Is Sampling Heuristics Necessary in Training Deep Object Detectors?

Joya Chen[1], Dong Liu[1], Tong Xu[1], Shiwei Wu[1], Shilong Zhang[1], Bin Luo[2], Xuezheng Peng[2], and Enhong Chen[1]

[1] University of Science and Technology of China [2] Tencent TaiQ Team
{chenjoya,zsl1996,dwustc}@mail.ustc.edu.cn
{dongeliu,tongxu}@ustc.edu.cn {luobinluo,reuspeng}@tencent.com

Abstract. In training deep object detectors, the problem of foreground-background imbalance has been addressed by several heuristic methods, such as online hard example mining (OHEM), Focal Loss, and gradient harmonizing mechanism (GHM). These methods either re-sample the training examples (*i.e.* hard sampling), or re-weight them discriminatively (*i.e.* soft sampling). In this paper, we challenge the necessity of such hard/soft sampling heuristics in training deep object detectors. First, without hard/soft sampling, we reveal that the scale and the stability of the classification loss greatly influence the final accuracy. Thus, we propose a guided loss scaling technique to control the classification loss during training, without using any hyper-parameter. We also propose to optimally initialize the model to ensure the stability of the classification loss. Moreover, we propose an adaptive thresholding technique to refine predictions during inference. These three ingredients constitute our *Sampling-Free* mechanism, which is fully data diagnostic and avoids the laborious hyper-parameter search for hard/soft sampling. We verify the effectiveness of our Sampling-Free mechanism in training one-stage, two-stage, multi-stage, and anchor-free object detectors, where our method always achieves higher accuracy on COCO and PASCAL VOC datasets. We also use the Sampling-Free mechanism for instance segmentation to demonstrate its generalization ability. Code is released at: `https://github.com/ChenJoya/sampling-free`.

1 Introduction

With the development of deep learning [21,23], recent years have witnessed remarkable advancement in deep object detectors. Among them, representative successes include two-stage R-CNN object detectors [1,5,12,14,25,26,32,34,40]: their first stage uses a region proposal network (RPN) to generate some candidates from dense, predefined bounding-boxes (*i.e.* anchors [40]), then the second stage uses a region-of-interest subnetwork (RoI-subnet) for object classification and localization. To pursue higher computational efficiency, one-stage approaches [20,27,30,37,38,39,50,53] directly recognize objects from the dense anchors without generating candidate proposals. Both two-stage and one-stage detectors adopt the anchoring scheme, where massive anchors ($\sim 100k$) are uniformly sampled over an image.

Nevertheless, when training these detectors, only a few anchors (~ 100) that highly overlap with objects will be assigned to foreground examples, which always results in an extreme imbalance between foreground and background within the anchors. In previous studies [24,27,30,40,42], such imbalance may impede the training from convergence, and limit the accuracy as well. More recently, anchor-free object detectors [6,16,19,22,43,46,49,51,52] have gained much attention due to the replacement of anchors by points or regions (*e.g.* corner points or central regions), but they still suffer from the imbalance between foreground and background within points or regions.

To address the foreground-background imbalance, several heuristic methods have been proposed in recent years. These methods can be divided into two categories. The first category selects a subset of training examples, e.g. mini-batch biased sampling [40], online hard example mining [42] (OHEM), IoU-balanced sampling [34]. The second category assigns different weights to the samples discriminatively, like Focal Loss [27], gradient harmonizing mechanism [24] (GHM). According to [17], these two categories can be named "hard sampling" and "soft sampling," respectively. Thus, we will use the term "sampling heuristics" to refer to these solutions in the following.

Unlike a common class imbalance that introduced by biased dataset [47], the foreground-background imbalance should be attributed to the large searching space of object detectors, thus it equally exists in training and inference with the same distribution. Sampling heuristics is only used during training but not during inference, thereby resulting in a misalignment between training and inference. Moreover, the existing heuristic sampling methods introduce hyper-parameters that need to be carefully tuned. For example, in [24], the authors have mentioned that the optimal strategy for GHM is hard to define.

Can we discard sampling heuristics when training deep object detectors? In the past, it was believed [24,27] that the non-sampling object detector will suffer from extreme imbalance during training and yield very low accuracy. Although several objectness modules [20,39,50] try to address the imbalance by reducing background examples in a cascade fashion, they incur extra computational and memory costs, and most of them still use the sampling heuristics. Therefore, it seems no possibility to train a non-sampling object detector to match the accuracy of sampling-based one.

In this paper, we successfully discard sampling heuristics in training object detectors but achieve better accuracy. Our study reveals that without sampling heuristics, the scale and the stability of the classification loss greatly influence the final accuracy. This motivates us to propose a *Sampling-Free* mechanism, which consists of three ingredients: (1) guided loss scaling for controlling the scale of the classification loss, (2) optimal bias initialization for ensuring the stability of the classification loss, (3) adaptive thresholding for refining predictions during inference. As shown in Fig. 1, unlike sampling heuristics, our method treats all training examples equally, thus no hyper-parameter is introduced in our method for hard sampling or soft sampling.
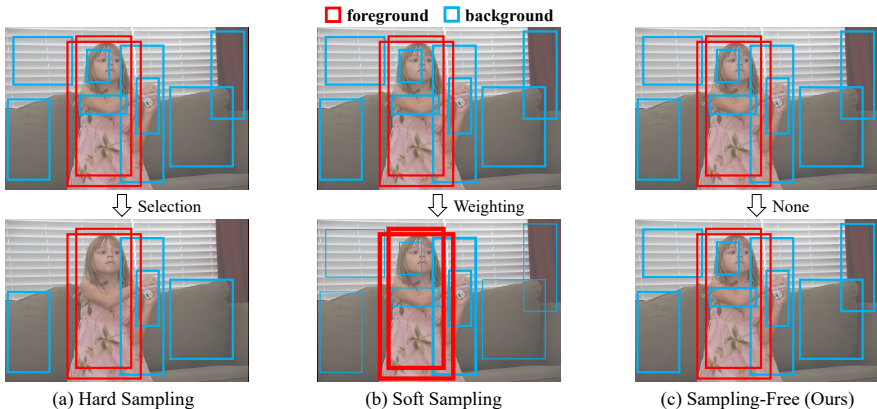
Fig. 1. This figure illustrates the differences between sampling heuristics and the Sampling-Free mechanism for addressing the foreground-background imbalance. (a) Hard sampling (*e.g.* mini-batch biased sampling [40], OHEM [42], IoU-balanced sampling [34]) selects a subset of training examples; (b) Soft sampling (*e.g.* Focal Loss [27], GHM [24]) uses all training examples but focuses on some of them by weighting. For instance, thicker boxes in (b) denote training examples with higher weights. (c) Sampling-Free equally uses all training examples.

Experimental results on COCO [28] and PASCAL VOC [7] datasets have demonstrated that our method improves the accuracy of one-stage, two-stage, multi-stage, and anchor-free object detectors. Moreover, it can also yield considerable gains for the instance segmentation task. There is no hyper-parameter introduced in our method, which avoids laborious tuning required in sampling heuristics. Our Sampling-Free mechanism provides a new perspective to address the foreground-background imbalance.

2   Related Works

Deep Object Detectors. Classical object detectors [8,45] mostly rely on a paradigm of hand-crafted feature extraction, which usually requires carefully designing. With the development of deep learning [21,23], deep object detectors quickly come to dominate the object detection. Among them, two-stage approaches lead the top accuracy on several object detection benchmarks, such as PASCAL VOC [7] and COCO [28]. It firstly generates some object candidates by region proposal stage [40,44,54], then determines the accurate object location and category by a RoI-subnet. A large number of R-CNN variants [1,5,12,14,25,26,32,34,40] appear over the years, yielding a large improvement in detection performance.

On the other hand, one-stage detectors, which are popularized by YOLO [37] and SSD [30], are much faster than two-stage approaches due to the elimination of the RPN but have difficulties to match the accuracy of two-stage approaches.

A series of advances [20,27,38,39,50,53] in recent years promote one-stage frameworks to be more accurate. In practice, both one-stage and two-stage approaches rely on a dense anchoring scheme to cover objects, which are also known as anchor-based object detectors. For simplicity, several anchor-free object detectors [6,19,22,43,46,49,51,52] originated from DenseBox [16] are proposed recently, which achieves better accuracy and computational efficiency than anchor-based object detectors.

Class Imbalance. The class imbalance problem has always been an issue in machine learning applications. In such a problem, some classes have more instances than others, where the standard models tend to be overwhelmed by the majority class. Popular solutions for addressing the imbalance can be divided into three aspects: (1) threshold moving: using a lower decision threshold for preserving more minority class [4]; (2) sampling heuristics: a biased selection for specific classes, including undersampling (*e.g.* EasyEnsemble [31], BalanceCascade [31]) and oversampling (*e.g.* SMOTE [2]), and the combination of them (*e.g.*, RUSBoost [41]). In practice, some of them [10] also adopt the ensemble technique to obtain a more robust model.

Sampling Heuristics in Object Detection. Due to the numerous background examples (boxes/points/regions) during training, modern object detectors always suffer from an extreme imbalance between the foreground and the background, which results in the popularity of sampling heuristics. In [17], the authors summarize the sampling heuristics into two groups: (1) hard sampling methods, such as mini-batch biased sampling [40], OHEM [42], and IoU-balanced sampling [34], are in common use in two-stage detectors; (2) soft sampling methods, such as Focal Loss [27], GHM [24], have been widely adopted in one-stage and anchor-free object detectors.

However, sampling heuristics may not be the optimal strategy in all imbalanced cases. In [33], it has been demonstrated that for the metric of the ROC curve, sampling methods produce the same result as moving the inference threshold or adjusting the cost matrix. Moreover, the common class imbalance is usually introduced by the biased dataset, whereas the foreground-background imbalance in object detection is caused by the large searching space of detection frameworks, which equally exists in training and inference. Furthermore, they are heuristic and usually require laborious hyper-parameters tuning. These observations motivate us to explore a Sampling-Free method.

Multi-Task Weighting. A simple way to weigh different tasks is to tune the weights of their loss functions. In recent years, several adaptive weighting methods are proposed. For example, Guo et al. [13] propose to weigh the losses dynamically based on a predefined key performance indicator (*e.g.* accuracy, average precision) for each task. Other methods can use the network outputs to weigh tasks, such as the uncertainty of the estimations [18] or their loss values [29]. The proposed Sampling-Free mechanism controls the scale of the classification loss to address the foreground-background imbalance, which seems not reported before in the literature, to our best knowledge.

3 Methodology

In this section, we will introduce our Sampling-Free mechanism starting from investigating the challenges of training the deep object detector without sampling heuristics (also called "training the non-sampling model"). As our investigation will reveal, the scale and the stability of the classification loss are the keys to train the non-sampling model. Motivated by this, we will propose our method to ensure stable training and control the classification loss.

3.1 Investigation for Training the Non-Sampling Model

We investigate the well-known soft sampling method Focal Loss [27], which is widely used in the one-stage (anchor-free) object detectors [19,22,43,49,52,53]. In previous studies [24,27], Focal Loss helps RetinaNet [27] to yield $4 \sim 7$ higher average precision (AP) on COCO [28] than the standard cross-entropy loss (CE Loss), a non-sampling loss function. Therefore, we will experimentally explore what causes this gap in accuracy. For simplicity, we denote the RetinaNet with Focal Loss and CE Loss as *RetinaNet-FL* and *RetinaNet-CE*, respectively.

There are two differences between *RetinaNet-FL* and *RetinaNet-CE*, one of which is the classification loss. Specifically, *RetinaNet-FL* uses Focal Loss (L^{FL}) as the classification loss, whereas *RetinaNet-CE* uses CE Loss (L^{CE}) as the classification loss:

$$L^{FL}(\alpha, \gamma) = -\frac{1}{N_f} \sum_{i=1}^{N} \sum_{j=1}^{C} [\mathbf{1}_{y^i=j} \alpha (1 - p_{ij})^\gamma \log(p_{ij})$$
$$+ \mathbf{1}_{y^i \neq j} (1 - \alpha) p_{ij}{}^\gamma \log(1 - p_{ij})], \tag{1}$$

$$L^{CE} = -\frac{1}{N_f} \sum_{i=1}^{N} \sum_{j=1}^{C} [\mathbf{1}_{y^i=j} \log(p_{ij}) + \mathbf{1}_{y^i \neq j} \log(1 - p_{ij})]. \tag{2}$$

Here p_{ij} is the confidence score of j-th ($j \in [1, C]$) class of anchor i ($i \in [1, N]$), where N is the number of training anchors, C is the number of object classes. y^i denotes the ground-truth label of anchor i, where $y^i = 0$ means the background label. N_f is number of foreground anchors, where $N \gg N_f$ is the reality in the foreground-background imbalance. $\mathbf{1}_K$ is an indicator function, which outputs 1 if the condition K is true, otherwise $\mathbf{1}_K = 0$. α and γ are the hyper-parameters to do soft sampling in Focal Loss.[1] In contrast, there is no hyper-parameters in L^{CE} to down-weight the large loss value of numerous background anchors, which always results in network diverging during training *RetinaNet-CE*. To avoid this, we multiply L^{CE} with a weight w (*i.e.* $wL^{CE}, w < 1$), and explore how to adjust w in the investigation.

---

[1] We use $\alpha = 0.25, \gamma = 2$ for the high accuracy of RetinaNet, as reported in [27].

Another difference between *RetinaNet-FL* and *RetinaNet-CE* is in the model initialization. Focal Loss uses a biased initialization that initializes the final classification convolutional layer with the bias $b = -\log\frac{1-\pi}{\pi}$. Then, at the beginning of the training, $p_{ij} = \pi$ is tenable for every training anchor as p_{ij} is computed by sigmoid activation. A heuristic value $\pi = 10^{-2}$ is used in Focal Loss to avoid network diverging. Unfortunately, this will result in network diverging during training *RetinaNet-CE*. We will also explore how to adjust π in the following.
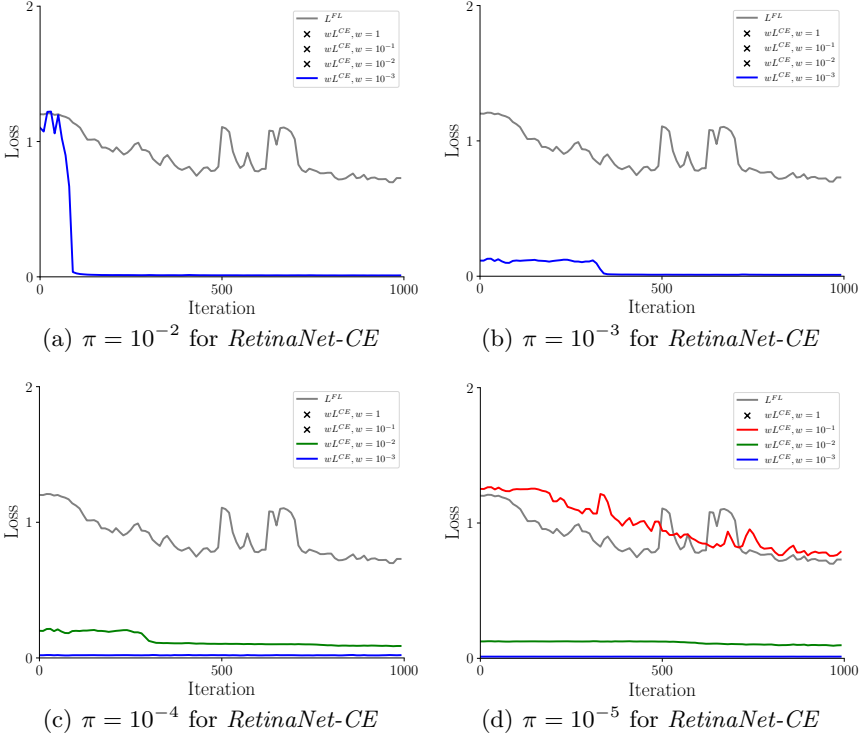
Fig. 2. Loss curves of L^{FL} and L^{CE} in different settings. "**X**" means the network diverging. The model is RetinaNet with ResNet-50-FPN [15,26] backbone, trained on COCO `train2017` [28] with $1\times$ learning schedule [11] (12 epochs), implemented by `maskrcnn-benchmark` [9]. We only show the first $1k$ iterations for better visualization.

Stability of Classification Loss. Our first discovery is that the non-sampling model has poor stability on the classification loss. As shown in Fig. 2(a), training *RetinaNet-CE* without any modifications to π and w fails quickly, with the network diverging in a few iterations. However, the curve of L^{FL} in Fig. 2 shows that *RetinaNet-FL* can be stably trained. We observe that L^{CE} is too large to be stably trained. Therefore, we reduce L^{CE} by tuning w and π. Fortunately, we can obtain converged models in some specific settings, as shown in Fig. 2.

(a) Accuracy of *RetinaNet-CE* in different π, w

AP	$w = 1$	$w = 10^{-1}$	$w = 10^{-2}$	$w = 10^{-3}$
$\pi = 10^{-2}$	n/a	n/a	n/a	0.8
$\pi = 10^{-3}$	n/a	n/a	n/a	0.8
$\pi = 10^{-4}$	n/a	n/a	26.0	0.8
$\pi = 10^{-5}$	n/a	**35.6**	26.1	0.8

(b) Accuracy comparison between *RetinaNet-FL* and *RetinaNet-CE*

Model	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet-FL	**36.4**	**55.0**	**39.0**	**19.9**	**40.3**	**48.9**
RetinaNet-CE ($\pi = 10^{-5}, w = 10^{-1}$)	35.6	53.9	38.2	18.6	39.2	47.4

Table 1. Accuracy of *RetinaNet-CE* in different w and π, evaluated on COCO `minival`. "n/a" refers to the network diverging. Results are achieved by the models in Fig. 2.

Scale of Classification Loss. Our second discovery is that the scale of the classification loss will greatly influence the accuracy of the non-sampling model. As shown in Table 1(a), if a model can be stably trained, then its accuracy will be similar (± 0.1 AP) in the same w, but different w leads to great changes in AP. Table 1(b) shows that *RetinaNet-CE* with $\pi = 10^{-5}, w = 10^{-1}$ has already achieved the comparable accuracy of *RetinaNet-FL* (35.6 AP vs. 36.4 AP). This is an inspiring result, as previous works [24,27] reported there is a $4 \sim 7$ AP gap between CE Loss and Focal Loss. Our investigation successfully narrows this gap. However, it is still unknown why adjusting π and w can help to train the non-sampling model. Next, we will perform analysis for this.

Analysis. At the beginning of the training, $p_{ij} = \pi$ is tenable, which can be used to reformulate L^{FL} and wL^{CE}:

$$L^{FL}(\alpha, \gamma) = -\alpha(1 - \pi)^\gamma \log(\pi) - (1 - \alpha)\pi^\gamma (\frac{NC}{N_f} - 1) \log(1 - \pi)], \qquad (3)$$

$$wL^{CE} = -w \log(\pi) - w(\frac{NC}{N_f} - 1) \log(1 - \pi). \qquad (4)$$

Here some symbols can be assigned with specific values. Specifically, $C = 80$ denotes 80 object classes on COCO [28]. $\frac{N}{N_f}$ represents the all-to-foreground ratio, which is $\frac{N}{N_f} \approx 10^3$ for RetinaNet on COCO. $\alpha = 0.25, \gamma = 2, \pi = 10^{-2}$ are used for training *RetinaNet-FL*. With these specific values, we can compute $L^{FL} \approx 1.19$, which matches the initial value of L^{FL} during training *RetinaNet-FL* (see Fig. 2). We observe that $L^{CE} \approx 1.23$ in $w = 10^{-1}, \pi = 10^{-5}$ is more similar value with $L^{FL} \approx 1.19$ than other settings. This may be the reason why *RetinaNet-CE* with $w = 10^{-1}, \pi = 10^{-5}$ achieves comparable AP of *RetinaNet-FL*. We believe that the deeper reason is that this setting maintains a balance between the classification loss (L^{FL} or L^{CE}) and the bounding-box regression

(a) *RetinaNet-FL*   (b) *RetinaNet-CE* ($\pi = 10^{-5}$)
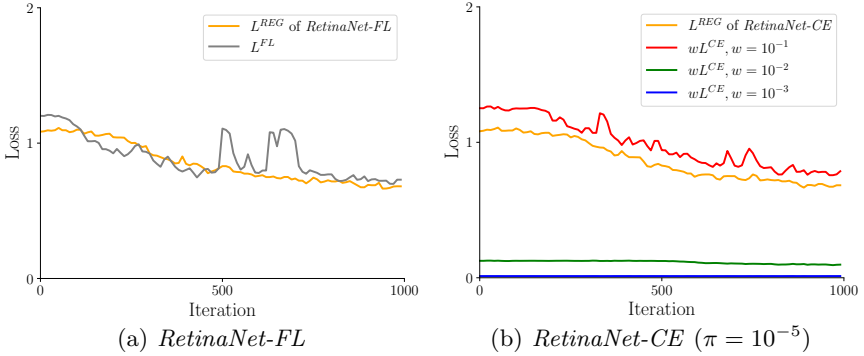
Fig. 3. Loss curves of bounding-box regression (L^{REG}) and classification (L^{FL} or L^{CE}), which are from the same models in Fig. 2. Note that the curves of L^{REG} with different w in (b) are very similar, thus we only show the curve of L^{REG} with $w = 10^{-1}$ in (b).

loss (L^{REG}). As shown in Fig. 3(a), *RetinaNet-FL* achieves a quite good balance between L^{REG} and L^{FL}, whereas Fig. 3(b) shows that only *RetinaNet-CE* with $\pi = 10^{-5}, w = 10^{-1}$ can obtain this balance. Other settings, however, will lead the training to be dominated by the bounding-box regression task, which is not good for training an accurate object detector.

3.2   Sampling-Free Mechanism

Inspired by the investigation, we propose a novel Sampling-Free mechanism, which addresses the foreground-background imbalance from the initialization and the loss scale. Our principle is to pursue a simple yet effective solution, thus we will not introduce extra hyper-parameters to address the imbalance.

Guided Loss Scaling. Usually, the loss function to train a deep object detector is composed of a bounding-box regression loss term and a classification loss term, where the former is for the object location and the latter is for the object category. Let Θ denote the trainable parameters of the detector, we have

$$\Theta^* = \arg\min_{\Theta} L(\Theta), \text{where } L(\Theta) = L^{REG}(\Theta) + wL^{CLS}(\Theta). \quad (5)$$

Here w is a pre-defined weighting hyper-parameter. According to the stochastic gradient descent (SGD), we have

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta(\frac{\partial L}{\partial \Theta})^{(t)}, \quad (6)$$

where t indexes the training iterations and η is the learning rate, the gradient $\frac{\partial L}{\partial \Theta}$ is computed per mini-batch. As mentioned in Sec. 3.1, when we use L^{CE} as L^{CLS}, it is essential to control its scale. A straightforward way is to adjust the weight w. However, it results in a new hyper-parameter.

```
import torch
def guided_loss(box_cls_loss, box_reg_loss):
    # box_cls_loss: classification loss (per-batch)
    # box_reg_loss: bounding-box regression loss (per-batch)
    with torch.no_grad():
        w = box_reg_loss / box_cls_loss
    box_cls_loss *= w
    return box_cls_loss
```

Fig. 4. Python code of Guided Loss Scaling based on PyTorch [35].

Our key idea is to adjust w dynamically during training. That says, instead of using a constant w, we define

$$(\frac{\partial L}{\partial \Theta})^{(t)} = (\frac{\partial L^{REG}}{\partial \Theta})^{(t)} + w^{(t)}(\frac{\partial L^{CE}}{\partial \Theta})^{(t)}, \quad (7)$$

where

$$w^{(t)} = \frac{L^{REG}(\Theta^{(t)})}{L^{CE}(\Theta^{(t)})}. \quad (8)$$

In other words, we use the regression loss value of the current mini-batch as the "target" of the rescaled classification loss value. Thus, this technique is termed guided loss scaling. It can be interpreted threefold. First, based on the study in Sec. 3.1 (especially Fig. 3), it appears a good choice to let the regression loss value and the classification loss value be similar, where the classification loss is either Focal Loss or CE Loss. Second, it is convenient to use the regression loss as the guidance (see Fig. 4), because the regression loss is already there for object detection. Third, classification loss like CE Loss is greatly influenced by the foreground-background imbalance, but regression loss is little influenced as it is computed merely for foreground anchors. Thus, the regression loss is helpful to combat the instability due to the imbalance.

Optimal Bias Initialization. Sec. 3.1 has shown that adjusting π can help to avoid network diverging. However, it is difficult to determine π. We propose Optimal Bias Initialization to compute π from data statistics rather than tuning it. Our idea is to initialize the bias of the last convolutional layer to obtain the minimal classification loss. The derivative of L^{CE} in Eq. 4 is:

$$\frac{\partial L^{CE}}{\partial \pi} = -\frac{1}{\pi} + (\frac{NC}{N_f} - 1)\frac{1}{1-\pi}, \quad (9)$$

where $\pi = \frac{N_f}{NC}$ leads to $\frac{\partial L^{CE}}{\partial \pi} = 0$, and L^{CE} will attain the minimal value. Therefore, we can obtain the optimal initial bias for sigmoid activation:

$$b = -\log\frac{1-\pi}{\pi} = -\log(\frac{NC}{N_f} - 1). \quad (10)$$

(a) Varying θ for *RetinaNet-FL*   (b) Varying θ for *RetinaNet-CE*

Threshold	AP	AP$_{50}$	AP$_{75}$	Inference Speed
$\theta = 0.05$	36.4	55.0	39.0	114 ms
$\theta = 0.01$	36.4	55.0	39.0	115 ms
$\theta = 0$	36.4	55.0	39.0	198 ms
$\theta = \frac{N_f}{N}$	36.4	55.0	39.0	140 ms

Threshold	AP	AP$_{50}$	AP$_{75}$	Inference Speed
$\theta = 0.05$	35.6	53.9	38.2	108 ms
$\theta = 0.01$	36.2	54.7	38.6	109 ms
$\theta = 0$	**36.3**	54.8	38.7	197 ms
$\theta = \frac{N_f}{N}$	**36.3**	54.8	38.7	114 ms

Table 2. Varying inference threshold θ for *RetinaNet-FL* and *RetinaNet-CE*. The inference speed is measured on a single `Nvidia-Titan Xp` GPU, with batch size 1.

Here $\frac{N}{N_f}$ can be computed by pre-defined anchors, thus the computation is efficient as it does not require network forwarding. With $\frac{N}{N_f} \approx 10^3, C = 80$, we can obtain the optimal $\pi \approx 1.25 \times 10^{-5}$, which is similar with $\pi = 10^{-5}$ that performs best in our experiments. It is worth noting that the accuracy of the model is robust to our initialization strategy, as the model can "utilize" the imbalanced distribution to obtain a lower loss (see the sudden drop of L^{CE} in Fig. 2(a)). We initialize the model to ensure the stability of the classification loss.

Adaptive Thresholding. As *RetinaNet-CE* does not down-weight any background example during training, it will be more inclined to predict unknown examples as background to converge to a low loss value. If we use the same inference threshold ($\theta = 0.05$) in *RetinaNet-FL*, some correct predictions of *RetinaNet-CE* will be incorrectly removed. A simple way is to set $\theta = 0$ to obtain the maximum AP, but this will incur more computations in inference due to the post-processing. We propose to adaptively set it according to the imbalanced statistics. The ratio of $\frac{N_f}{N}$ is the foreground-to-all ratio, which can represent the predictive bias for foreground examples. Therefore, it is suitable to use it as the inference threshold. As shown in Table 2, our method improves the AP and maintains the speed, with the elimination of tuning the inference threshold.

Applications to Different Detectors. We have introduced our method for training the one-stage object detector. It is also applicable for anchor-free approaches that we only need to regard the concept of "anchors" as "regions" or "points". For two-stage [14,40] approaches, we can replace the mini-batch biased sampling[2] with Sampling-Free in both RPN and RoI-subnet, using all training examples. That is to say, we train RPN with all training anchors ($\sim 100k$ per-image) and train RoI-subnet with all proposals ($\sim 1k$ per-image). For multi-stage approaches [1], we regard the multiple RoI-subnets as one RoI-subnet. It is worth noting two points: (1) We observe that the bounding-box regression loss is ~ 2 times higher than the classification loss in RoI-subnet of two-stage/multi-stage approaches. Therefore, we multiply the computed w in the guided loss scaling with 2 to maintain the consistency. (2) As the imbalance in RoI-subnet has been greatly alleviated by RPN (*e.g.*, from $1 : 1k$ to $1 : 10$), the optimal bias initialization is not used in RoI-subnet.

---

[2] A common setting of mini-batch biased sampling in training Faster R-CNN: (1) RPN randomly selects 256 anchors with 1:1 foreground-to-background ratio, (2) RoI-subnet randomly selects 512 proposals with 1:3 foreground-to-background ratio.

4   Experiments

In this section, we will validate our Sampling-Free mechanism on object detection and instance segmentation tasks. Before that, we will describe the implementation details, and perform ablation studies for one-stage and two-stage object detectors. Finally, we will report the improvements for various object detectors. Here we mainly show the experimental results on the challenging COCO [28] benchmark, with the comparison to the popularized sampling heuristics (mini-batch biased sampling, Focal Loss, GHM). *More results including the performance on PASCAL VOC, different backbone and learning schedule, the comparison to other sampling heuristics are provided in the supplementary material.*

4.1   Implementation Details

Datasets and Metrics. We use the well-known COCO [28] benchmark to validate our method. Following the common practice [27,40], we train models on the `train2017` split and perform ablation studies on `minival` split, then submit results to the public server of `test-dev` split to report performance. COCO-style average precision (AP) is used as the evaluation metrics.

Baselines and Platforms. We use six object detectors for experiments:

- YOLOv3 [39] (one-stage) is an efficient object detector that uses the objectness module to address the imbalance. In our experiments, we will replace its objectness module with the Sampling-Free mechanism to improve it.

- RetinaNet [27] (one-stage) is an accurate one-stage object detector, which applies Focal Loss to address the imbalance. With the Sampling-Free mechanism, we will show that the standard CE Loss works better than Focal Loss.

- Faster R-CNN [40] (two-stage) is one of the most classical object detectors, with mini-batch biased sampling used in the RPN and RoI-subnet. We will validate whether the Sampling-Free mechanism is effective for the two stages.

- FoveaBox [19] (anchor-free) is a simple and yet effective anchor-free object detector, using Focal Loss to address the imbalance. Like RetinaNet, we will show that Sampling-Free works well in that.

- Mask R-CNN [14] (two-stage) is a simple, flexible, and general framework for object instance segmentation, with mini-batch biased sampling to address the imbalance in RPN and RoI-subnet. We will show that Sampling-Free can help it to achieve better box AP and mask AP.

- Cascade R-CNN [1] (multi-stage) has a cascaded RoI-subnet pipeline to achieve high-quality object detection, with mini-batch biased sampling to address the imbalance in RPN and all RoI-subnets. Using the Sampling-Free mechanism in all stages, we will show that Cascade R-CNN will be more accurate.

Among them, YOLOv3 is implemented on `darknet` [36]; RetinaNet and Faster/Mask R-CNN are implemented on `maskrcnn-benchmark` [9]; FoveaBox are implemented on `mmdetection` [3], and Cascade R-CNN is implemented on `detectron2` [48]. We follow the public standard training configurations to implement them, which means that we have not made any changes for the hyper-parameters of them.

	Method	RetinaNet (ResNet-50-FPN, 1×)					
Sampling-Free	Optimal Bias Initialization	✗	✗	✓	✓	✓	✓
	Guided Loss Scaling	✗	✓	✗	✓	✓	✓
	Adaptive Thresholding	✗	✓	✓	✗	✓	✓
Focal Loss		✓	✗	✗	✗	✗	✓
AP		36.4	n/a	n/a	36.4	**36.8**	36.7
AP_{50}		55.0	n/a	n/a	54.0	54.9	55.9
AP_{75}		39.0	n/a	n/a	39.3	39.7	39.3
AP_S		19.9	n/a	n/a	20.1	20.6	20.6
AP_M		40.3	n/a	n/a	39.8	40.3	40.3
AP_L		48.9	n/a	n/a	47.9	48.3	49.0

Table 3. Ablations of each ingredient of our Sampling-Free mechanism on COCO `minival`. Better AP can be achieved by replacing Focal Loss with Sampling-Free, but the hybrid of Sampling-Free and Focal Loss decreases the accuracy.

Method	Faster R-CNN (ResNet-50-FPN, 1×)													
	RPN	RoI	RPN	RoI	RPN	RoI	RPN	RoI	RPN	RoI	RPN	RoI	RPN	RoI
Mini-Batch Biased Sampling	✓	✓	✗	✓	✓	✗	✗	✗	✓	✓	✓	✓	✓	✓
Sampling-Free	✗	✗	✓	✗	✗	✓	✓	✓	✗	✗	✗	✓	✓	✓
AP	36.8		37.5 (+0.7)	38.1 (+1.2)	**38.4 (+1.6)**		36.5 (-0.3)		36.8 (+0.0)		36.4 (-0.4)			
AP_{50}	58.4		59.0 (+0.6)	59.6 (+1.2)	**59.9 (+1.5)**		58.2 (-0.2)		58.7 (+0.3)		57.9 (-0.5)			
AP_{75}	40.0		40.4 (+0.4)	41.6 (+1.6)	**41.7 (+1.7)**		39.6 (-0.4)		40.0 (+0.0)		39.4 (-0.6)			
AP_S	20.7		21.5 (+0.8)	22.2 (+1.5)	**22.3 (+1.6)**		21.2 (+0.5)		21.1 (+0.4)		21.1 (+0.4)			
AP_M	39.7		40.7 (+1.0)	41.2 (+1.5)	**41.6 (+1.9)**		39.4 (-0.3)		40.0 (+0.3)		39.1 (-0.6)			
AP_L	47.9		48.8 (+0.9)	50.0 (+2.1)	**50.9 (+3.0)**		47.6 (-0.3)		47.8 (-0.1)		47.5 (-0.4)			

Table 4. Results of our Sampling-Free mechanism for Faster R-CNN on COCO `minival`. Better AP can be achieved by replacing sampling heuristics with Sampling-Free, but the hybrid of Sampling-Free and sampling heuristics decreases the accuracy.

4.2 Ablation Studies

The Effectiveness of Each Ingredient. Table 3 shows the effectiveness of each ingredient in Sampling-Free. It illustrates that the optimal bias initialization and the guided loss scaling should be synchronously used to avoid the network diverging. The RetinaNet with Sampling-Free achieves 0.4 AP higher than its Focal Loss counterpart. But more importantly, our method successfully bridges the $4 \sim 7$ AP gap between CE Loss and Focal Loss in previous works [24,27].

Sampling-Free for RPN and RoI-subnet. As shown in Table 4, the vanilla Faster R-CNN achieves 36.8 AP, which uses mini-batch biased sampling in both RPN and RoI-subnet. We gradually incorporate Sampling-Free into RPN and RoI-subnet, then we observe that better accuracy can be always obtained by replacing mini-batch biased sampling with Sampling-Free. If we use Sampling-Free in both RPN and RoI-subnet, an impressive 1.6 AP improvement can be obtained, with the gains from all AP metrics.

Incompatibility with Sampling Heuristics. Our Sampling-Free mechanism focuses on the scale and the stability of the classification loss to address the imbalance, thus it is technically feasible to synchronously use that with sampling heuristics. Unfortunately, it seems that Sampling-Free is incompatible with sampling heuristics. As shown in Table 3, 0.1 AP degradation is incurred when we use Sampling-Free with sampling heuristics. This degradation also occurs in the two-stage approach (see Table 4): using both Sampling-Free and sampling heuristics will not achieve better results, whether for RPN or RoI-subnet.

Model	Method	AP	Training Speed (ms)	Memory (MB)
RetinaNet	Focal Loss	36.4	171	1816
(ResNet-50-FPN, 1×)	Sampling-Free	**36.8**	**165**	1816
Faster R-CNN	Mini-Batch Biased Sampling	36.8	**172**	1714
(ResNet-50-FPN, 1×)	Sampling-Free	**38.4**	184	**1669**

Table 5. Training speed and memory cost of our Sampling-Free mechanism.

Bounding-Box Regression Loss	Classification Loss	AP	Training Speed
Smooth L1 [12]	GHM-C	35.8	328 ms
	Sampling-Free	**36.0**	**238 ms**
GHM-R	GHM-C	36.9	336 ms
	Sampling-Free	**37.1**	**244 ms**

Table 6. GHM [24] and Sampling-Free on COCO `minival` (ResNet-50-FPN, 1×). Implementation is based on the code of GHM in `mmdetection` [3].

Training Speed and Memory Cost. We show the results of training speed and memory cost of Sampling-Free in Table 5, which is measured on a single `Nvidia-Titan Xp` GPU with batch size 1. Collaborated with Sampling-Free, training RetinaNet becomes faster than with Focal Loss, with the equivalent memory cost. Interestingly, although the training speed becomes slower for Faster R-CNN with Sampling-Free (as more background proposals are used in RoI-subnet), the memory cost is reduced, which is owing to that the operation of mini-batch biased sampling also requires considerable memory costs.

Comparison to GHM. We note that GHM [24] shows better accuracy than Focal Loss [27], which has two ingredients—GHM-C and GHM-R. GHM-C is designed for the classification loss, thus the comparison between GHM-C and Sampling-Free is fair. Table 6 shows that Sampling-Free achieves 0.2 AP higher than GHM-C, and it also produces a 27% acceleration during training.

Fig. 5. Mask R-CNN [14] (37.8 box AP, 34.2 mask AP on COCO `minival`) vs. Mask R-CNN with Sampling-Free (39.0 box AP, 34.9 mask AP on COCO `minival`) in ResNet-50-FPN backbone. The latter exhibits better detection and segmentation results.

(a) Object detection results of Sampling-Free on COCO `test-dev`

Method	Platform	Backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
YOLOv3 [39]	darknet	DarkNet-53	33.0	57.9	34.4	18.3	35.4	41.9
YOLOv3 (*Sampling-Free*)			**33.9**	**58.4**	**35.5**	**18.4**	**35.7**	**43.2**
RetinaNet [27]	maskrcnn-benchmark		36.6	**56.0**	39.1	19.5	38.9	**47.1**
RetinaNet (*Sampling-Free*)			**36.9**	55.1	**39.6**	**19.8**	**39.3**	46.7
FoveaBox [19]	mmdetection		36.7	56.6	39.2	20.3	39.9	45.3
FoveaBox (*Sampling-Free*)		ResNet-50-FPN	**36.8**	**56.8**	39.2	**20.6**	**40.2**	**45.4**
Faster R-CNN [40]	maskrcnn-benchmark		37.2	59.3	40.3	21.3	39.5	46.9
Faster R-CNN (*Sampling-Free*)			**38.7**	**60.5**	**42.2**	**22.2**	**41.0**	**49.1**
Cascade R-CNN [1]	detectron2		41.8	59.8	45.2	24.3	44.3	52.5
Cascade R-CNN (*Sampling-Free*)			**42.8**	**60.8**	**46.7**	**25.0**	**45.1**	**53.7**
RetinaNet [27]	maskrcnn-benchmark		38.8	**58.4**	41.7	20.9	41.7	49.5
RetinaNet (*Sampling-Free*)			**39.1**	57.7	**42.2**	**21.4**	**42.0**	**49.7**
FoveaBox [19]	mmdetection		38.4	58.2	41.1	**21.5**	42.0	47.2
FoveaBox (*Sampling-Free*)		ResNet-101-FPN	**38.6**	**58.6**	**41.3**	21.4	**42.2**	**47.6**
Faster R-CNN [40]	maskrcnn-benchmark		39.3	61.4	42.7	22.1	41.9	50.1
Faster R-CNN (*Sampling-Free*)			**40.7**	**62.4**	**44.3**	**23.0**	**43.5**	**51.8**
Cascade R-CNN [1]	detectron2		43.4	61.6	47.1	24.7	46.2	54.4
Cascade R-CNN (*Sampling-Free*)			**44.4**	**62.7**	**48.3**	**25.7**	**47.2**	**55.8**

(b) Box AP results on COCO `minival`   (c) Mask AP results on COCO `minival`

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Mask R-CNN [14]	37.8	59.3	41.1	21.5	41.1	49.9
w. Sampling-Free	**39.0**	**60.3**	**42.5**	**22.5**	**41.9**	**51.2**

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Mask R-CNN [14]	34.2	55.9	36.3	15.6	36.8	50.6
w. Sampling-Free	**34.9**	**56.8**	**37.1**	**16.2**	**37.3**	**51.2**

Table 7. Results of different detectors with our Sampling-Free mechanism on COCO.

4.3   Results

Object Detection. Table 7(a) shows the results of Sampling-Free for various object detectors. In both lightweight and heavy backbones, we observe that the detectors with Sampling-Free always yield better accuracy than the sampling-based ones. Among them, the two-stage and multi-stage approaches obtain large gains, where the Sampling-Free mechanism yields $1.4 \sim 1.5$ AP improvements for Faster R-CNN, 1.0 AP improvements for Cascade R-CNN, respectively. Although the gains in one-stage and anchor-free approaches are not impressive, it still demonstrates that sampling heuristics is unnecessary in training them.

Instance Segmentation. As shown in Table 7(b) and (c), Mask R-CNN with Sampling-Free achieves 1.2 box AP and 0.7 mask AP gains. We visualize the detection and segmentation results in Fig. 5, which shows that the Mask R-CNN with Sampling-Free exhibits better detection and segmentation results.

5   Conclusion

In this paper, we explore whether sampling heuristics is necessary for training object detectors. Our empirical study reveals that without sampling heuristics, the scale and the stability of the classification loss greatly influence the final accuracy. Inspired by this, we propose a novel Sampling-Free mechanism to control the classification loss, without new hyper-parameters introduced. Extensive experiments on PASCAL VOC and COCO demonstrate that Sampling-Free can help various object detectors to be more accurate. It also yields considerable gains for the instance segmentation task. Our Sampling-Free mechanism provides a new perspective to address the foreground-background imbalance.

References

1. Cai, Z., Vasconcelos, N.: Cascade R-CNN: delving into high quality object detection. In: CVPR. pp. 6154–6162 (2018)
2. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. JAIR **16**, 321–357 (2002)
3. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: Mmdetection: Open mmlab detection toolbox and benchmark. arXiv:1906.07155 (2019)
4. Collell, G., Prelec, D., Patil, K.R.: A simple plug-in bagging ensemble based on threshold-moving for classifying binary and multiclass imbalanced data. Neurocomputing **275**, 330–340 (2018)
5. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: NIPS. pp. 379–387 (2016)
6. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: Centernet: Keypoint triplets for object detection. In: ICCV. pp. 6569–6578 (2019)
7. Everingham, M., Gool, L.J.V., Williams, C.K.I., Winn, J.M., Zisserman, A.: The pascal visual object classes (VOC) challenge. IJCV **88**(2), 303–338 (2010)
8. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A.: Cascade object detection with deformable part models. In: CVPR. pp. 2241–2248 (2010)
9. Francisco, M., Ross, G.: maskrcnn-benchmark (2018), `https://github.com/facebookresearch/maskrcnn-benchmark`
10. Galar, M., Fernández, A., Tartas, E.B., Sola, H.B., Herrera, F.: A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. TSMC, Part C **42**(4), 463–484 (2012)
11. Girshick, R., Radosavovic, I., Gkioxari, G., Dollár, P., He, K.: Detectron (2018), `https://github.com/facebookresearch/detectron`
12. Girshick, R.B.: Fast R-CNN. In: ICCV. pp. 1440–1448 (2015)
13. Guo, M., Haque, A., Huang, D., Yeung, S., Fei-Fei, L.: Dynamic task prioritization for multitask learning. In: ECCV. pp. 282–299 (2018)
14. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: ICCV. pp. 2980–2988 (2017)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
16. Huang, L., Yang, Y., Deng, Y., Yu, Y.: Densebox: Unifying landmark localization with end to end object detection. arXiv:1509.04874 (2015)
17. Kemal, Cam, B.C., Kalkan, S., Akbas, E.: Imbalance problems in object detection: A review. arXiv:1909.00169 (2019)
18. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: CVPR. pp. 7482–7491 (2018)
19. Kong, T., Sun, F., Liu, H., Jiang, Y., Shi, J.: Foveabox: Beyond anchor-based object detector. arXiv:1904.03797 (2019)
20. Kong, T., Sun, F., Yao, A., Liu, H., Lu, M., Chen, Y.: RON: reverse connection with objectness prior networks for object detection. In: CVPR. pp. 5244–5252 (2017)
21. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
22. Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. In: ECCV. pp. 765–781 (2018)

23. LeCun, Y., Bengio, Y., Hinton, G.E.: Deep learning. Nature **521**(7553), 436–444 (2015)
24. Li, B., Liu, Y., Wang, X.: Gradient harmonized single-stage detector. In: AAAI. pp. 8577–8584 (2019)
25. Li, Y., Chen, Y., Wang, N., Zhang, Z.: Scale-aware trident networks for object detection. In: ICCV. pp. 6054–6063 (2019)
26. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. In: CVPR. pp. 936–944 (2017)
27. Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV. pp. 2999–3007 (2017)
28. Lin, T., Maire, M., Belongie, S.J., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: common objects in context. In: ECCV. pp. 740–755 (2014)
29. Liu, S., Johns, E., Davison, A.J.: End-to-end multi-task learning with attention. In: CVPR. pp. 1871–1880 (2019)
30. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., Berg, A.C.: SSD: single shot multibox detector. In: ECCV. pp. 21–37 (2016)
31. Liu, X., Wu, J., Zhou, Z.: Exploratory undersampling for class-imbalance learning. IEEE Trans. Systems, Man, and Cybernetics, Part B **39**(2), 539–550 (2009)
32. Lu, X., Li, B., Yue, Y., Li, Q., Yan, J.: Grid R-CNN. In: CVPR. pp. 7363–7372 (2019)
33. Maloof, M.: Learning when data sets are imbalanced and when costs are unequal and unknown. In: ICML 2003 Workshop on Learning from Imbalanced Data Sets II. pp. 73–80 (2003)
34. Pang, J., Chen, K., Shi, J., Feng, H., Ouyang, W., Lin, D.: Libra R-CNN: towards balanced learning for object detection. In: CVPR. pp. 821–830 (2019)
35. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
36. Redmon, J.: Darknet: Open source neural networks in c. http://pjreddie.com/darknet/ (2013–2016)
37. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR. pp. 779–788 (2016)
38. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: CVPR. pp. 6517–6525 (2017)
39. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv:1804.02767 (2018)
40. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. TPAMI **39**(6), 1137–1149 (2017)
41. Seiffert, C., Khoshgoftaar, T.M., Hulse, J.V., Napolitano, A.: Rusboost: A hybrid approach to alleviating class imbalance. TSMC, Part A **40**(1), 185–197 (2010)
42. Shrivastava, A., Gupta, A., Girshick, R.B.: Training region-based object detectors with online hard example mining. In: CVPR. pp. 761–769 (2016)
43. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: fully convolutional one-stage object detection. In: ICCV. pp. 9627–9636 (2019)
44. Uijlings, J.R.R., van de Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective search for object recognition. IJCV **104**(2), 154–171 (2013)
45. Viola, P.A., Jones, M.J.: Robust real-time face detection. IJCV **57**(2), 137–154 (2004)
46. Wang, J., Chen, K., Yang, S., Loy, C.C., Lin, D.: Region proposal by guided anchoring. In: CVPR. pp. 2965–2974 (2019)

47. Weiss, G.M.: Mining with rarity: a unifying framework. SIGKDD Explorations **6**(1), 7–19 (2004)
48. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. `https://github.com/facebookresearch/detectron2` (2019)
49. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: Reppoints: Point set representation for object detection. In: ICCV. pp. 9657–9666 (2019)
50. Zhang, S., Wen, L., Bian, X., Lei, Z., Li, S.Z.: Single-shot refinement neural network for object detection. In: CVPR. pp. 4203–4212 (2018)
51. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv:1904.07850 (2019)
52. Zhou, X., Zhuo, J., Krähenbühl, P.: Bottom-up object detection by grouping extreme and center points. In: CVPR. pp. 850–859 (2019)
53. Zhu, C., He, Y., Savvides, M.: Feature selective anchor-free module for single-shot object detection. In: CVPR. pp. 840–849 (2019)
54. Zitnick, C.L., Dollár, P.: Edge boxes: Locating object proposals from edges. In: ECCV. pp. 391–405 (2014)