Revisiting the Sibling Head in Object Detector

Guanglu Song[1] Yu Liu[2]* Xiaogang Wang[2]
[1]SenseTime X-Lab
[2]The Chinese University of Hong Kong, Hong Kong
[1]songguanglu@sensetime.com, [2]{yuliu, xgwang}@ee.cuhk.edu.hk

Abstract

The "shared head for classification and localization" (sibling head), firstly denominated in Fast RCNN [9], has been leading the fashion of the object detection community in the past five years. This paper provides the observation that the spatial misalignment between the two object functions in the sibling head can considerably hurt the training process, but this misalignment can be resolved by a very simple operator called task-aware spatial disentanglement *(TSD). Considering the classification and regression, TSD decouples them from the spatial dimension by generating two disentangled proposals for them, which are estimated by the shared proposal. This is inspired by the natural insight that for one instance, the features in some salient area may have rich information for classification while these around the boundary may be good at bounding box regression. Surprisingly, this simple design can boost all backbones and models on both MS COCO and Google OpenImage consistently by ∼3% mAP. Further, we propose a progressive constraint to enlarge the performance margin between the disentangled and the shared proposals, and gain ∼1% more mAP. We show the TSD breaks through the upper bound of nowadays single-model detector by a large margin (**mAP 49.4 with ResNet-101, 51.2 with SENet154**), and is the core model of our **1st place solution on the Google OpenImage Challenge 2019**.*

1. Introduction

Since the breakthrough of object detection performance has been achieved by seminal R-CNN families [10, 9, 30] and powerful FPN [21], the subsequent performance enhancement of this task seems to be hindered by some concealed bottlenecks. Even the advanced algorithms bolstered by AutoML [8, 38] have been delved, the performance gain is still limited to an easily accessible improvement range. As the most obvious distinction from the generic object classification task, the specialized sibling head for both classification and localization comes into focus and is widely used in most of advanced detectors including single stage family [25, 33, 12], two-stage family [5, 18, 40, 26, 19] and anchor free family [17]. Considering the two different tasks share almost the same parameters, a few works become conscious about the conflict between the two object functions in the sibling head and try to find a trade-off way.

*Corresponding author

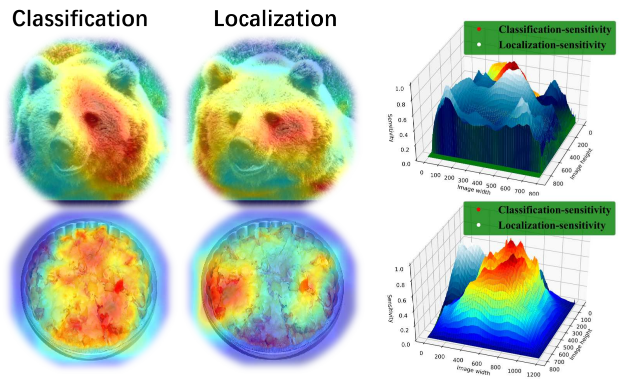


Figure 1. Illustration of the task spatial misalignment. The first column is the sensitive location for classification and the second column is the sensitive location for localization. The third column is the 3D visualization of the sensitivity distribution.

IoU-Net [15] is the first to reveal this problem. They find the feature which generates a good classification score always predicts a coarse bounding box. To handle this problem, they first introduce an extra head to predict the IoU as the localization confidence, and then aggregate the localization confidence and the classification confidence together to be the final classification score. This approach does reduce the misalignment problem but in a compromise manner – the essential philosophy behind it is relatively raising the confidence score of a tight bounding box and reduce the score of a bad one. The misalignment still exists in each spatial point. Along with this direction, Double-Head R-CNN [35] is proposed to disentangle the sibling head into two specific branches for classification and localization, respectively. Despite of elaborate design of each branch, it can be deemed to disentangle the information by adding a new branch, essentially reduce the shared parameters of the two tasks. Although the satisfactory performance can be obtained by this detection head disentanglement, conflict between the two tasks still remain since the features fed into the two branches are produced by ROI Pooling from the same proposal.

In this paper, we meticulously revisit the sibling head in the anchor-based object detector to seek the essence of the tasks misalignment. We explore the spatial sensitivity of classification and localization on the output feature maps of each layer in the feature pyramid of FPN. Based on the commonly used sibling head (a fully connected head *2-fc*), we illustrate the spatial sensitive heatmap in Figure.1. The first column is the spatial sensitive heatmap for classification and the second column is for localization. The warmer the better for the color. We also show their 3D visualizations in the third column. It's obvious that for one instance, the features in some salient areas may have rich information for classification while these around the boundary may be good at bounding box regression. This essential tasks misalignment in spatial dimension greatly limits the performance gain whether evolving the backbone or enhancing the detection head. In other words, if a detector try to infer the classification score and regression result from a same spatial point/anchor, it will always get an imperfect trade-off result.

This significant observation motivates us to rethink the architecture of the sibling head. The optimal solution for the misalignment problem should be explored by the spatial disentanglement. Based on this, we propose a novel operator called *task-aware spatial disentanglement* (TSD) to resolve this barrier. The goal of TSD is to **spatially disentangle the gradient flows of classification and localization**. To achieve this, TSD generates two disentangled proposals for these two tasks, based on the original proposal in classical sibling head. It allows two tasks to adaptively seek the optimal location in space without compromising each other. With the simple design, the performance of all backbones and models on both MS COCO and Google OpenImage are boosted by $\sim$3% mAP. Furthermore, we propose a *progressive constraint* (PC) to enlarge the performance margin between TSD and the classical sibling head. It introduces the hyper-parameter *margin* to advocate the more confident classification and precise regression. $\sim$1% more mAP is gained on the basis of TSD. Whether for variant backbones or different detection frameworks, the integrated algorithms can steadily improve the performance by $\sim$4% and even $\sim$6% for lightweight MobileNetV2. Behind the outstanding performance gains, only a slight increased parameter is required, which is negligible for some heavy backbones.

To summarize, the contributions of this paper are as follows:

1) We delve into the essential barriers behind the tangled tasks in RoI-based detectors and reveal the bottlenecks that limit the upper bound of detection performance.

2) We propose a simple operator called *task-aware spatial disentanglement* (TSD) to deal with the tangled tasks conflict. Through the task-aware proposal estimation and the detection head, it could generate the task-specific feature representation to eliminate the compromises between classification and localization.

3) We further propose a *progressive constraint* (PC) to enlarge the performance margin between TSD and the classical sibling head.

4) We validate the effectiveness of our approach on the standard COCO benchmark and large-scale OpenImage dataset with thorough ablation studies. Compared with the state-of-the-art methods, our proposed method achieves the **mAP of 49.4 using a single model with ResNet-101 backbone and mAP of 51.2 with heavy SENet154.**

2. Methods

In this section, we first describe the overall framework of our proposed *task-aware spatial disentanglement* (TSD), then detail the sub-modules in Sec. 2.2 and 2.3. Finally, we delve into the inherent problem in sibling head and demonstrate the advantage of TSD.

2.1. TSD

As shown in Figure.2 (a), denote a rectangular bounding box proposal as P and the ground-truth bounding box as $\mathcal{B}$ with class y, the classical Faster RCNN [30] aims to minimize the classification loss and localization loss based on the shared P:

$$\mathcal{L} = \mathcal{L}_{cls}(\mathcal{H}_1(F_l, P), y) + \mathcal{L}_{loc}(\mathcal{H}_2(F_l, P), \mathcal{B}) \quad (1)$$

where $\mathcal{H}_1(\cdot) = \{f(\cdot), \mathcal{C}(\cdot)\}$ and $\mathcal{H}_2(\cdot) = \{f(\cdot), \mathcal{R}(\cdot)\}$. $f(\cdot)$ is the feature extractor and $\mathcal{C}(\cdot)$ and $\mathcal{R}(\cdot)$ are the functions for transforming feature to predict specific category and localize object. Seminal work [35] thinks the shared f for classification and localization is not optimal, and they disentangle it to f_c and f_r for classification and regression, respectively. Although the appropriate head-decoupling brings a reasonable improvement, the inherent conflict caused by the tangled tasks in the spatial dimension is still lurking.

For this potential problem, our goal is to alleviate the inherent conflict in sibling head by disentangling the tasks from the spatial dimension. We propose a novel TSD head for this goal as shown in Figure 2. In TSD, the Eq. 1 can be written as:

$$\mathcal{L} = \mathcal{L}^D_{cls}(\mathcal{H}^D_1(F_l, \hat{P}_c), y) + \mathcal{L}^D_{loc}(\mathcal{H}^D_2(F_l, \hat{P}_r), \mathcal{B}) \quad (2)$$

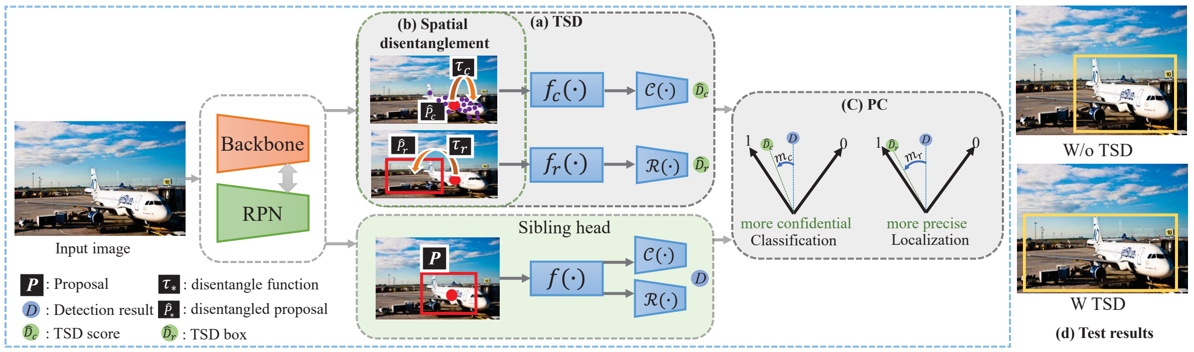


Figure 2. Illustration of the proposed TSD cooperated with Faster RCNN [30]. Input images are first fed into the FPN backbone and then, region proposal P is generated by RPN. TSD adopts the RoI feature of P as input and estimates the derived proposals $\hat{P}_c$ and $\hat{P}_r$ for classification and localization. Finally, two parallel branches are used to predict specific category and regress precise box, respectively.

where disentangled proposals $\hat{P}_c = \tau_c(P, \Delta C)$ and $\hat{P}_r = \tau_r(P, \Delta R)$ are estimated from the shared P. ΔC is a pointwise deformation of P and ΔR is a proposal-wise translation. In TSD, $\mathcal{H}_1^D(\cdot) = \{f_c(\cdot), \mathcal{C}(\cdot)\}$ and $\mathcal{H}_2^D(\cdot) = \{f_r(\cdot), \mathcal{R}(\cdot)\}$.

In particular, TSD tasks the RoI feature of P as input, and then generates the disentangled proposals $\hat{P}_c$ and $\hat{P}_r$ for classification and localization, respectively. Different tasks can be disentangled from the spatial dimension via the separated proposals. The classification-specific feature maps $\hat{F}_c$ and localization-specific feature maps $\hat{F}_r$ can be generated through parallel branches. In the first branch, $\hat{F}_c$ is fed into a three-layer fully connected networks for classification. In the second branch, the RoI feature $\hat{F}_r$ corresponding to derived proposal $\hat{P}_r$ will be extracted and fed into a similar architecture with the first branch to perform localization task. By disentangling the shared proposal for the classification and localization, TSD can learn the task-aware feature representation adaptively. TSD is applicable to most existing RoI-based detectors. As the training procedure adopts an end-to-end manner cooperated with the well-designed progressive constraint (PC), it is robust to the change of backbones and input distributions (e.g., training with different datasets.).

2.2. Task-aware spatial disentanglement learning

Inspired by Figure.1, we introduce the *task-aware spatial disentanglement* learning to alleviate the misalignment caused by the shared spatial clues. As shown in Figure.2 (b), define the RoI feature of P as F, we embed the deformation-learning manner into TSD to achieve this goal. For localization, a three-layer fully connected network $\mathcal{F}_r$ is designed to generate a proposal-wise translation on P to produce a new derived proposal $\hat{P}_r$. This procedure can be formulated as:

$$\Delta R = \gamma \mathcal{F}_r(F; \theta_r) \cdot (w, h) \tag{3}$$

where $\Delta R \in \mathbb{R}^{1\times1\times2}$ and the output of $\mathcal{F}_r$ for each layer is $\{256, 256, 2\}$. γ is a pre-defined scalar to modulate the magnitude of the ΔR and (w, h) is the width and height of P. The derived function $\tau_r(\cdot)$ for generating $\hat{P}_r$ is:

$$\hat{P}_r = P + \Delta R \tag{4}$$

Eq. 4 indicates the proposal-wise translation where the coordinate of each pixel in P will be translated to a new coordinate with the same ΔR. The derived proposal $\hat{P}_r$ only focuses on the localization task and in the pooling function, we adopt the bilinear interpolation the same as [5] to make ΔR differentiable.

For classification, given the shared P, a pointwise deformation on a regular grid $k \times k$ is generated to estimate a derived proposal $\hat{P}_c$ with an irregular shape. For (x,y)-th grid, the translation $\Delta C(x, y, *)$ is performed on the sample points in it to obtain the new sample points for $\hat{P}_c$. This procedure can be formulated as:

$$\Delta C = \gamma \mathcal{F}_c(F; \theta_c) \cdot (w, h) \tag{5}$$

where $\Delta C \in \mathbb{R}^{k\times k\times 2}$. $\mathcal{F}_c$ is a three-layer fully connected network with output $\{256, 256, k \times k \times 2\}$ for each layer and θ_c is the learned parameter. The first layer in $\mathcal{F}_r$ and $\mathcal{F}_c$ is shared to reduce the parameter. For generating feature map $\hat{F}_c$ by irregular $\hat{P}_c$, we adopt the same operation with deformable RoI pooling [5]:

$$\hat{F}_c(x, y) = \sum_{p \in G(x,y)} \frac{\mathcal{F}_B(p_0 + \Delta C(x, y, 1), p_1 + \Delta C(x, y, 2))}{|G(x, y)|} \tag{6}$$

where $G(x, y)$ is the (x,y)-th grid and $|G(x, y)|$ is the number of sample points in it. (p_x, p_y) is the coordinate of the sample point in grid $G(x, y)$ and $\mathcal{F}_B(\cdot)$ is the bilinear interpolation [5] to make the ΔC differentiable.

2.3. Progressive constraint

At the training stage, the TSD and the sibling detection head defined in Eq. 1 can be jointly optimized by $\mathcal{L}_{cls}$ and $\mathcal{L}_{loc}$. Beyond this, we further design the *progressive constraint* (PC) to improve the performance of TSD as shown in Figure.2 (c). For classification branch, PC is formulated as:

$$\mathcal{M}_{cls} = |\mathcal{H}_1(y|F_l, P) - \mathcal{H}_1^D(y|F_l, \tau_c(P, \Delta C)) + m_c|_+ \quad (7)$$

where $\mathcal{H}(y|\cdot)$ indicates the confidence score of the y-th class and m_c is the predefined margin. $|\cdot|_+$ is same as ReLU function. Similarly, for localization, there are:

$$\mathcal{M}_{loc} = |IoU(\hat{\mathcal{B}}, \mathcal{B}) - IoU(\hat{\mathcal{B}}_D, \mathcal{B}) + m_r|_+ \quad (8)$$

where $\hat{\mathcal{B}}$ is the predicted box by sibling head and $\hat{\mathcal{B}}_D$ is regressed by $\mathcal{H}_2^D(F_l, \tau_r(P, \Delta R))$. If P is a negative proposal, M_{loc} is ignored. According to these designs, the whole loss function of TSD with Faster RCNN can be define as:

$$\mathcal{L} = \underbrace{\mathcal{L}_{rpn} + \mathcal{L}_{cls} + \mathcal{L}_{loc}}_{classical\ loss} + \underbrace{\mathcal{L}_{cls}^D + \mathcal{L}_{loc}^D + \mathcal{M}_{cls} + \mathcal{M}_{loc}}_{TSD\ loss} \quad (9)$$

We directly set the loss weight to 1 without carefully adjusting it. Under the optimization of $\mathcal{L}$, TSD can adaptively learn the task-specific feature representation for classification and localization, respectively. Extensive experiments in Sec.3 indicates that disentangling the tangled tasks from the spatial dimension can significantly improve the performance.

2.4. Discussion in context of related works

In this section, we delve into the inherent conflict in tangled tasks. Our work is related to previous works in different aspects. We discuss the relations and differences in detail.

2.4.1 Conflict in sibling head with tangled tasks

Two core designs in classical Faster RCNN are predicting the category for a given proposal and learning a regression function. Due to the essential differences in optimization, classification task requires translation-agnostic property and to the contrary, localization task desires translation-aware property. The specific translation sensitivity property for classification and localization can be formulated as:

$$\begin{aligned} \mathcal{C}(f(F_l, P)) &= \mathcal{C}(f(F_l, P + \varepsilon)), \\ \mathcal{R}(f(F_l, P)) &\neq \mathcal{R}(f(F_l, P + \varepsilon)) \end{aligned} \quad (10)$$

where $\forall \varepsilon, IoU(P + \varepsilon, \mathcal{B}) \geq T$. $\mathcal{C}$ is to predict category probability and $\mathcal{R}$ is the regression function whose output is $(\Delta\hat{x}, \Delta\hat{y}, \Delta\hat{w}, \Delta\hat{h})$. $f(\cdot)$ is the shared feature extractor in classical sibling head and T is the threshold to determine whether P is a positive sample. There are entirely different properties in these two tasks. The shared spatial clues in F_l and feature extractor for these two tasks will become the obstacles to hinder the learning. Different from [35, 15, 5, 43] where the evolved backbone or feature extractor is designed, TSD decouples the classification and regression from spatial dimension by separated $\hat{P}_*$ and $f_*(\cdot)$.

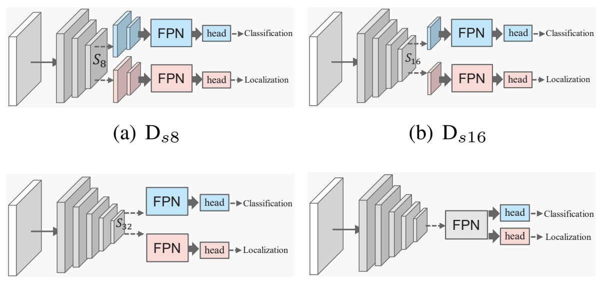
2.4.2 Different from other methods

IoU-Net [15] first illustrates the misalignment between classification and regression. To alleviate this, it directly predicts the IoU to adjust the classification confidence via an extra branch. Unfortunately, this approach does not solve the inherent conflict between tangled tasks. For this same problem, Double-Head R-CNN [35] explores the optimal architectures for classification and localization, respectively. To learn more effective feature representation, DCN [5] with deformable RoI pooling is proposed to extract the semantic information from the irregular region. Whether evolving the backbone or adjusting the detection head, performance can be improved, but the increase is limited.

In this paper, we observe that the essential problem behind the limited performance is the misaligned sensitivity in the spatial dimension between classification and localization. Neither designing better feature extraction methods nor searching for the best architecture can solve this problem. In this dilemma, TSD is proposed to decouple the classification and localization from both the spatial dimension and feature extractor. TSD first performs spatial disentanglement for classification and localization via separated proposals and feature extractors to break the predicament. With the further well-designed PC, it can learn the optimal sensitive location for classification and localization, respectively. Moreover, TSD is still applicable to DCN [5] although deformable RoI pooling in DCN is used to assist in estimating $\hat{F}_c$. By *task-aware spatial disentanglement*, the simple TSD can easily achieve excellent performance for different backbones.

3. Experiments

We perform extensive experiments with variant backbones on the 80-category MS-COCO dataset [23] (object detection and instance segmentation) and 500-category OpenImageV5 challenge dataset [16]. For COCO dataset, following the standard protocol [27], training is performed on the union of 80k *train* images and 35k subset of *val* images and testing is evaluated on the remaining 5k val images (*minival*). We also report results on 20k *test-dev*. For OpenImage dataset, following the official protocol [16], the model is trained on 1,674,979 training images and evaluated



(a) D_{s8} (b) D_{s16}

(c) D_{s32} (d) D_{head}

Figure 3. Ablation studies on variant disentanglement options. (a)-(d) indicate disentangling the detector from stride 8, stride 16, stride 32 and sibling head, respectively.

on the 34,917 val images. The AP$_{.5}$ on public leaderboard is also reported.

3.1. Implementation details

We initialize weights from pre-trained models on ImageNet [31] and the configuration of hyper-parameters follows existing Faster RCNN [30]. Images are resized such that the shorter edge is 800 pixels. The anchor scale and aspect ratio are set to 8 and $\{0.5, 1, 2\}$. We train the models on 16 GPUs (effective mini-batch size is 32) for 13 epochs, with a learning rate warmup strategy [11] from 0.00125 to 0.04 in the first epoch. We decrease the learning rate by 10 at epoch 8 and epoch 11, respectively. RoIAlign [13] is adopted in all experiments, and the pooling size is 7 in both $\mathcal{H}_1^*$ and $\mathcal{H}_2^*$. We use SGD to optimize the training loss with 0.9 momentum and 0.0001 weight decay. No data augmentations except standard horizontal flipping are used. Synchronized BatchNorm mechanism [29, 11] is used to make multi-GPU training more stable. At the inference stage, NMS with 0.5 IoU threshold is applied to remove duplicate boxes. For experiments in the OpenImage dataset, class-aware sampling is used.

3.2. Ablation studies

In this section, we conduct detailed ablation studies on COCO *minival* to evaluate the effectiveness of each module and illustrate the advance and generalization of the proposed TSD. m_c and m_r are set to 0.2 in these experiments.

Task-aware disentanglement. When it comes to tangled tasks conflict in sibling detection head, it's natural to think about decoupling different tasks from the backbone or detection head. To evaluate these ideas, we conduct several experiments to illustrate the comparison between them. As shown in Figure.3, we design different decoupling options including backbone disentanglement and head disentanglement. Detailed performance is shown in Table.1. Decoupling the classification and localization from the backbone largely degrades the performance. It clearly shows that the

Disentanglement	#param	AP	AP$_{.5}$	AP$_{.75}$
ResNet-50	41.8M	36.1	58.0	38.8
ResNet-50+D$_{s8}$	81.1M	22.3	46.3	16.7
ResNet-50+ D$_{s16}$	74.0M	22.0	46.2	16.3
ResNet-50+ D$_{s32}$	59M	20.3	44.7	13.2
ResNet-50+ D$_{head}$	55.7M	37.3	59.4	40.2
TSD w/o PC	58.9M	**38.2**	**60.5**	**41.1**

Table 1. Detailed performance and #parameter of different disentanglement methods.

semantic information in the backbone should be shared by different tasks. As expected, the task-specific head can significantly improve the performance. Compared with D$_{head}$, TSD w/o PC can further enhance the AP with the slight increased parameters, even for the demanding AP$_{.75}$. When faced with heavy backbones, a slight increased parameter is trivial but can still significantly improve the performance. This also substantiates the discussion in Sec. 2.4.1 that disentangling the tasks from spatial dimension can effectively alleviate the inherent conflict in sibling detection head.

Method	AP	AP$_{.5}$	AP$_{.75}$
TSD w/o PC	38.2	60.5	41.1
+ Joint training with sibling head $\mathcal{H}_*$	39.7	61.7	42.8

Table 2. Result of joint training with sibling $\mathcal{H}_*$. The ResNet-50 with FPN is used as the basic detector.

Joint training with sibling head $\mathcal{H}_*$. In TSD, the shared proposal P can also be used to perform classification and localization in an extra sibling head. We empirically observe that the training of sibling head is complementary to the training of TSD, and the results are demonstrated in Table.2. This indicates that the derived proposals $\hat{P}_c$ and $\hat{P}_r$ are not conflict with the original proposal P. At the inference stage, only the TSD head is retained.

Method	TSD	PC		AP	AP$_{.5}$	AP$_{.75}$
		$\mathcal{M}_{cls}$	$\mathcal{M}_{loc}$			
ResNet-50	✓			39.7	61.7	42.8
ResNet-50	✓	✓		40.1	61.7	43.2
ResNet-50	✓		✓	40.8	61.7	43.8
ResNet-50	✓	✓	✓	41.0	61.7	44.3

Table 3. Ablation studies on PC. All of the experiments is joint training with sibling head $\mathcal{H}_*$. m$_c$ and m$_r$ are set to 0.2.

Effectiveness of PC. In Sec. 2.3, we further propose the PC to enhance the performance of TSD. Table.3 reports the detailed ablations on it. We find that PC significantly improves the AP$_{.75}$ by 1.5 and AP$_{.5}$ is barely affected. This demonstrates that PC aims to advocate more confidential classification and precise regression for the accurate boxes. Even on the strict testing standards AP (IoU from 0.5:0.95), 1.3 AP gain can also be obtained.

Method	PC	$\hat{P}_c$	$\hat{P}_r$	AP	AP$_{.5}$	AP$_{.75}$
TSD		*Point.w*	-	38.0	60.3	40.89
TSD		*Point.w*	*Point.w*	38.5	60.7	41.7
TSD		*Point.w*	*Prop.w*	38.2	60.5	41.1
TSD	✓	*Prop.w*	*Prop.w*	39.8	60.1	42.9
TSD	✓	*Point.w*	*Point.w*	40.7	61.8	44.4
TSD	✓	*Point.w*	*Prop.w*	41.0	61.7	44.3

Table 4. Results of different proposal learning manners for $\mathcal{H}_*^D$.

Derived proposal learning manner for $\mathcal{H}_*^D$. There are different programmable strategies to generate the derived proposal $\hat{P}_r$ and $\hat{P}_c$ including proposal-wise translation (*Prop.w*) in Eq. 4, pointwise deformation (*Point.w*) such as deformable RoI pooling [5] or the tricky combination of them. To explore the differences of these learning manners, we conduct extensive experiments for COCO *minival* with ResNet-50. Table.4 demonstrates the comparison results. These comparisons illustrate that *Point.w* is beneficial to the classification task and cooperated with PC, *Prop.w* performs a slight advantage on localization. For generating the derived proposals, classification requires the optimal local features without regular shape restrictions and regression requires the maintenance of global geometric shape information.

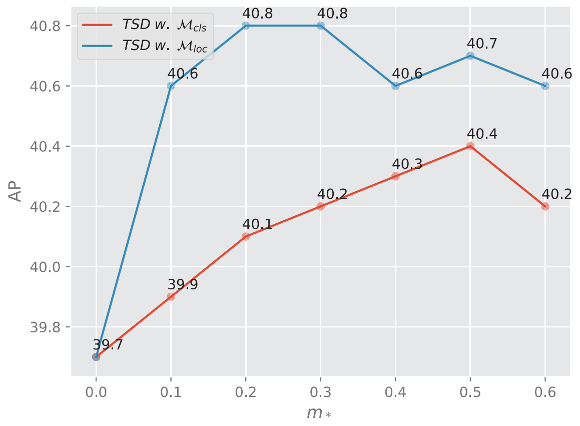


Figure 4. Results of TSD with variant m$_*$ for PC. These experiments are conducted based on ResNet-50 with FPN.

Delving to the effective PC. PC demonstrates its superiority on regressing more precise bounding boxes. The hyper-parameters m_c and m_r play important roles in the training of TSD and to better understand their effects on performance, we conduct detailed ablation studies on them. Figure.4 reports the results and note that both of the $\mathcal{M}_{los}$ and $\mathcal{M}_{cls}$ can further improve the performance.

3.3. Applicable to variant backbones

Since the TSD and PC have demonstrated their outstanding performance on ResNet-50 with FPN, we further delve

Method	Ours	AP	AP$_{.5}$	AP$_{.75}$	runtime
ResNet-50		36.1	58.0	38.8	159.4 ms
ResNet-50	✓	**41.0**	**61.7**	**44.3**	174.9 ms
ResNet-101		38.6	60.6	41.8	172.4ms
ResNet-101	✓	**42.4**	**63.1**	**46.0**	189.0ms
ResNet-101-DCN		40.8	63.2	44.6	179.3ms
ResNet-101-DCN	✓	**43.5**	**64.4**	**47.0**	200.8ms
ResNet-152		40.7	62.6	44.6	191.3ms
ResNet-152	✓	**43.9**	**64.5**	**47.7**	213.2ms
ResNeXt-101 [36]		40.5	62.6	44.2	187.5ms
ResNeXt-101 [36]	✓	**43.5**	**64.5**	**46.9**	206.6ms

Table 5. Results of TSD + PC with variant backbones. DCN means deformable convolution. The runtime includes network forward and post-processing (e.g., NMS for object detection). The runtime is the averaged value on a single Tesla V100 GPU and CPU E5-2680 v4.

into the adaptation on variant backbones. Based on Faster R-CNN, we directly conduct several experiments with different backbones and Table.5 summarizes the detailed performance on COCO *minival*. TSD can steadily improve the performance by 3%$\sim$5% with additional $\sim$10% time cost. Note that ResNet-50+TSD with 58.9M parameter can even outperform the ResNet-152 with 76.39M parameter. Based on the ResNet family, TSD is a more preferred choice than increasing backbone to improve performance. **If not specified, all subsequent TSD indicates TSD+PC**.

Method	TSD	AP$_{.5}$ (Val)	AP$_{.5}$ (LB)
ResNet-50		64.64	49.79
ResNet-50	✓	**68.18**	**52.55**
Cascade-DCN-SENet154		69.27	55.979
Cascade-DCN-SENet154	✓	**71.17**	**58.34**
DCN-ResNeXt101*		68.70	55.05
DCN-ResNeXt101*	✓	**71.71**	**58.59**
DCN-SENet154*		70	57.771
DCN-SENet154*	✓	**72.19**	**60.5**

Table 6. Results of TSD on OpenImage dataset. * indicates we expand the anchor scale to {8, 11, 14} and anchor aspect ratio to {0.1, 0.5, 1, 2, 4, 8}. Furthermore, mult-scale test is used for public leaderboard (LB) except for ResNet-50.

3.4. Applicable to Mask R-CNN

The proposed algorithms largely surpass the classical sibling head in Faster R-CNN. Its inherent properties determine its applicability to other R-CNN families such as Mask R-CNN for instance segmentation. To validate this, we conduct experiments with Mask R-CNN [13]. Performances are shown in Table.7 and the training configuration in Mask R-CNN is the same as the experiments in Faster R-CNN. It's obvious that TSD is still capable of detection branch in Mask R-CNN. The instance segmentation mask AP can also obtain promotion.

Method	Ours	AP^{bb}	AP$^{bb}_{.5}$	AP$^{bb}_{.75}$	AP^{mask}	AP$^{mask}_{.5}$	AP$^{mask}_{.75}$
ResNet-50 w. FPN		37.2	58.8	40.2	33.6	55.3	35.4
ResNet-50 w. FPN	✓	**41.5**	**62.1**	**44.8**	**35.8**	**58.3**	**37.7**
ResNet-101 w. FPN		39.5	61.2	43.0	35.7	57.9	38.0
ResNet-101 w. FPN	✓	**43.0**	**63.6**	**46.8**	**37.2**	**59.9**	**39.5**

Table 7. Results of Mask R-CNN with TSD. The proposed methods are only applied on the detection branch in Mask R-CNN. AP^{bb} means the detection performance and AP^{mask} indicates the segmentation performance.

Method	backbone	$b\&w$	AP	AP$_{.5}$	AP$_{.75}$	AP$_s$	AP$_m$	AP$_l$
RefineDet512 [41]	ResNet-101		36.4	57.5	39.5	16.6	39.9	51.4
RetinaNet800 [22]	ResNet-101		39.1	59.1	42.3	21.8	42.7	50.2
CornerNet [17]	Hourglass-104 [28]		40.5	56.5	43.1	19.4	42.7	53.9
ExtremeNet [42]	Hourglass-104 [28]		40.1	55.3	43.2	20.3	43.2	53.1
FCOS [34]	ResNet-101		41.5	60.7	45.0	24.4	44.8	51.6
RPDet [39]	ResNet-101-DCN	✓	46.5	67.4	50.9	30.3	49.7	57.1
CenterNet511 [6]	Hourglass-104	✓	47.0	64.5	50.7	28.9	49.9	58.9
TridentNet [20]	ResNet-101-DCN	✓	48.4	69.7	53.5	31.8	51.3	60.3
NAS-FPN [8]	AmoebaNet (7 @ 384)	✓	48.3	-	-	-	-	-
Faster R-CNN w FPN [21]	ResNet-101		36.2	59.1	39.0	18.2	39.0	48.2
Auto-FPN† [38]	ResNet-101		42.5	-	-	-	-	-
Regionlets [37]	ResNet-101		39.3	59.8	-	21.7	43.7	50.9
Grid R-CNN [27]	ResNet-101		41.5	60.9	44.5	23.3	44.9	54.1
Cascade R-CNN [2]	ResNet-101		42.8	62.1	46.3	23.7	45.5	55.2
DCR [4]	ResNet-101		40.7	64.4	44.6	24.3	43.7	51.9
IoU-Net† [15]	ResNet-101		40.6	59.0	-	-	-	-
Double-Head-Ext† [35]	ResNet-101		41.9	62.4	45.9	23.9	45.2	55.8
SNIPER [32]	ResNet-101-DCN	✓	46.1	67.0	51.6	29.6	48.9	58.1
DCNV2 [43]	ResNet-101	✓	46.0	67.9	50.8	27.8	49.1	59.5
PANet [24]	ResNet-101	✓	47.4	67.2	51.8	30.1	51.7	60.0
GCNet [3]	ResNet-101-DCN	✓	48.4	67.6	52.7	-	-	-
TSD†	ResNet-101		**43.1**	**63.6**	**46.7**	**24.9**	**46.8**	**57.5**
TSD	ResNet-101		**43.2**	64.0	**46.9**	**24.0**	**46.3**	**55.8**
TSD*	ResNet-101-DCN	✓	**49.4**	69.6	**54.4**	**32.7**	**52.5**	**61.0**
TSD*	SENet154-DCN [14]	✓	**51.2**	**71.9**	**56.0**	**33.8**	**54.8**	**64.2**

Table 8. Comparisons of single-model results for different algorithms evaluated on the COCO *test-dev* set. $b\&w$ indicates training with bells and whistles such as multi-scale train/test, Cascade R-CNN or DropBlock [7]. † indicates the result on COCO *minival* set.

3.5. Generalization on large-scale OpenImage

In addition to evaluate on the COCO dataset, we further corroborate the proposed method on the large-scale OpenImage dataset. As the public dataset with large-scale boxes and hierarchy property, it brings a new challenge to the generalization of detection algorithms. To fully delve the effectiveness of the proposed algorithm, we run a number of ablations to analyze TSD. Table.6 illustrates the comparison and note that, even for heavy backbone, TSD can still give satisfactory improvements. Furthermore, TSD is complementary to Cascade R-CNN [2] and embedding it into this framework can also enhance the performance by a satisfactory margin.

3.6. Comparison with state-of-the-Arts

In this section, we evaluate our proposed method on COCO *test-dev* set and compare it with other state-of-the-art methods. m_c and m_r are set to 0.5 and 0.2, respectively. For a fair comparison, we report the results of our methods under different settings in Table.8. For comparison with Grid R-CNN [27], we extend the training epochs for ResNet-101 to be consistent with it. For comparing with the best single-model TridentNet*, in **TSD***, we apply the same configuration with it including multi-scale training, soft-NMS [1], deformable convolutions and the 3× training scheme on ResNet-101. The best single-model ResNet-101-DCN gives an **AP of 49.4**, already surpassing all of the other methods with the same backbone. To our best knowledge, for a single model with ResNet-101 backbone, our result is the best entry among the state-of-the-arts. TSD demonstrates its advantage on promoting precise localization and confidential classification, especially on higher IoU thresholds (AP$_{.75}$). Furthermore, we explore the upper-bound of TSD with a heavy backbone. Surprisingly, it can

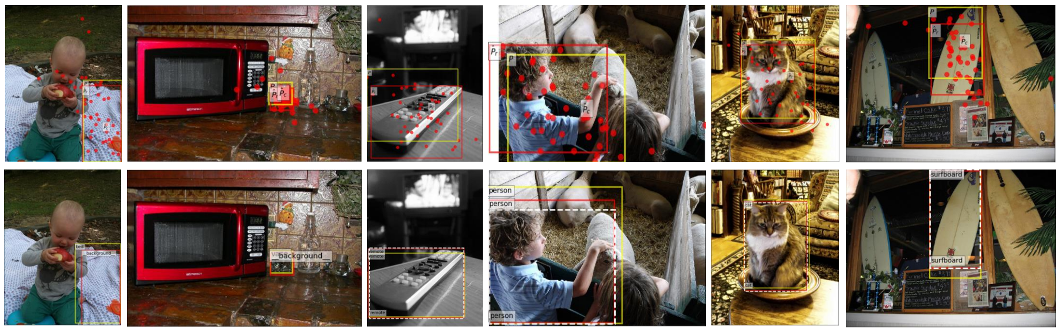


Figure 5. Visualization of the learnt $\hat{P}_r$ and $\hat{P}_c$ on examples from the COCO *minival* set. The first row indicates the proposal P (yellow box) and the derived $\hat{P}_r$ (red box) and $\hat{P}_c$ (pink point, center point in each grid). The second row is the final detected boxes where the white box is ground-truth. TSD deposes the false positives in the first two columns and in other columns, it regresses more precise boxes.

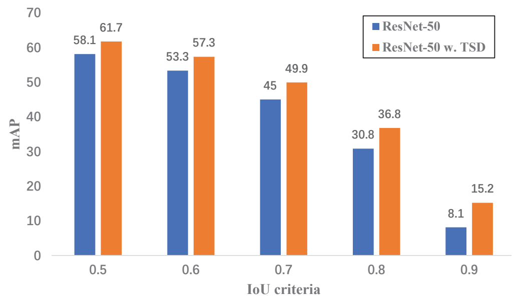


Figure 6. mAP across IoU criteria from 0.5 to 0.9 with 0.1 interval.

Criteria	TSD	$AP_{.5}$	$AP_{.6}$	$AP_{.7}$	$AP_{.8}$	$AP_{.9}$
AP_{small}		38.4	33.7	26.7	16.2	3.6
AP_{small}	✓	**40.0**	**35.6**	**28.8**	**17.7**	**5.3**
AP_{medium}		62.9	58.4	49.7	33.6	8.7
AP_{medium}	✓	**67.7**	**62.4**	**54.9**	**40.2**	**15.4**
AP_{large}		69.5	65.5	56.8	43.2	14.8
AP_{large}	✓	**74.8**	**71.6**	**65.0**	**53.2**	**27.9**

Table 9. mAP across scale criteria from 0.5 to 0.9 with 0.1 interval.

achieve the **AP of 51.2** with the single-model SENet154-DCN on COCO *test-dev* set. Soft-NMS is not used in this evaluation.

3.7. Analysis and discussion

Performance in different IoU criteria. Since TSD exhibits superior ability on regressing precise localization and predicting confidential category, we conduct several evaluations with more strict IoU criteria on COCO *minival*. Figure.6 illustrates the comparison between TSD based Faster R-CNN and baseline Faster R-CNN with the same ResNet-50 backbone across IoU thresholds from 0.5 to 0.9. Obviously, with the increasing IoU threshold, the improvement brought by TSD is also increasing.

Performance in different scale criteria. We have analyzed the effectiveness of TSD under different IoU criteria. To better explore the specific improvement, we further test the mAP under objects with different scales. Table.9 reports the performance and TSD shows successes in objects with variant scales, especially for medium and large objects.

What did TSD learn? Thanks to the *task-aware spatial disentanglement* (TSD) and the progressive constraint (PC), stable improvements can be easily achieved whether for variant backbones or variant datasets. Beyond the quantitative promotion, we wonder what TSD learned compared with the sibling head in Faster R-CNN. To better interpret this, We showcase the illustrations of our TSD compared with sibling head as shown in Figure. 5. As expected, through TSD, it can depose many false positives and regress the more precise box boundary. For $\hat{P}_r$, it tends to translate to the boundary that is not easily regressed. For $\hat{P}_c$, it tends to concentrate on the local appearance and object context information as it did in sibling head with deformable RoI pooling [5]. Note that the tangled tasks in sibling head can be effectively separated from the spatial dimension.

4. Conclusion

In this paper, we present a simple operator TSD to alleviate the inherent conflict in sibling head, which learns the task-aware spatial disentanglement to bread through the performance limitation. In particular, TSD derives two disentangled proposals from the shared proposal and learn the specific feature representation for classification and localization, respectively. Further, we propose a progressive constraint to enlarge the performance margin between the disentangled and the shared proposals, which provides additional performance gain. Without bells and whistles, this simple design can easily boost most of the backbones and models on both COCO and large scale OpenImage consistently by 3%~5%, and is the core model in our 1st solution of OpenImage Challenge 2019.

References

[1] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017. 7

[2] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 7

[3] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. *arXiv preprint arXiv:1904.11492*, 2019. 7

[4] Bowen Cheng, Yunchao Wei, Honghui Shi, Rogerio Feris, Jinjun Xiong, and Thomas Huang. Revisiting rcnn: On awakening the classification power of faster rcnn. In *The European Conference on Computer Vision (ECCV)*, September 2018. 7

[5] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 1, 3, 4, 6, 8

[6] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 7

[7] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. In *Advances in Neural Information Processing Systems*, pages 10727–10737, 2018. 7

[8] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7036–7045, 2019. 1, 7

[9] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):142–158, 2015. 1

[11] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 5

[12] Zekun Hao, Yu Liu, Hongwei Qin, Junjie Yan, Xiu Li, and Xiaolin Hu. Scale-aware face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6186–6195, 2017. 1

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 5, 6

[14] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 7

[15] Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 784–799, 2018. 1, 4, 7

[16] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv:1811.00982*, 2018. 4

[17] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018. 1, 7

[18] Buyu Li, Yu Liu, and Xiaogang Wang. Gradient harmonized single-stage detector. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8577–8584, 2019. 1

[19] Hongyang Li, Yu Liu, Wanli Ouyang, and Xiaogang Wang. Zoom out-and-in network with map attention decision for region proposal and object detection. *International Journal of Computer Vision*, 127(3):225–238, 2019. 1

[20] Yanghao Li, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Scale-aware trident networks for object detection. *arXiv preprint arXiv:1901.01892*, 2019. 7

[21] 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 conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 1, 7

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 7

[23] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 4

[24] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018. 7

[25] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 1

[26] Yu Liu, Hongyang Li, Junjie Yan, Fangyin Wei, Xiaogang Wang, and Xiaoou Tang. Recurrent scale approximation for object detection in cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 571–579, 2017. 1

[27] Xin Lu, Buyu Li, Yuxin Yue, Quanquan Li, and Junjie Yan. Grid r-cnn. In *Proceedings of the IEEE Conference on Com-*

puter Vision and Pattern Recognition, pages 7363–7372, 2019. 4, 7

[28] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016. 7

[29] Chao Peng, Tete Xiao, Zeming Li, Yuning Jiang, Xiangyu Zhang, Kai Jia, Gang Yu, and Jian Sun. Megdet: A large mini-batch object detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6181–6189, 2018. 5

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 1, 2, 3, 5

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 5

[32] Bharat Singh, Mahyar Najibi, and Larry S Davis. Sniper: Efficient multi-scale training. In *Advances in Neural Information Processing Systems*, pages 9310–9320, 2018. 7

[33] Guanglu Song, Yu Liu, Ming Jiang, Yujie Wang, Junjie Yan, and Biao Leng. Beyond trade-off: Accelerate fcn-based face detector with higher accuracy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7756–7764, 2018. 1

[34] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 7

[35] Yue Wu, Yinpeng Chen, Lu Yuan, Zicheng Liu, Lijuan Wang, Hongzhi Li, and Yun Fu. Rethinking classification and localization in r-cnn. *arXiv preprint arXiv:1904.06493*, 2019. 1, 2, 4, 7

[36] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 6

[37] Hongyu Xu, Xutao Lv, Xiaoyu Wang, Zhou Ren, Navaneeth Bodla, and Rama Chellappa. Deep regionlets for object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 798–814, 2018. 7

[38] Hang Xu, Lewei Yao, Wei Zhang, Xiaodan Liang, and Zhenguo Li. Auto-fpn: Automatic network architecture adaptation for object detection beyond classification. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1, 7

[39] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. *arXiv preprint arXiv:1904.11490*, 2019. 7

[40] Xingyu Zeng, Wanli Ouyang, Junjie Yan, Hongsheng Li, Tong Xiao, Kun Wang, Yu Liu, Yucong Zhou, Bin Yang, Zhe Wang, et al. Crafting gbd-net for object detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(9):2109–2123, 2017. 1

[41] Shifeng Zhang, Longyin Wen, Xiao Bian, Zhen Lei, and Stan Z Li. Single-shot refinement neural network for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4203–4212, 2018. 7

[42] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 850–859, 2019. 7

[43] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets v2: More deformable, better results. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9308–9316, 2019. 4, 7