The following publication S. Li, C. He, R. Li and L. Zhang, "A Dual Weighting Label Assignment Scheme for Object Detection," 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 2022, pp. 9377-9386 is available at https://doi.org/10.1109/CVPR52688.2022.00917.

A Dual Weighting Label Assignment Scheme for Object Detection

Shuai Li, Chenhang He, Ruihuang Li, Lei Zhang
The Hong Kong Polytechnic University
{csshuaili, csche, csrhli, cslzhang}@comp.polyu.edu.hk

Abstract

Label assignment (LA), which aims to assign each training sample a positive (pos) and a negative (neg) loss weight, plays an important role in object detection. Existing LA methods mostly focus on the design of pos weighting function, while the neg weight is directly derived from the pos weight. Such a mechanism limits the learning capacity of detectors. In this paper, we explore a new weighting paradigm, termed dual weighting (DW), to specify pos and neg weights separately. We first identify the key influential factors of pos/neg weights by analyzing the evaluation metrics in object detection, and then design the pos and neg weighting functions based on them. Specifically, the pos weight of a sample is determined by the consistency degree between its classification and localization scores, while the neg weight is decomposed into two terms: the probability that it is a neg sample and its importance conditioned on being a neg sample. Such a weighting strategy offers greater flexibility to distinguish between important and less important samples, resulting in a more effective object detector. Equipped with the proposed DW method, a single FCOS-ResNet-50 detector can reach 41.5% mAP on COCO under 1× schedule, outperforming other existing LA methods. It consistently improves the baselines on COCO by a large margin under various backbones without bells and whistles. Code is available at https://github.com/strongwolf/DW.

1. Introduction

As a fundamental vision task, object detection has been drawing significant attention from researchers for decades. The community has recently witnessed a fast evolution of detectors with the development of convolutional neural networks (CNNs) [13–15, 34–37] and visual transformers (ViTs) [4, 6, 8, 10, 27, 39, 40, 42, 50]. Current state-of-the-art detectors [1, 22, 24, 29–31, 38, 46, 48, 49] mostly perform dense detection by predicting class labels and regression offsets with a set of pre-defined anchors. As the basic unit in detector training, anchors need to be assigned with proper

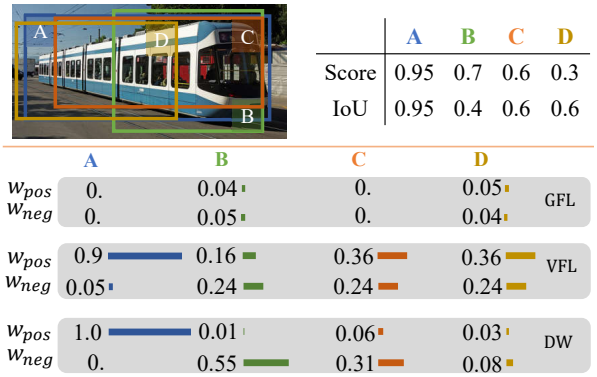

Figure 1. An illustration of the difference between the proposed DW method and existing label assignment methods, *e.g.*, GFL [21] and VFL [43]. For ambiguous anchors B, C and D, GFL and VFL will assign nearly the same *pos* and *neg* weights to {B, D} and {C, D}, respectively. In contrast, our DW assigns a distinct (*pos*, *neg*) pair for each anchor.

classification (*cls*) and regression (*reg*) labels to supervise the training process. Such a label assignment (LA) process can be regarded as a task of assigning loss weight to each anchor. The *cls* loss (*reg* loss can be similarly defined) for an anchor can be generally expressed as:

$$\mathcal{L}_{cls} = -w_{pos} \times \ln(s) - w_{neg} \times \ln(1-s), \quad (1)$$

where w_{pos} and w_{neg} are the positive (*pos*) and negative (*neg*) weights, respectively, and s is the predicted classification score. Depending on the design of w_{pos} and w_{neg}, the LA methods can be roughly divided into two categories: hard LA and soft LA.

Hard LA assumes that each anchor is either *pos* or *neg*, which means that $w_{pos}, w_{neg} \in \{0, 1\}$ and $w_{neg} + w_{pos} = 1$. The core idea of this strategy is to find a proper division boundary to split the anchors into a positive set and a negative set. The division rule along this line of research can be further categorized into static and dynamic ones. Static rules [18, 24, 32, 38] adopt pre-defined metrics such as the IoU or the distance from the anchor center to the ground truth (GT) center to match objects or background

to each anchor. Such static assignment rules ignore the fact that the division boundaries of objects with different sizes and shapes may vary. Recently, many dynamic assignment rules [12, 26] have been proposed. For instance, ATSS [44] splits the training anchors of an object based on their IoU distributions. Prediction-aware assignment strategies [4, 17, 19] regard the predicted confidence score as a reliable indicator for estimating an anchor's quality. Both static and dynamic assignment methods ignore the fact that samples are not equally important. The evaluation metric in object detection suggests that an optimal prediction should have not only a high classification score but also an accurate localization, which implies that anchors with higher consistencies between the *cls* head and *reg* head should have greater importance during training.

With the above motivation, researchers have opted to assign soft weights to anchors. GFL [21] and VFL [43] are two typical methods which define soft label targets based on IoUs and then translate them into loss weights by multiplying a modulation factor. Some other works [9, 11] compute sample weights by jointly considering the *reg* score and *cls* score. Existing methods mainly focus on the design of *pos* weighting function while the *neg* weight is simply derived from the *pos* weight, which may limit the learning capacity of detectors due to little new supervision information provided by *neg* weights. We argue that such coupled weighting mechanism cannot distinguish each training sample at a finer level. Fig. 1 shows an example. Four anchors have different prediction results. However, GFL and VFL assign nearly the same (*pos*, *neg*) weight pair to (B, D) and (C, D), respectively. GFL also assigns both zero *pos* and *neg* weight to anchor A and C since each one has the same *cls* score and IoU. As the *neg* weighting function is highly correlated with the *pos* one in existing soft LA methods, anchors with different attributes can sometimes be assigned nearly the same (*pos*, *neg*) weights, which may impair the effectiveness of the trained detector.

To provide more discriminative supervision signals to the detector, we propose a new LA scheme, termed dual weighting (DW), to specify *pos* and *neg* weights from different perspectives and make them complementary to each other. Specifically, the *pos* weights are dynamically determined by the combination of confidence scores (obtained from the *cls* head) and the *reg* scores (obtained from the *reg* head). The *neg* weight for each anchor is decomposed into two terms: the probability that it is a *neg* sample and its importance conditioned on being a *neg* sample. The *pos* weight reflects the consistency degree between the *cls* head and *reg* head, and it will push anchors with higher consistencies to move forward in the anchor list, while the *neg* weight reflects the inconsistency degree and pushes the inconsistent anchors to the rear of the list. By this means, at inference the bounding boxes with higher *cls* scores and more precise locations will have better chances to survive after NMS, and those bounding boxes with imprecise locations will fall behind and be filtered out. Referring to Fig. 1, DW distinguishes four different anchors by assigning them distinct (*pos*, *neg*) weight pairs, which can provide the detector with more fine-grained supervision training signals.

In order to provide our weighing functions with more accurate *reg* scores, we further propose a box refinement operation. Specifically, we devise a learned prediction module to generate four boundary locations based on the coarse regression map, and then aggregate the prediction results of them to get the updated bounding box for the current anchor. This light-weight module enables us to provide more accurate *reg* scores to DW by only introducing moderate computational overhead.

The advantage of our proposed DW method is demonstrated by comprehensive experiments on MS COCO [23]. In particular, it boosts the FCOS [38] detector with ResNet-50 [13] backbone to a 41.5/42.2 AP w/wo box refinement on the COCO validation set under the common $1\times$ training scheme, surpassing other LA methods.

2. Related Work

Hard Label Assignment. Labeling each anchor to be a *pos* or *neg* sample is a key procedure to train a detector. Classical anchor-based object detectors [24, 32] set an anchor's label by measuring its IoU with the GT objects. Recently, anchor-free detectors have attracted much attention due to their concise design and comparable performance. FCOS [38] and Foveabox [18] both select *pos* samples by a center sampling strategy: anchors that are close to the GT centers are sampled as positives and others are negatives or ignored during training. The above mentioned LA methods adopt a fixed rule for GT boxes with diverse shapes and sizes, which is sub-optimal.

Some advanced LA strategies [12, 16, 17, 25, 28, 44] have been proposed to dynamically choose *pos* samples for each GT. ATSS [44] selects top-k anchors from each level of the feature pyramid and adopts the `mean+std` IoU of these top anchors as the *pos*/*neg* division threshold. PAA [17] adaptively separates anchors into *pos*/*neg* ones based on the joint status of *cls* and *reg* losses in a probabilistic manner. OTA [12] handles the LA problem from a global perspective by formulating the assignment process as an optimal transportation problem. Transformer-based detectors [4, 6, 27, 50] adopt a one-to-one assignment scheme by finding the best *pos* sample for each GT. Hard LA treats all samples equally, which however is less compatible with the evaluation metric in object detection.

Soft Label Assignment. Since the predicted boxes have different qualities in evaluation, the samples should be treated differently during training. Many works [3, 20–22, 43] have been proposed to address the inequality issue

of training samples. Focal Loss [22] adds a modulated factor on the cross entropy loss to down-weight the loss assigned to well-classified samples, which pushes the detector to focus on hard samples. Generalized focal loss [21] assigns each anchor a soft weight by jointly considering the *cls* score and localization quality. Varifocal loss [43] utilizes an IoU-aware *cls* label to train the *cls* head. Most methods mentioned above focus on computing the *pos* weight and simply define the *neg* weight as a function of 1 - w_{pos}. In this paper, we decouple this procedure and separately assign *pos* and *neg* loss weights for each anchor. Most soft LA methods assign weights to loss. There is a special case that weights are assigned to score, which can be formulated as $\mathcal{L}_{cls} = -\ln(w_{pos} \times s) - \ln(1 - w_{neg} \times s)$. Typical methods include FreeAnchor [45] and Autoassign [47]. It should be noted that our method is distinct from them. To match anchors in a fully differential manner, w_{pos} and w_{neg} in Autoassign still receive gradients. In our method, however, the loss weights are carefully designed and entirely detached from the network, which is a common practice for weighting loss.

3. Proposed Method

3.1. Motivation and Framework

To be compatible with NMS, a good dense detector should be able to predict consistent bounding boxes that have both high classification scores and precise locations. However, if all the training samples are equally treated, there will be a misalignment between the two heads: the location with the highest category score is usually not the best position for regressing the object boundary. This misalignment can degrade the performance of detectors, especially under high IoU metrics. Soft LA, which treats the training samples in a soft manner by weighting loss, is an attempt to enhance the consistency between the *cls* and *reg* heads. With soft LA, the loss of an anchor can be expressed as:

$$\begin{aligned} \mathcal{L}_{cls} &= -w_{pos} \times \ln(s) - w_{neg} \times \ln(1-s), \\ \mathcal{L}_{reg} &= w_{reg} \times \ell_{reg}(b, b'), \end{aligned} \tag{2}$$

where s is the predicted *cls* score, b and b' are the locations of the predicted bounding box and the GT object, respectively, and ℓ_{reg} is the regression loss such as Smooth L_1 Loss [32], IoU Loss [41] and GIoU Loss [33]. The inconsistency problem between *cls* and *reg* heads can be mitigated by assigning larger w_{pos} and w_{reg} to anchors with higher consistencies. These well-trained anchors are thus able to predict high *cls* scores and precise locations simultaneously at inference.

Existing works commonly set w_{reg} equal to w_{pos} and mainly focus on how to define the consistency and integrate it into loss weights. Table 1 summarizes the formulations of w_{pos} and w_{neg} for a *pos* anchor in recent representative methods. One can see that current methods commonly define a metric t to indicate the consistency degree between the two heads at the anchor level, and then design the inconsistency metric as a function of $1 - t$. The consistent and inconsistent metrics are finally integrated into the *pos* and *neg* loss weights by adding a scaling factor ($(s-t)^2$, s^2 or t), respectively.

Table 1. Comparison of different weighting functions.

Method	w_{pos}	w_{neg}	t
GFL [21]	$(s-t)^2 \times t$	$(s-t)^2 \times (1-t)$	IoU
VFL [43]	$t \times t$	$t \times (1-t)$	IoU
TOOD [9]	$(s-t)^2 \times t$	$(s-t)^2 \times (1-t)$	$f(IoU, s)$
MuSu [11]	$(s-t)^2 \times t$	$s^2 \times (1-t)^4$	$f(IoU, s)$
Ours (DW)	$f_{pos}(IoU, s)$	$P_{neg} \times I_{neg}$	-

Different from above methods where w_{pos} and w_{neg} are highly correlated, we propose to set *pos* and *neg* weights separately in a prediction-aware manner. Specifically, the *pos* weighting function takes the predicted *cls* score s and the IoU between the predicted box and the GT object as inputs, and set the *pos* weight by estimating the consistency degree between the *cls* and *reg* heads. The *neg* weighting function takes the same inputs as the *pos* weighting function but formulates the *neg* weight as the multiplication of two terms: the probability that the anchor is a *neg* one, and its importance conditioned on that it is *neg*. By this way, the ambiguous anchors that have similar *pos* weights can receive more fine-grained supervision signals with distinct *neg* weights, which is not available in existing methods.

The pipeline of our DW framework is shown in Fig. 2. As a common practice [9, 11, 12, 38], we first construct a bag of candidate positives for each GT object by selecting anchors near the GT center (center prior). Anchors outside the candidate bag are considered as *neg* samples which will not be involved in the design process of weighting functions since their statistics (*e.g.*, IoU, *cls* score) are very noisy at the early training stage. Anchors inside the candidate bag will be assigned to three weights including w_{pos}, w_{neg} and w_{reg}, to supervise the training process more effectively.

3.2. Positive Weighting Function

The *pos* weight of a sample should reflect its importance for accurately detecting an object in both classification and localization. We try to find out the factors affecting this importance by analyzing the evaluation metric of object detection. During testing on COCO, all the predictions for one category should be properly ranked by a ranking metric. Existing methods commonly use the *cls* score [32] or the combination of *cls* score and predicted IoU [44] as the ranking metric. The correctness of each bounding box will be checked from the begin of the ranking list. A prediction will be defined as a correct one if and only if:

a. The IoU between the predicted bounding box and its nearest GT object is larger than a threshold θ;

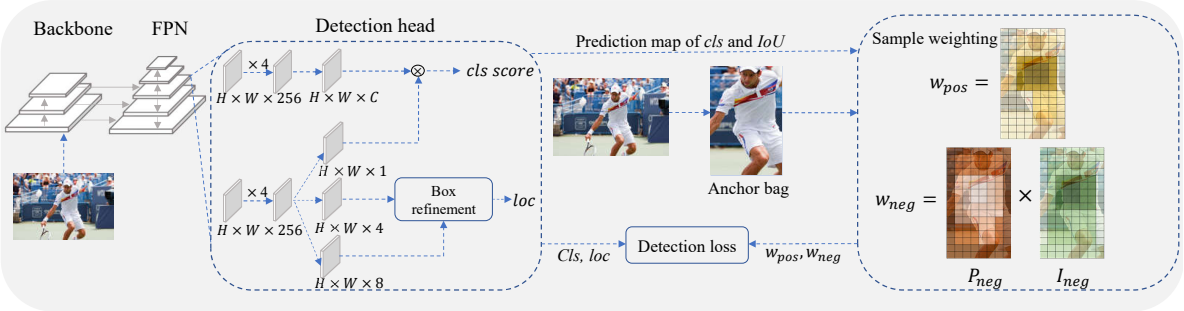

Figure 2. Pipeline of DW. The left part shows the overall detection model which consists of backbone, FPN and detection head. The outputs from the classification branch ($H \times W \times C$) and the centerness branch ($H \times W \times 1$) are multiplied as the final *cls* score. The box refinement module utilizes the four predicted boundary points ($H \times W \times 8$) to adjust the coarse prediction ($H \times W \times 4$) to finer locations.The right part shows the weighting process. Given an object, a candidate anchor bag is first constructed by selecting the anchor points near the object center. Each anchor will then be assigned a *pos* weight and *neg* weight from different aspects.

b. No box that satisfies the above condition ranks in front of the current box.

In a word, only the first bounding box that has a larger IoU than θ in the prediction list will be defined as a *pos* detection, while all the other bounding boxes should be considered as false positives of the same GT. It can be seen that high ranking score and high IoU are both *sufficient* and *necessary* conditions for a *pos* prediction. This implies that anchors that simultaneously satisfy the two conditions are more likely to be defined as *pos* predictions during testing, and thus they should have higher importance during training. From this perspective, the *pos* weight w_{pos} should be positively correlated with the IoU and ranking score, *i.e.*, $w_{pos} \propto IoU$ and $w_{pos} \propto s$. To specify the *pos* function, we first define a consistency metric, denoted as t, to measure the alignment degree between the two conditions:

$$t = s \times IoU^{\beta}, \tag{3}$$

where β is used to balance the two conditions. To encourage a large variance of *pos* weights among different anchors, we add an exponential modulation factor:

$$w_{pos} = e^{\mu t} \times t, \tag{4}$$

where μ is a hyper-parameter to control the relative gaps of different *pos* weights. Finally, the *pos* weight of each anchor for each instance is normalized by the sum of all *pos* weights within the candidate bag.

3.3. Negative Weighting Function

Though *pos* weights can enforce consistent anchors to have both high *cls* scores and large IoUs, the importance of less consistent anchors cannot be distinguished by *pos* weights. Referring to Fig. 1, anchor D has a finer location (a larger IoU than θ) but a lower *cls* score, while anchor B has a coarser location (a smaller IoU than θ) but a higher *cls* score. They may have the same consistency degree t and thus will be pushed forward with the same *pos* strength which can not reflect their differences. To provide more discriminative supervision information for the detector, we propose to indicate their importance faithfully by assigning more distinct *neg* weights to them, which are defined as the multiplication of the following two terms.

Probability of being a Negative Sample. According to the evaluation metric of COCO, an IoU smaller than θ is a sufficient condition for a false prediction. This means that a predicted bounding box that does not satisfy the IoU metric will be regarded as a *neg* detection, even if it has a high *cls* score. That is, IoU is the only factor to determine the probability of being a *neg* sample, denoted by P_{neg}. Since COCO adopts an IoU interval ranging from 0.5 to 0.95 to estimate AP, the probability P_{neg} for a bounding box should satisfy the following rules:

$$P_{neg} = \begin{cases} 1, & \textit{if IoU} < 0.5, \\ [0,1], & \textit{if IoU} \in [0.5,0.95], \\ 0, & \textit{if IoU} > 0.95, \end{cases} \tag{5}$$

Any monotonically decreasing function defined within the interval [0.5,0.95] is qualified for P_{neg}. For simplicity, we instantiate P_{neg} as the following function:

$$P_{neg} = -k \times IoU^{\gamma_1} + b, \quad \textit{if IoU} \in [0.5,0.95], \tag{6}$$

which passes through the points (0.5, 1) and (0.95,0). Once γ_1 is determined, the parameters k and b can be obtained by the method of undetermined coefficients. Fig. 3 plots the curves of P_{neg} vs. IoU with different values of γ_1.

Importance Conditioned on being a Negative Sample. At inference, a *neg* prediction in the ranking list will not affect the recall but decrease the precision. To delay this process, the *neg* bounding boxes should rank as behind as

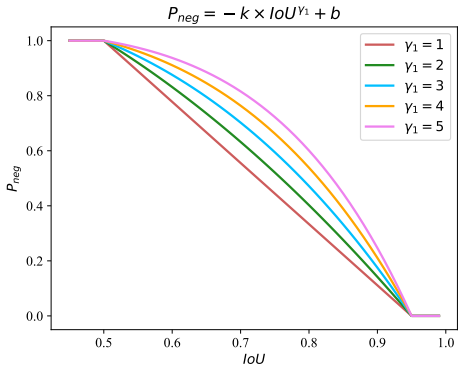

Figure 3. Curves of P_{neg} in [0.5,0.95] vs. IoU with different γ_1.

possible, *i.e.*, their ranking scores should be as small as possible. Based on this point, the *neg* predictions with larger ranking scores are more important than those with smaller ranking scores as they are harder examples for network optimization. Thus, the importance of *neg* samples, denoted by I_{neg}, should be a function of the ranking score. For simplicity, we set it as

$$I_{neg} = s^{\gamma_2}, \tag{7}$$

where γ_2 is a factor to indicate how much preference should be given to important *neg* samples.

Finally, the *neg* weight $w_{neg} = P_{neg} \times I_{neg}$ becomes

$$w_{neg} = \begin{cases} s^{\gamma_2}, & \text{if IoU} < 0.5, \\ (-k \times IoU^{\gamma_1} + b) \times s^{\gamma_2}, & \text{if IoU} \in [0.5,0.95], \\ 0, & \text{if IoU} > 0.95, \end{cases} \tag{8}$$

which is negatively correlated with IoU but positively correlated with s. It can be seen that for two anchors with the same *pos* weight, the anchor with a smaller IoU will have a larger *neg* weight. The definition of w_{neg} is well compatible with the inference process and it can further distinguish ambiguous anchors having almost the same *pos* weights. Please refer to Fig. 1 for examples.

3.4. Box Refinement

Since both the *pos* and *neg* weighting functions take IoU as an input, more accurate IoUs can induce higher quality samples, benefiting the learning of stronger features. We propose a box refinement operation to refine the bounding boxes based on the predicted offset map $O \in R^{H \times W \times 4}$, where $O(j,i) = \{\Delta l, \Delta t, \Delta r, \Delta b\}$ represents the predicted distances from the center of current anchor to the leftmost *l*, topmost *t*, rightmost *r* and bottommost *b* sides of the GT object, respectively, as shown in Fig. 4. Motivated by the fact that points near the object boundary are more likely to predict accurate locations, we devise a learnable prediction module to generate a boundary point for each side based on

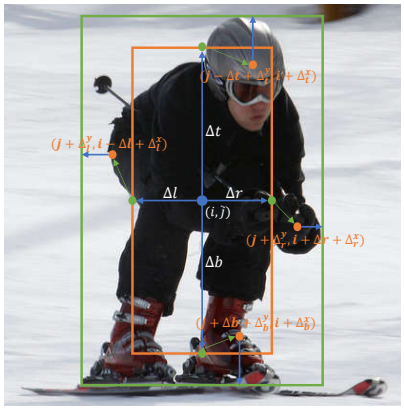

Figure 4. Illustration of the box refinement operation. A coarse bounding box (orange box) of an anchor at location (*j*,*i*) is first generated by predicting four distances = $\{\Delta l, \Delta t, \Delta r, \Delta b\}$. Four boundary points (orange points) are then predicted with respect to the four side points (green points). Finally, a finer bounding box (green box) is generated by aggregating the prediction results of the four boundary points.

the coarse bounding box. Referring to Fig. 4, the coordinates of the four boundary points are defined as:

$$\begin{aligned} B_l &= \left(j + \Delta_l^y, i - \Delta l + \Delta_l^x\right), B_t = \left(j - \Delta t + \Delta_t^y, i + \Delta_t^x\right), \\ B_r &= \left(j + \Delta_r^y, i + \Delta r + \Delta_r^x\right), B_b = \left(j + \Delta b + \Delta_b^y, i + \Delta_b^x\right), \end{aligned} \tag{9}$$

where $\{\Delta_l^x, \Delta_l^y, \Delta_t^x, \Delta_t^y, \Delta_r^x, \Delta_r^y, \Delta_b^x, \Delta_b^y\}$ are the outputs of the refinement module.

The refined offset map O' is updated as:

$$O'(j,i) = \left\{ \begin{array}{l} \Delta l + \Delta_l^x + O(B_l, 0),\ \Delta t + \Delta_t^y + O(B_t, 1) \\ \Delta r + \Delta_r^x + O(B_r, 2),\ \Delta b + \Delta_b^y + O(B_b, 3) \end{array} \right\} \tag{10}$$

3.5. Loss Function

The proposed DW scheme can be applied to most existing dense detectors. Here we adopt the representative dense detector FCOS [38] to implement DW. As shown in Fig. 2, the whole network structure comprises a backbone, FPN and detection head. Following the conventions [11, 38, 47], we multiply the outputs of centerness branch and classification branch as the final *cls* score. The final loss of our network is

$$\mathcal{L}_{det} = \mathcal{L}_{cls} + \beta \mathcal{L}_{reg}, \tag{11}$$

where β is a balancing factor which is the same as the one in Eq. 3, and

$$\begin{aligned} \mathcal{L}_{cls} &= \sum_{n=1}^{N} -w_{pos}^n \times \ln\left(s^n\right) - w_{neg}^n \times \ln\left(1 - s^n\right) \\ &\quad + \sum_{m=1}^{M} FL\left(s^m, 0\right), \\ \mathcal{L}_{reg} &= \sum_{n=1}^{N} w_{pos}^n \times GIoU\left(b, b'\right), \end{aligned} \tag{12}$$

where N and M are the total number of anchors inside and outside the candidate bag, respectively, FL is the Focal Loss [22], GIoU is the regression loss [33], s is the predicted *cls* score, b and b' are the locations of the predicted box and the GT object, respectively.

Table 2. Detection performances by setting different hyper-parameters in w_{pos}.

β	5						3	4	6	7
μ	3	4	5	6	7	8	5			
AP	40.8	41.2	**41.5**	**41.5**	41.4	41.2	40.8	41.3	41.4	41
AP50	59.1	59.7	59.8	60.1	59.8	59.6	59.9	59.9	59.6	59
AP75	43.9	44.2	45	44.6	45.1	44.5	43.6	44.5	44.9	44.4

Table 3. Detection performances by setting different values of γ_1 and γ_2 in w_{neg}.

γ_1	γ_2	AP	AP50	AP75
1	1	41.	59.2	44.1
1	2	41.3	59.7	44.6
2	2	**41.5**	59.8	45
3	2	41.3	59.7	44.4
4	2	41.2	59.4	44.4
5	2	41.1	59.5	44.5
2	3	41.3	59.6	44.5

4. Experiments

Dataset and Evaluation Metric. Extensive experiments are conducted on the large-scale detection benchmark MS-COCO [23] which contains 115K, 5K and 20K images for `train`, `val` and `test-dev` sets, respectively. We report the analysis and ablation studies on `val` set and compare with other state-of-the-arts on the `test-dev` set. The performance is measured by COCO Average Precision (AP).

Implementation Details. We use ResNet-50 pretrained on ImageNet [7] with FPN [32] as our backbone for all experiments unless otherwise specified. Following the common practice, most models are trained with 12 epochs which is denoted as $1\times$ in [5]. The initial learning rate is 0.01 and is decayed by a factor of 10 after the 8^{th} and 11^{th} epoch. For all ablations, we use an image scale of 800 pixels for training and testing unless otherwise specified. All experiments are trained with SGDM [2] on 8 GPUs with a total batch size 16 (2 images per GPU). At inference, we filter out background boxes with a threshold 0.05 and remove redundant boxes by NMS with a threshold 0.6 to get the final predicted results. The hyper-parameters γ_1, γ_2, β and μ are set to 2, 2, 5 and 5, respectively.

4.1. Ablation Studies

Hyper-parameters of Positive Weighting. There are two hyper-parameters for *pos* weights: β and μ. β balances the contributions between the *cls* score and the IoU in the consistency metric t. As β increases, the contribution degree of IoU also increases. μ controls the relative scales of *pos* weights. A larger μ enables the most consistent samples to have relatively larger *pos* weights compared with less consistent samples. We show the performance of DW by varying β from 3 to 7 and μ from 3 to 8 in Table 2. One can see that the best result is achieved when β is 5 and μ is 5. Other combinations of β and μ will degrade the AP performance from 0.1 to 0.7. Therefore, we set β and μ to 5 in all the rest experiments.

Table 4. Comparisons of different ways to select candidate bag.

Center prior		AP	AP50	AP75
Threshold	1	41.2	59.7	44.7
	1.3	41.3	59.6	44.4
	1.7	41.4	59.5	44.6
	2.0	41.3	59.6	44.4
	2.5	41.1	59.1	44.3
Top-k	9	41.2	59.4	44.3
	12	41.2	59.4	44.6
	15	41.2	59.6	44.4
Soft center prior	e^{-r^2}	**41.5**	59.8	45.0

Table 5. Comparisons of different ways to formulate w_{neg}.

w_{pos}	w_{neg}		AP	AP50	AP75
	P_{neg}	I_{neg}			
√	×	×	39.5	58.6	42.9
√	√	×	40.5	58.7	43.9
√	×	√	40.0	58.5	42.9
√	$1-w_{pos}$		40.7	59.5	44.1
√	√	√	**41.5**	59.8	45.0

Hyper-parameters of Negative Weighting. We also conduct several experiments to investigate the robustness of DW to hyper-parameters γ_1 and γ_2, which are used to modulate the relative scales of *neg* weights. The AP results by using different combinations of γ_1 and γ_2 range from 41 to 41.5, as shown in Table 3. This implies that the performance of DW is not sensitive to the two hyper-parameters. We adopt $\gamma_1 = 2$, $\gamma_2 = 2$ in all our experiments.

Construction of Candidate Bag. As a common practice in object detection, soft LA is only applied on anchors inside the candidate bag. We test three ways of candidate bag construction which are all based on the distance r (normalized by the feature stride) from the anchor point to the corresponding GT center. The first way is to select anchors with a distance smaller than a threshold. The second is to select the top-k nearest anchors from each level of FPN. The third is to give each anchor a soft center weight e^{-r^2} and multiply it with w_{pos}. The results are shown in Table 4. It can be seen that the AP performance fluctuates slightly between 41.1 and 41.5, which demonstrates that our DW is robust to the separation methods of candidate bag.

Design of Negative Weighting Function. We investigate the influence of *neg* weighting functions by replacing it with other alternatives, as shown in Table 5. We can see that only using the *pos* weights degrades the performance to 39.5, which indicates that for some low-quality anchors, only assigning them small w_{pos} is not enough to decrease their ranking scores. They can be enforced to rank behind with a larger w_{neg}, leading to a higher AP during testing.

Table 6. Comparison among different weighting strategies of LA.

Method	AP	AP50	AP75	Reference
FoveaBox [18]	36.4	55.8	38.8	-
FCOS [38]	38.6	57.4	41.4	ICCV19
ATSS [44]	39.2	57.4	42.2	CVPR20
PAA [17]	40.4	58.4	43.9	ECCV20
OTA [12]	40.7	58.4	44.3	CVPR21
Autoassign [47]	40.4	59.6	43.7	-
Autoassign(detach)	39.8	59.6	42.8	-
Autoassign(weight loss)	36.6	56.2	39.1	-
NoisyAnchor [19]	38.0	56.9	40.6	CVPR2020
MAL [16]	39.2	58.0	42.3	CVPR2020
GFL [21]	39.9	58.5	43.0	NeurIPS20
VFL [43]	40.2	58.2	44.0	CVPR21
FCOS+GFLv2 [20]	40.6	58.2	43.9	CVPR21
ATSS+GFLv2 [20]	41.1	58.8	44.9	CVPR21
MuSu [11]	40.6	58.9	44.3	ICCV21
TOOD [9]	40.3	58.5	43.8	ICCV21
DW	41.5	59.8	45.0	
DW+box refine	**42.2**	60.4	45.3	

Without using I_{neg} or P_{neg}, we obtain 40.5 AP and 40.0 AP, respectively, which verifies that both the two terms are necessary. As done in existing methods, we attempt to replace w_{neg} with $1-w_{pos}$, but achieve a performance of 40.7 AP, 0.8 points lower than our standard DW.

Box Refinement. Without box refinement, our DW method reaches 41.5 AP, which to our best knowledge is the first method to achieve a performance of more than 41 AP on COCO without increasing any parameters and training cost over FCOS-ResNet-50. With box refinement, DW can reach 42.2 AP, as shown in Table 6. Table 7 also shows that box refinement can consistently boost the performance of DW with different backbones.

Weighting Strategies. To demonstrate the effectiveness of our DW strategy, we compare it with other LA methods using different weighting strategies. The results are shown in Table 6. The first five rows are hard LA methods while the others are soft LA.

The best performance for hard LA is achieved by OTA, 40.7 AP. Since OTA formulates LA as an Optimal Transport problem, it will increase the training time by more than 20%. GFLv2 utilizes an extra sophisticated branch to estimate the localization quality and achieves the second best performance of 41.1 AP among soft LA methods.

Unlike the mainstream methods where weights are assigned to loss, Autoassign assigns weights to *cls* score and updates them by their gradients during training. We tried to detach the weights in Autoassign and assigned them to loss, but only obtained 39.8 and 36.6 AP, respectively, 0.6 and 3.8 points lower than the original performance. This implies that the weighting scheme in Autoassign cannot work well when adapting it to the mainstream practice.

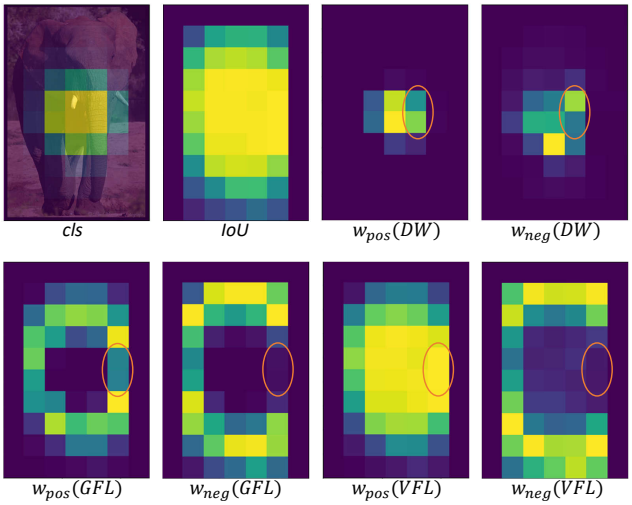

Figure 5. Visualization of *cls* score, IoU, *pos* and *neg* weights.

4.2. Comparison with State-of-the-Arts

We compare our DW with other one-stage detectors on `test-dev` 2017 in Table 7. Following previous works [9, 21, 43], the multi-scale training strategy and 2× learning schedule (24 epochs) are adopted during training. We report the results of single-model single-scale testing for all methods. Other settings are consistent with [9, 21, 43].

Apart from the LA strategy, some works [9, 20, 43] also utilize additional feature learning modules to boost their detectors. For fair comparisons, in Table 7 we compare with them by reporting the performance w/wo this auxiliary module. It can be seen that our DW method with ResNet-101 achieves 46.2 AP, outperforming all other competitive methods with the same backbone, including VFL (44.9 AP), GFL (45.0 AP) and OTA (45.3 AP). When using more powerful backbones like ResNet-101-DCN and ResNeXt-101-64x4d, DW reaches 49.3 and 48.2 AP, surpassing GFL by 2 and 2.2 points, respectively. We can also see that the operation of box refinement consistently improves DW with different backbones. It is worth mentioning that when we replace the detection head in FCOS by the one proposed in TOOD [9], DW achieves 49.8 AP, 1.5 points better than TOOD. This demonstrates the good generalization capacity of our DW strategy to other detection heads.

4.3. Discussions

Visualization of DW. To further understand the difference between DW and existing methods, we show the visualization maps of *cls* score, IoU, *pos* and *neg* weights of DW and two representative methods, GFL [21] and VFL [43], in Fig 5. It can be seen that *pos* and *neg* weights in DW are mainly centralized on the central region of the GT, while

Table 7. Performance comparison with state-of-the-art dense detectors on COCO 2017 `test-dev` set. All models listed below adopt multi-scale training. '*' indicates we use the same detection head proposed in TOOD [9]. 'Aux.' means the auxiliary learning module.

Method	Backbone	Aux.	AP	AP50	AP75	AP_S	AP_M	AP_L
FCOS [38]	ResNet-101	×	41.5	60.7	45.0	24.4	44.8	51.6
ATSS [44]	ResNet-101	×	43.6	62.1	47.4	26.1	47.0	53.6
PAA [17]	ResNet-101	×	44.8	63.3	48.7	26.5	48.8	56.3
GFL [21]	ResNet-101	×	45.0	63.7	48.9	27.2	48.8	54.5
OTA [12]	ResNet-101	×	45.3	63.5	49.3	26.9	48.8	56.1
IQDet [26]	ResNet-101	×	45.1	63.4	49.3	26.7	48.5	56.6
MuSu [11]	ResNet-101	×	44.8	63.2	49.1	26.2	47.9	56.4
Autoassign [47]	ResNet-101	×	44.5	64.3	48.4	25.9	47.4	55.0
VFL [43]	ResNet-101	×	44.9	64.1	48.9	**27.1**	48.7	55.1
DW (ours)	ResNet-101	×	**46.2**	**64.8**	**50.0**	**27.1**	**49.4**	**58.5**
GFLv2 [20]	ResNet-101	√	46.2	64.3	50.5	27.8	49.9	57.0
DW+box refine (ours)	ResNet-101	√	46.8	65.1	50.5	27.7	49.9	59.1
ATSS [44]	ResNet-101-DCN	×	46.3	64.7	50.4	27.7	49.8	58.4
PAA [44]	ResNet-101-DCN	×	47.4	65.7	51.6	27.9	51.3	60.6
GFL [21]	ResNet-101-DCN	×	47.3	66.3	51.4	28.0	51.1	59.2
MuSu [11]	ResNet-101-DCN	×	47.4	65.0	51.8	27.8	50.5	60.0
VFL [43]	ResNet-101-DCN	×	48.5	67.4	52.9	29.1	**52.2**	61.9
DW (ours)	ResNet-101-DCN	×	**49.3**	**67.6**	**53.3**	**29.2**	**52.2**	**63.5**
GFLv2 [20]	ResNet-101-DCN	√	48.3	66.5	52.8	28.8	51.9	60.7
DW+box refine (ours)	ResNet-101-DCN	√	49.5	67.7	53.4	28.9	52.2	63.7
ATSS [44]	ResNeXt-101-64x4d	×	45.6	64.6	49.7	28.5	48.9	55.6
PAA [17]	ResNeXt-101-64x4d	×	46.6	65.6	50.8	28.8	50.4	57.9
GFL [21]	ResNeXt-101-64x4d	×	46.0	65.1	50.1	28.2	49.6	56.0
OTA [12]	ResNeXt-101-64x4d	×	47.0	65.8	51.1	29.2	50.4	57.9
DW(ours)	ResNeXt-101-64x4d	×	**48.2**	**67.1**	**52.2**	**29.6**	**51.2**	**60.8**
VFL [43]	ResNeXt-101-64x4d	√	48.5	67.0	52.6	30.1	51.7	59.7
TOOD [9]	ResNeXt-101-64x4d	√	48.3	66.5	52.4	**30.7**	51.3	58.6
DW+box refine (ours)	ResNeXt-101-64x4d	√	48.7	67.1	52.7	29.7	51.6	61.1
DW* (ours)	ResNeXt-101-64x4d	√	49.8	67.7	53.8	30.4	52.3	63.0

GFL and VFL assign weights on a much wider region. This difference implies that DW can focus more on important samples and reduce the contribution of easy samples, such as those near the boundary of the object. This is why DW is more robust to the selection of candidate bag.

We can also see that anchors in the central region have distinct (*pos*, *neg*) weight pairs in DW. In contrast, the *neg* weights are highly correlated with *pos* weights in GFL and VFL. Anchors highlighted by the orange circle have almost the same *pos* weight and *neg* weight in GFL and VFL, while DW can distinguish them by assigning them different weights, providing the network higher learning capacity.

Limitation of DW. Although DW can well distinguish the importance of different anchors for an object, it will decrease the number of training samples at the same time, as shown in Fig 5. This may affect the training efficacy on small objects. As shown in Table 7, the improvement of DW on small objects is not as high as that on large objects. To mitigate this issue, we may dynamically set different hyper-parameters of w_{pos} based on object size to balance the training samples between small and large objects.

5. Conclusion

We proposed an adaptive label assignment scheme, named dual weighting (DW), to train accurate dense object detectors. DW broke the convention of coupled weighting in previous dense detectors, and it dynamically assigned individual *pos* and *neg* weights for each anchor by estimating the consistency and inconsistency metrics from different aspects. A new box refinement operation was also developed to directly refine boxes on the regression map. DW was highly compatible with the evaluation metric. Experiments on the MS COCO benchmark verified the effectiveness of DW under various backbones. With and without box refinement, DW with ResNet-50 achieved 41.5 AP and 42.2 AP, respectively, recording new state-of-the-art. As a new label assignment strategy, DW also demonstrated good generalization performance to different detection heads.

Negative societal impacts of object detection mainly arise from the abuse on military applications and privacy issues, which warrants careful consideration before applying this technology to real life.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 1

[2] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010. 6

[3] Yuhang Cao, Kai Chen, Chen Change Loy, and Dahua Lin. Prime sample attention in object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11583–11591, 2020. 2

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 1, 2

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 6

[6] Xiyang Dai, Yinpeng Chen, Jianwei Yang, Pengchuan Zhang, Lu Yuan, and Lei Zhang. Dynamic detr: End-to-end object detection with dynamic attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2988–2997, 2021. 1, 2

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 6

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. 2021. 1

[9] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R Scott, and Weilin Huang. Tood: Task-aligned one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3510–3519, 2021. 2, 3, 7, 8

[10] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. *arXiv preprint arXiv:2101.07448*, 2021. 1

[11] Ziteng Gao, Limin Wang, and Gangshan Wu. Mutual supervision for dense object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3641–3650, 2021. 2, 3, 5, 7, 8

[12] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 303–312, 2021. 2, 3, 7, 8

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1, 2

[14] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 1

[15] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017. 1

[16] Wei Ke, Tianliang Zhang, Zeyi Huang, Qixiang Ye, Jianzhuang Liu, and Dong Huang. Multiple anchor learning for visual object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10206–10215, 2020. 2, 7

[17] Kang Kim and Hee Seok Lee. Probabilistic anchor assignment with iou prediction for object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 355–371. Springer, 2020. 2, 7, 8

[18] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, Lei Li, and Jianbo Shi. Foveabox: Beyound anchor-based object detection. *IEEE Transactions on Image Processing*, 29:7389–7398, 2020. 1, 2, 7

[19] Hengduo Li, Zuxuan Wu, Chen Zhu, Caiming Xiong, Richard Socher, and Larry S Davis. Learning from noisy anchors for one-stage object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10588–10597, 2020. 2, 7

[20] Xiang Li, Wenhai Wang, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11632–11641, 2021. 2, 7, 8

[21] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *arXiv preprint arXiv:2006.04388*, 2020. 1, 2, 3, 7, 8

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 1, 2, 3, 5

[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 *ECCV*, pages 740–755. Springer, 2014. 2, 6

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer, 2016. 1, 2

[25] Yang Liu, Xu Tang, Junyu Han, Jingtuo Liu, Dinger Rui, and Xiang Wu. Hambox: Delving into mining high-quality anchors on face detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13043–13051. IEEE, 2020. 2

[26] Yuchen Ma, Songtao Liu, Zeming Li, and Jian Sun. Iqdet: Instance-wise quality distribution sampling for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1717–1725, 2021. 2, 8

[27] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3651–3660, 2021. 1, 2

[28] Qi Ming, Zhiqiang Zhou, Lingjuan Miao, Hongwei Zhang, and Linhao Li. Dynamic anchor learning for arbitrary-oriented object detection. *arXiv preprint arXiv:2012.04150*, 2020. 2

[29] Han Qiu, Yuchen Ma, Zeming Li, Songtao Liu, and Jian Sun. Borderdet: Border feature for dense object detection. In *European Conference on Computer Vision*, pages 549–564. Springer, 2020. 1

[30] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. 1

[31] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 1

[32] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, pages 91–99, 2015. 1, 2, 3, 6

[33] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 658–666, 2019. 3, 5

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1

[35] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017. 1

[36] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 1

[37] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE TPAMI*, pages 2818–2826, 2016. 1

[38] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, pages 9627–9636, 2019. 1, 2, 3, 5, 7, 8

[39] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 1

[40] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021. 1

[41] Jiahui Yu, Yuning Jiang, Zhangyang Wang, Zhimin Cao, and Thomas Huang. Unitbox: An advanced object detection network. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 516–520, 2016. 3

[42] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021. 1

[43] Haoyang Zhang, Ying Wang, Feras Dayoub, and Niko Sunderhauf. Varifocalnet: An iou-aware dense object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8514–8523, 2021. 1, 2, 3, 7, 8

[44] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9759–9768, 2020. 2, 3, 7, 8

[45] Xiaosong Zhang, Fang Wan, Chang Liu, Xiangyang Ji, and Qixiang Ye. Learning to match anchors for visual object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 3

[46] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. 1

[47] Benjin Zhu, Jianfeng Wang, Zhengkai Jiang, Fuhang Zong, Songtao Liu, Zeming Li, and Jian Sun. Autoassign: Differentiable label assignment for dense object detection. *arXiv preprint arXiv:2007.03496*, 2020. 3, 5, 7, 8

[48] Chenchen Zhu, Fangyi Chen, Zhiqiang Shen, and Marios Savvides. Soft anchor-point object detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 91–107. Springer, 2020. 1

[49] Chenchen Zhu, Yihui He, and Marios Savvides. Feature selective anchor-free module for single-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 840–849, 2019. 1

[50] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2020. 1, 2