.1. Implementation Details

All experiments are implemented with Caffe2 on Detectron [12] codebase for fair comparison. End-to-end training is used for simplicity. We replace the traditional classification and localization loss function with Equation 1 and 2, unless otherwise noted. The optimization targets are all based on the refined anchors after the anchor regression. We set $\alpha = 1$ and it works well for all experiments. During training, no data augmentation is used except standard hor-

[2] Only the last layer's parameters of the bounding box regression are not shared.

izontal image flipping . Inference is performed on a single image scale, with no further bells and whistles.

6.2. Baseline

To valid the effectiveness of the consistent optimization, we conduct experiments based on the most recently proposed RetinaNet [24]. RetinaNet is able to match the speed of previous one-stage detectors while get comparable accuracy compared with existing state-of-the-art two-stage detectors. We use data parallel synchronized SGD over 4 GPUs with a total of 8 images per mini-batch (2 images per GPU). Unless otherwise specified, the experimental settings are exactly the same as that in [12]. For all ablation studies we use an image scale with short side 500 pixels for training and testing using ResNet-50 with a Feature Pyramid Network (FPN) [23] constructed on top. More model setting details are referred to [24]. We conduct ablation studies and analyze the behavior of the consistent optimization with various design choices.

6.3. Ablation Study

Comparison with different design choices: We first compare the design choices of the consistent optimization. The cascade and gated cascade version can be seen as cascade extensions of RetinaNet. Table 1 shows the performances on MS COCO dataset. To our surprise, using new features does not help boosting the performance, even we combine the previous prediction head. The last implementation in Figure 4 enjoys both accuracy gains and efficiency.

	AP	AP_{50}	AP_{75}
RetinaNet baseline	32.5	50.9	34.8
+ Cascade	33.4	51.3	36.0
+ Gated Cascade	33.6	51.5	36.3
+ Consistent Only	33.7	51.7	36.2

Table 1: The impact of the different design choices.

In the cascade setting, we rely the new branches to fit the data after the regression, like Cascade R-CNN [4] does. However, the position of anchor box has been changed after the first regression. In Cascade R-CNN, the authors utilize the ROI-Pooling to extract more accurate features. In the single stage detectors, it is hard for the convolutional networks to learn such transformations in-place. We also tried to use the deformable convolution [6] to learn the transformations, but failed. This experiment demonstrates that the improvements come from better training of the detector, not more parameters or architecture designs.

The Hyper-parameters: The hyper-parameters in the consistent terms are the IoU thresholds μ_{pos} and μ_{neg}, which define the positive and negative samples. We conduct several experiments to compare the sensitivity of the hyper-parameters, as shown in Table 2. The detection performance of the model is robust to the hyper-parameters. Higher μ_{pos} gets slight better performance on strict IoU thresholds. However, the whole performances are similar. We use the setting of $\mu_{pos} = 0.6$ and $\mu_{neg} = 0.5$ for all other experiments in this work.

μ_{pos}	μ_{neg}	AP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
0.5	0.5	33.5	51.7	47.5	40.7	29.4	11.2
0.6	0.5	33.7	51.7	47.6	41.0	30.0	13.6
0.7	0.6	33.6	51.4	47.6	40.9	30.2	14.0

Table 2: The impact of the IoU threshold μ_{pos} and μ_{neg} on detection performance.

More Stages? Like Cascade R-CNN, we add more stages of classifications and localizations to compare the influence. The impact of the number of stages is summarized in Table 3. Adding consistent classification loss significantly improves the baseline detector (+0.8 AP). Two classification and regression terms get the best performance. Adding more regression stages leads to a slight performance decrease, while including more classification terms leads to no improvement. When we perform anchor regression once using the model trained by consistent classification and localization, the improvements are also significant (+1.0 AP). That means the consistent localization helps training the original anchor to predict the groundtruth location.

#cls	#reg	AP	AP_{50}	AP_{60}	AP_{70}	AP_{80}	AP_{90}
1	1	32.5	50.6	46.4	39.5	28.6	10.4
2	1	33.3	51.2	47.4	40.6	28.9	11.2
2	2*	33.5	51.3	47.4	40.8	29.8	11.8
2	2	33.7	51.7	47.6	41.0	30.0	13.6
3	2	33.6	51.7	47.7	41.0	29.6	13.4

Table 3: The impact of the number of stages on detection performance. $\mu_{pos}^2 = 0.6$, $\mu_{neg}^2 = 0.5$, $\mu_{pos}^3 = 0.7$, $\mu_{neg}^3 = 0.6$ in this experiment setting. The first row is the baseline model result. "*" means performing anchor regression once during inference.

Both consistent classification and localization improve the performance at different IoU thresholds. The gap of 1.2 AP, shows the effectiveness of the consistent optimization for training the single shot detector. Different from previous works that always perform better for strict IoU thresholds [4][18], consistent optimization enjoys gains for all localization qualities.

backbone	scale	consistent	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50	500	✗	32.5	50.9	34.8	13.9	35.8	46.7
ResNet-50	500	✓	33.7	51.7	36.2	14.6	37.7	49.6
ResNet-50	800	✗	35.7	55.0	38.5	18.9	38.9	46.3
ResNet-50	800	✓	36.4	55.4	39.2	20.0	39.6	49.7
ResNet-101	500	✗	34.4	53.1	36.8	14.7	38.8	49.1
ResNet-101	500	✓	35.5	53.5	38.3	16.2	39.8	52.0
ResNet-101	800	✗	37.8	57.5	40.8	20.2	41.1	49.2
ResNet-101	800	✓	38.7	58.2	41.7	21.4	42.6	52.2

Table 4: Detailed comparison on different resolutions and model capacities.

Parameter and Timing: The number of the parameters are almost not increased with consistent optimization, at both training and inference phases. During training the consistent loss needs to compute the classification and localization targets of the refined boxes. At inference, we only add the two-stage regression which is implemented by a single convolutional layer. Recent works on boosting one stage detectors [44][42] or two stage detectors [16][18][4] mostly rely on more parameters compared with the baseline models.

6.4. Generalization Capacity

Across model depth and scale: To validate the generalization capacity of consistent optimization, we conduct experiments across model depth and input scale based on RetinaNet [24]. As shown in Table 4, the proposed method improves on these baselines consistently by ∼1.0 point, independently of model capacities and input scales. These results suggest that the consistent optimization is widely applicable within one stage object detector. When analysing the performance on small, medium and large object scales, we observe that most improvements come from the larger objects.

Results on SSD: We further do experiments under Single Shot MultiBox Detector (SSD) [26] baseline to validate its generalization capacity. SSD is built on top of a "base" network that ends with some convolutional layers. Each of the added layers, and some of the earlier base network layers are used to predict scores and offsets for the predefined anchor boxes. The experiment is conducted based on re-implementation of SSD512 using PyTorch [30], more details are referred to [26]. We use the VGG-16 [37] and ResNet-50 [15] models pretrained on the `ImageNet1k` as the start models[3], and fine-tune them on the MS COCO dataset.

Table 5 shows the comparison results when adding consistent loss on SSD baseline. The first row is the results reported in the paper. Others are our re-implementation results using PyTorch. The detection results show that the consistent optimization also has significant improvements over the popular SSD architecture. These reinforce our belief on the generalization capacity of the consistent optimization.

backbone	consistent	AP	AP_{50}	AP_{75}
VGG-16 [26]	✗	28.8	48.5	30.3
VGG-16	✗	28.9	47.9	30.6
VGG-16	✓	30.5	49.6	31.7
ResNet-50	✗	30.6	50.0	32.2
ResNet-50	✓	31.9	51.0	33.8

Table 5: The impact of the consistent optimization on SSD detector.

6.5. Comparison to State of the Art

The consistent optimization extension of RetinaNet, is compared to state-of-the-art object detectors (both one-stage and two-stage) in Table 6. We report the standard COCO metrics including AP (averaged over IoU thresholds), AP_{50}, AP_{75}, and AP_S, AP_M, AP_L (AP at different scales) on the `test-dev` set. The experimental settings are described in §6.1.

The first group of detectors on Table 6 are two-stage detectors, the second group one-stage detectors, and the last group the consistent optimization extension of RetinaNet. The extension from RetinaNet to ConRetinaNet improves detection performance by ∼1 point, and it also **outperforms all single-stage detector under ResNet-101 backbone, under all evaluation metrics**. This includes the very recent one-stage RefineDet [44] and two-stage relation networks [16]. Specifically, ConRetinaNet-ResNet-50 outperforms DSSD-ResNet-101 [10] and RefineDet-ResNet-101 [44] with large margins. We also compare the single model ConRetinaNet with CornerNet [22], which uses heavier Hourglass-104 and more training and testing time data augmentations.

[3] https://github.com/pytorch/vision

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
two-stage							
Faster R-CNN+++[15]*	ResNet-101	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN by G-RMI[17]	Inception-ResNet-v2	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w FPN[23]	ResNet-101	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN w TDM[36]	Inception-ResNet-v2	36.8	57.7	39.2	16.2	39.8	52.1
Deformable R-FCN [6]*	Aligned-Inception-ResNet	37.5	58.0	40.8	19.4	40.1	52.5
Mask R-CNN[13]	ResNet-101	38.2	60.3	41.7	20.1	41.1	50.2
Relation[16]	DCN-101	39.0	58.6	42.9	-	-	-
Regionlets [16]	ResNet-101	39.3	59.8	-	21.7	43.7	50.9
DeNet768 [39]	ResNet-101	39.5	58.0	42.6	18.9	43.5	54.1
IoU-Net[18]	ResNet-101	40.6	59.0	-	-	-	-
Cascade R-CNN[4]	ResNet-101	42.8	62.1	46.3	23.7	45.5	55.2
Faster R-CNN by MSRA[45]	DCN-v2-101	44.0	65.9	48.1	23.2	47.7	59.6
one-stage							
YOLOv2 [31]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
RON384 [20]*	VGG-16	27.4	49.5	27.1	-	-	-
SSD513 [10]	ResNet-101	31.2	50.4	33.3	10.2	34.5	49.8
YOLOv3 [33]	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9
DSSD513 [10]	ResNet-101	33.2	53.3	35.2	13.0	35.4	51.1
RefineDet512 [44]	ResNet-101	36.4	57.5	39.5	16.6	39.9	51.4
RetinaNet [24]	ResNet-101	39.1	59.1	42.3	21.8	42.7	50.2
CornerNet511[22]*	Hourglass-104	40.5	56.5	43.1	19.4	42.7	53.9
ours							
ConRetinaNet	ResNet-50	37.3	56.4	40.3	21.3	40.2	51.0
ConRetinaNet	ResNet-101	40.1	59.6	43.5	23.4	44.2	53.3

Table 6: Object detection single-model results (bounding box AP) *v.s.* state-of-the-art on COCO `test-dev`. We show results for our ConRetinaNet-50 and ConRetinaNet-101 models with 800 input scale. Both RetinaNet and ConRetinaNet are trained with scale jitter and for 1.5× longer than the same model from Table 4. Our model achieves top results, outperforming most one-stage and two-stage models. The entries denoted by "*" used bells and whistles at inference.

We note that the Cascade R-CNN [4] and Faster R-CNN based on Deformbale ConvNets v2 [45] show better accuracy than ConRetinaNet. The difficulty to bring the cascade idea to the single stage detectors is how to associate the refined anchors with the corresponding features. An intersecting direction is to utilize the region based branch to get more accurate features. The deformable convolution shows better capability to model the geometric transformation of the objects. Replacing the backbone networks with Deformable ConvNets is supposed to get better performance, which is beyond the focus of this paper. At last, Cascade R-CNN and Deformbale ConvNets both require more parameters to get such results.

7. Conclusion and Future Work

In this paper, we propose the simple and effective consistent optimization to boost the performance of single stage object detectors. By examination of the model behaviors, we find that the optimization misalignment between training and inference is the bottleneck to get better results. We conduct extensive experiments to compare different model design choices, and demonstrate that the consistent optimization is the most important factor. Utilizing consistent optimization requires almost no additional parameters, and it shows its effectiveness using the strong RetinaNet baseline on challenging MS COCO dataset.

For the future work, we will try to combine the localization confidence which is proposed in [18] and consistent optimization to further improve the detector's quality. Another important direction is to associate the refined boxes and their corresponding features with geometric transformation on the feature maps.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 73–80. IEEE, 2010.

[2] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Proceed-*

ings of the IEEE conference on computer vision and pattern recognition, pages 328–335, 2014.

[3] H. Azizpour and I. Laptev. Object detection using strongly-supervised deformable part models. In *European Conference on Computer Vision*, pages 836–849. Springer, 2012.

[4] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. *arXiv preprint arXiv:1712.00726*, 2017.

[5] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[6] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 764–773, 2017.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[8] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, 2014.

[9] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[10] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

[11] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[12] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He. Detectron. https://github.com/facebookresearch/detectron, 2018.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer, 2014.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei. Relation networks for object detection. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2018.

[17] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, volume 4, 2017.

[18] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision, Munich, Germany*, pages 8–14, 2018.

[19] T. Kong, F. Sun, W. Huang, and H. Liu. Deep feature pyramid reconfiguration for object detection. In *European Conference on Computer Vision*, pages 172–188. Springer, 2018.

[20] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen. Ron: Reverse connection with objectness prior networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 2, 2017.

[21] T. Kong, A. Yao, Y. Chen, and F. Sun. Hypernet: Towards accurate region proposal generation and joint object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 845–853, 2016.

[22] H. Law and J. Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 6, 2018.

[23] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[25] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[26] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[27] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[28] M. Najibi, B. Singh, and L. S. Davis. Autofocus: Efficient multi-scale inference. *arXiv preprint arXiv:1812.01600*, 2018.

[29] K. Oksuz, B. C. Cam, E. Akbas, and S. Kalkan. Localization recall precision (lrp): A new performance metric for object detection. In *European Conference on Computer Vision (ECCV)*, volume 6, 2018.

[30] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[32] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint*, 2017.

[33] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[34] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[35] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[36] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta. Beyond skip connections: Top-down modulation for object detection. *arXiv preprint arXiv:1612.06851*, 2016.

[37] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[38] B. Singh and L. S. Davis. An analysis of scale invariance in object detection–snip. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3578–3587, 2018.

[39] L. Tychsen-Smith and L. Petersson. Improving object localization with fitness nms and bounded iou loss. *arXiv preprint arXiv:1711.00164*, 2017.

[40] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders. Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1879–1886. IEEE, 2011.

[41] X. Wang, T. X. Han, and S. Yan. An hog-lbp human detector with partial occlusion handling. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 32–39. IEEE, 2009.

[42] X. Wu, D. Zhang, J. Zhu, and S. C. Hoi. Single-shot bidirectional pyramid networks for high-quality object detection. *arXiv preprint arXiv:1803.08208*, 2018.

[43] T. Yang, X. Zhang, Z. Li, W. Zhang, and J. Sun. Metaanchor: Learning to detect objects with customized anchors. In *Advances in Neural Information Processing Systems*, pages 318–328, 2018.

[44] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li. Single-shot refinement neural network for object detection. *arXiv preprint*, 2017.

[45] X. Zhu, H. Hu, S. Lin, and J. Dai. Deformable convnets v2: More deformable, better results. *arXiv preprint arXiv:1811.11168*, 2018.

[46] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *European conference on computer vision*, pages 391–405. Springer, 2014.