DW 双加权方案复现工作

陈杰聪

摘要

目标检测中的标签分配任务用于为训练样本分配不同的权重，但是现有的标签分配方法大多侧重于对正权重函数的设计，而负权重则直接由正权重得到，这种机制限制了检测器的学习能力。因此，我们研究了一种新的权重设计机制，称为双重加权 (DW)。这种加权方法分别使用不同的策略计算样本的正负权重。具体地，因为目标检测任务分为分类和回归两阶段，样本的正权重需要由其分类阶段和回归阶段之间的一致性程度来确定，而负权重则被分解为两个项：它是负样本的概率和它作为负样本对损失计算产生影响的大小（重要性）。这种加权策略提供了更好的灵活性来区分重要和次要样本，使检测器的检测结果更有效。经过验证，本方法在

关键词：目标检测；标签分配；卷积神经网络

1 引言

在计算机视觉的众多任务中，目标检测占有十分重要的地位，几十年来一直受到研究人员的广泛关注，见证了卷积神经网络 (CNN)[1]和 transformer(VITS)[2]的快速发展。当前最先进的检测器[3]大多通过用一组预定义的锚来预测类别标签和回归偏移量以执行密集检测。作为探测器训练的基本单元，锚框需要被分配适当的分类 (CLS) 和回归 (REG) 标签，以监督训练过程。这样的标签分配 (LA) 过程可以被视为向每个锚框分配损失权重的任务。锚的 CLS 损失通常可以表示为 (REG 损失也具有类似的定义)：

$$L_{cls} = w_{pos} * ln(s) - w_{neg} * ln(1 - s) \tag{1}$$

硬 LA 假设每个锚点不是 Pos 就是 Neg，这意味着 Wpos，Wneg ∈ 0,1 且 Wneg+Wpos=1。该策略的核心思想是找到一个合适的划分边界，将锚点分为正集和负集。沿着这条研究路线的划分规则可以进一步分为静态规则和动态规则。静态规则[4]采用诸如 IOU 或从锚中心到 GT 中心的距离等预定义的度量将各个锚框分配为对象或背景，这种静态分配规则忽略了具有不同大小和形状的对象的划分边界有可能不同的事实。对于动态分配规则[5]来说，ATSS[6]基于对象的 IOU 分布来为锚框分配标签，预测感知分配策略[7]则将预测的置信度分数作为评估锚重要性的指标，以另外分配具有权重的标签。但是，静态分配方法和动态分配方法都忽略了样本并不同等重要的事实。通过对目标检测中的评价指标进行分析，最优预测不仅要有较高的分类分数，而且要有准确的定位，这意味着在训练过程中，CLS 头部和 REG 头部之间一致性较高的锚框应该具有更高的重要性。

软 LA 包括 GFL[8]和 VFL[9]两种典型的方法，它们基于 IOU 选择需要进行软标签分配的对象，然后通过乘以调制因子将其转换为损失权重。其他一些工作[10]通过联合考虑分类得分和回归分数来计算样本权重。现有的方法主要集中在正权重函数的设计上，而负权重简单地由正权值得到，因此负权重能提供的新信息很少，这会限制检测器的学习能力。我们认为，这种加权机制不能在更精细的水平上区分每个训练样本。

为此，我们的方法使用不同的策略计算正负权重，这种双加权方案（DW）在考虑了不同因素的条件下，对正负权重进行设计，使他们具有互补的关系，可以克服原有标签分配方法对于负权重设计

上缺陷。具体来说，正权重反映了锚框分类得分和回归得分的一致性程度，一致性高的样本会被视为更重要的样本，其对应的正权重会更大。负权重则反映了不一致的程度，越不一致的样本负权重越大。这两部分的设计都会导致在计算损失后，探测器将沿着更准确的方向更新。

2 相关工作

学习标签分配前，我们要先了解目标检测中常用损失数的形式，这里以 Focal loss 损失为例。之后再详细介绍现有的标签分配的不同方案，包括硬标签分配和软标签分配。

2.1 Focal loss

Focal Loss 首次在目标检测框架 RetinaNet 中提出，是对典型的交叉信息熵损失函数的改进，主要用于样本分类的不平衡问题，既考虑了正负样本数量的不平衡特性，也考虑了难易样本的不同重要性。我们先从二分类交叉熵损失入手：

$$CE(p,y)=\begin{cases}-log(p) & \text{if y=1}\\ -log(1-p) & \text{otherwise}\end{cases} \tag{2}$$

这里 p 代表模型预测属于前景的概率，y 取值为 1 和-1，代表前景和背景。令 Pt 定义如下:

$$p_t=\begin{cases}p & \text{if y=1}\\ 1-p & \text{otherwise}\end{cases} \tag{3}$$

则有

$$CE(p,y)=CE(p_t)=-log(p_t) \tag{4}$$

这时，引入处理正负样本数量不平衡的权重因子 α∈ [0,1]，若为正样本，权重因子就是 α，若为负样本，权重因子则为 1-α。所以，损失函数也可以改写为：

$$CE(p_t)=-\alpha_t log(p_t) \tag{5}$$

最后，再引入用于处理样本难易程度不同的调制因子 γ，使损失关注难分样本，更新公式如下：

$$FL(p_t)=-\alpha_t(1-p_t)^{\gamma}log(p_t) \tag{6}$$

当 Pt 趋向于 1，即说明该样本是易区分样本，此时调制因子 $(1-p_t)^{\gamma}$ 趋向于 0，说明对损失的贡献较小，即减低了易区分样本的损失比例。当 Pt 很小，也就是假如某个样本被分到正样本，但是该样本为前景的概率特别小，即被错分到正样本了，此时调制因子 $(1-p_t)^{\gamma}$ 趋向于 1，对 loss 也没有太大的影响。

2.2 硬标签分配

硬标签分配。将每个锚标记为 Pos 或 Neg 样本是检测器训练的关键步骤。传统的基于锚的对象检测器[11]通过用 GT 对象测量锚的 IOU 来设置锚的标签。近年来，无锚检测器因其简洁的设计和可比的性能而引起了人们的广泛关注。FCOS[12]和 Foveabox[4]都是通过中心抽样策略来选择 Pos 样本：靠近 GT 中心的锚被抽样为阳性，而其他锚则是阴性的或属于会在训练期间被忽略类别。上述 LA 方法对不同形状和大小的 GT 框采用固定的规则，但这其实不是最好的方法。为此又有一些先进的 LA 策略[5]为每个 GT 动态选择 Pos 样本。ATSS[6]从特征金字塔的每一层中选择 TOP-K 锚点，并采用这些顶

端锚点的平均值作为 Pos/Neg 划分阈值。PAA[7]基于 CLS 和 REG 损失的联合状态以概率的方式自适应地将锚分离为 Pos/Neg 锚。OTA[5]通过将分配过程描述为最优运输问题，从全局的角度处理 LA 问题。基于 Transformer 的检测器[4]采用一对一分配方案，为每个 GT 寻找最佳位置样本。Hard LA 对所有样本一视同仁，但这与目标检测中的评估指标不太匹配，没有考虑样本分类和回归过程的一致性问题。

2.3 软标签分配

软标签分配。由于预测的框在评价时的重要性程度不同（分类和回归得分越一致的样本越重要），因此在训练时对样本应有区别对待。很多工作[13]被提出来解决这种样本不平等的问题。Focal loss[14]在交叉熵损失上添加了一个调制因子，以降低分配给分类良好的样本的损失的权重，这推动检测器聚焦于难分样本。Generalized focal loss[8]综合考虑分类得分和定位得分，为每个锚点分配软权重。Varifocal loss[9]利用感知 Iou 的分类标签来训练分类头。上面提到的大多数方法都专注于计算正权重，并简单地将负权重定义为 1 - Wpos 的函数。在本文中，我们将这一过程解耦，并分别为每个锚点分配 Pos 和 Neg 损失权重。大多数软 LA 方法都是对损失分配权重。但有一种特殊情况是为评分结果分配权重，它可以表述为 L_{cls} =−ln (Wpos × s)−ln(1−Wneg × s)，典型的方法包括 FreeAnchor[15]和 Autoassign[16]，这种方法获取的得分权重需要计算梯度以进行更新，但我们的方法与他们不同，我们在计算损失权重时脱离了网络，可以简单看作是对损失进行系数的加权。

3 本文方法

3.1 本文方法概述

为了与 NMS 兼容，一个好的检测器应该能够预测具有高分类分数和精确定位位置的一致边界框。如果忽视这两者的一致性性质，同等对待所有的训练样本，那么分类和定位两过程的结果就会出现错位: 样本类别得分最高的位置通常不是回归边界的最精确位置。这种不一致会降低探测器的性能，特别是在 IoU 较高的情况下。Soft LA 通过加权损失以一种软的方式处理训练样本，试图增强 CLS 和 REG 头之间的一致性，我们的方法便是借鉴这一策略，对损失进行软加权，并专门设计不同的正负权重函数，在考虑分类和回归一致性的前提下，对负权重进行单独的设计，这可以为探测器提供新的输入信息，提高探测器的准确性。

具体的，我们以预测感知的方式分别设置 Pos 和 Neg 权重。正权重函数以预测的分类评分 s 和预测框与 GT 对象之间的 IoU（代表回归精确度）为输入，通过估计 CLS 与 REG 头部之间的一致性程度来设置正权重。负权函数的输入与正权函数相同，但将负权表示为两个项的乘法，分别为：样本为负的概率以及样本为负时，其对损失带来的影响的重要性程度。通过这种方法，具有相似正权重的模糊样本可以根据负权重接收到更细粒度的监督信号，这是现有方法所不能做到的。

DW 框架的流程如图 1 所示。作为一种常见的做法[10]，我们首先通过选择靠近 GT 中心 (中心先验) 的锚点，为每个 GT 对象构建一批候选阳性。候选包外的锚被认为是负样本，由于它们的统计数据 (如 IoU、分类分数) 在训练早期非常嘈杂，因此不参与权重函数的设计过程。候选包内的锚将被分配到 Wpos、Wneg 和 Wreg 三个重量，以更有效地监督训练过程。

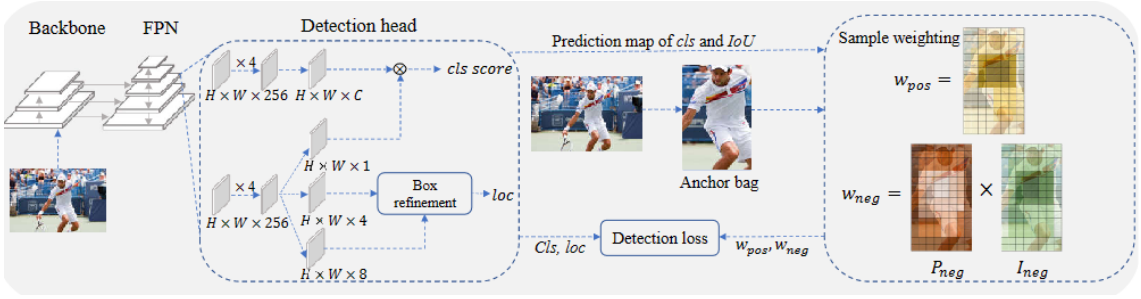

图 1: 流程图

3.2 正权重设计

样本的 Pos 权重应反映该样本在本次目标检测过程中的重要性。通过对目标检测评价指标的分析，找出目标检测过程中，影响样本重要性的因素。在 COCO 的测试中，对一个类别的所有预测都应该根据一个排名指标进行适当的排名。现有方法通常使用分类评分或结合了分类评分与回归得分的指标进行排名。总的来说，被判为 Pos 样本的影响因素包括分类得分 S 以及回归精确度 IoU，拥有越高的 s 或越高的 IoU，都越有可能被判为正样本。由此可见，同时满足高分类得分 S 以及高回归得分 IoU 的锚点更有可能在测试中被定义为 Pos 预测，因此在训练中它们应该具有更高的重要性。从这个角度来看，我们的 Pos 权重 Wpos 将被设计得与 IoU 和分类得分 S 正相关，即 Wpos $\propto$ IoU 和 Wpos $\propto$ S。为了指定 Pos 函数，我们首先定义一个一致性度量，记为 t，来衡量两种条件之间的对一致性程度:

$$t = s * IoU^{\beta} \tag{7}$$

其中 β 用于平衡两个条件。为了鼓励不同锚点之间的 Pos 权重能够具有较大差异，我们添加了一个指数调制因子:

$$w_{pos} = e^{\mu t} * t \tag{8}$$

其中 μ 是一个超参数，用于控制不同 Pos 权值的相对间隙。最后，每个实例的每个锚的 Pos 权值由候选包中所有 Pos 权值的和归一化。

3.3 负权重设计

虽然 Pos 权值可以强制分类和回归一致性高的锚点具有相应的高重要性，但不一致锚点的重要性不能仅仅通过正权重值来区分。例如当锚点 1 具有高分类得分以及低回归得分，而锚点 2 具有低分类得分以及高回归得分时，他们的关于分类和回归的一致性程度可以认为是相同的，故他们会以相同的 Pos 强度排列在类似位置，且反映不出它们的差异。为了给检测器提供更有鉴别性的监督信息，我们建议通过给它们分配更多不同的负权重来表明它们的不同重要性，并将负权重定义为两项因子之积，分别为：样本成为负样本的概率以及其作为负样本时将对损失带来的影响。

样本作为负样本的概率：根据 COCO 的评价指标，IoU 小于 θ 是错误预测的充分条件。这意味着不满足 IoU 度量的预测边界框将被视为阴性检测，即使它有很高的分类评分。也就是说，IoU 是决定成为负样本概率的唯一因素，用 P_{neg} 表示。由于 COCO 在估算 AP 时采用了介于 0.5~0.95 之间的 IoU 区间，因此边界盒的概率 P_{neg} 应满足以下规则:

$$P_{neg} = \begin{cases} 1, & if IoU < 0.5 \\ [0,1], & if IoU \in [0.5, 0.95] \\ 0, & if IoU > 0.95 \end{cases} \tag{9}$$

在区间 [0.5,0.95] 内定义的任何单调递减函数都可以被用于 P_{neg} 的计算。为了简单起见，我们将 P_{neg} 实例化为如下函数:

$$P_{neg} = -k * IoU^{\gamma 1} + b, \qquad if \quad IoU \in [0.5, 0.95] \tag{10}$$

它通过点 (0.5, 1) 和 (0.95,0)。一旦确定了 γ1，参数 k 和 b 可以通过未确定系数的方法获得。图 3 绘制了具有不同的 γ1 的 P_{neg} 与 IoU 曲线。

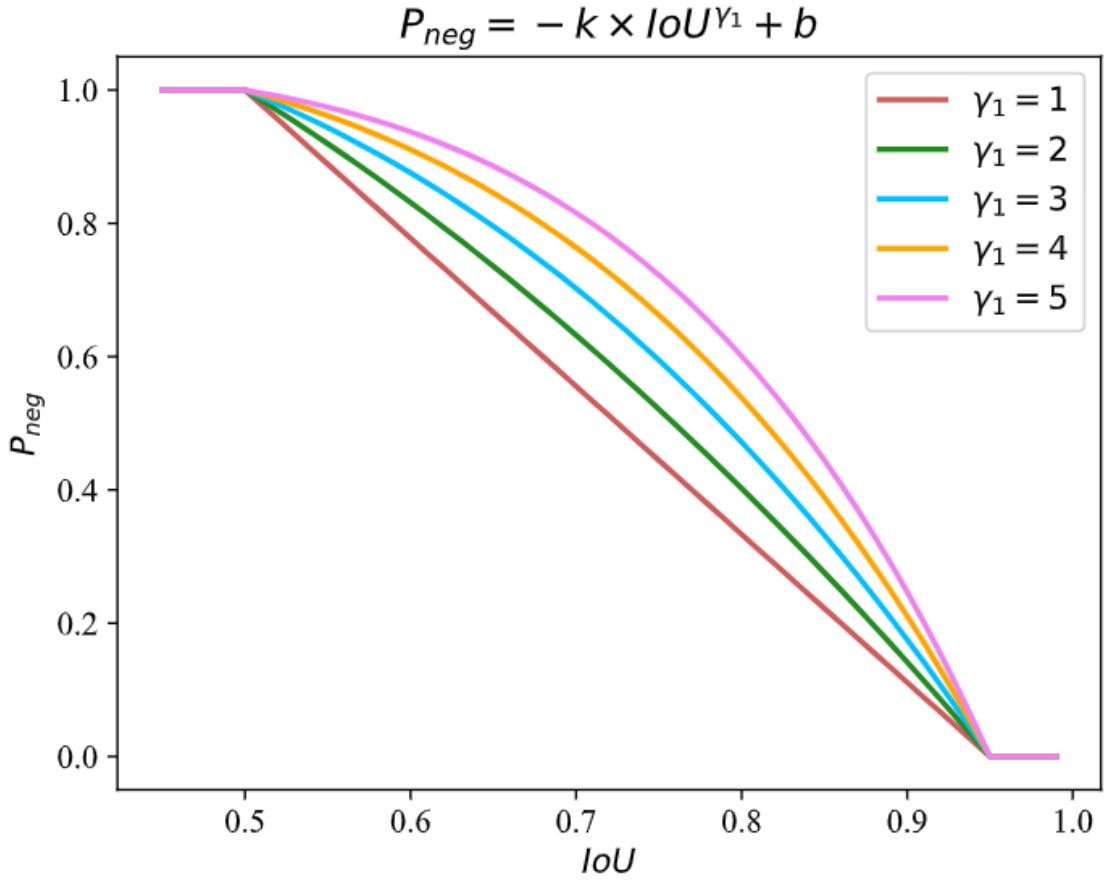

图 2: 不同 γ 的递减概率函数

作为负样本时，样本对损失带来的影响：在推理时，被判定为负样本且被选中的样本不会影响召回率，但会减低准确率。为了缓解这个过程，负样本的得分需要尽可能低，以尽量不要被选入候选包。基于这一点，我们可以直到，分类得分大的否定预测比分类得分小的否定预测更重要，因为它们是难判断样本。因此，用 I_{neg} 表示的阴性样本的重要性应该是排名得分的函数。为简单起见，我们将其设为：

$$I_{neg} = s^{\gamma 2} \tag{11}$$

其中，γ2 是一个因子，表示应该给予具有较大重要性的负样本多大的优先级。

最后得到负权重 Wneg = $P_{neg} \times I_{neg}$，即有：

$$w_{neg} = \begin{cases} s^{\gamma 2}, & if IoU < 0.5 \\ -k * IoU^{\gamma 1} + b * s^{\gamma 2}, & if IoU \in [0.5, 0.95] \\ 0, & if IoU > 0.95 \end{cases} \tag{12}$$

负权重与 IoU 呈负相关，与 S 呈正相关。可以看出，对于两个 Pos 权值相同的锚框，IoU 越小的样本负权值越大。Wneg 的定义符合推理过程，可以进一步区分具有几乎相同 Pos 权值的模糊锚点。

4 复现细节

4.1 复现工作

论文基于经典检测方法 Fcos 进行，故本次实验我以 Fcos 为 baseline 进行实验对比验证。通过 mmdetection 环境的搭建，向其注册了 DW 的结构，使 Fcos 和 DW 的实验环境相同，确保实验结果没有问题，并且可以更方便的进行实验。

由于论文使用的数据集为 COCO 数据集，数据量较大。一开始与师兄讨论后决定使用实验室的烟雾数据集，将其转为 COCO 格式后进行实验，但最后因为该数据集也处于未完成阶段，在转换过程发现有些图片没有标签，导致转换出现错误，因此还是用回 COCO 数据集。之后由于显卡问题，跑不动完整的 COCO，故在和老师讨论之后决定使用 1/10 的 COCO 数据集进行训练。但跑出来的结果差距实在是太大，fos 中的 mAP 只有 26.7%，DW 的 mAP 也只有 32%。

```
2022-12-01 02:58:03,846 - mmdet - INFO - Exp name: fcos_r50_caffe_fpn_gn-head_1x_coco.py
2022-12-01 02:58:03,846 - mmdet - INFO - Epoch(val) [12][500]    bbox_mAP: 0.2670, bbox_mAP_50: 0.4410, bbox_mAP_75: 0.2770,
bbox_mAP_s: 0.1460, bbox_mAP_m: 0.3270, bbox_mAP_l: 0.3650, bbox_mAP_copypaste: 0.267 0.441 0.277 0.146 0.327 0.365
```

图 3: fcos_miniCOCO

```
2022-11-30 16:46:41,833 - mmdet - INFO - Exp name: dw_r50_fpn_1x_coco.py
2022-11-30 16:46:41,834 - mmdet - INFO - Epoch(val) [12][500]    bbox_mAP: 0.3200, bbox_mAP_50: 0.4890, bbox_mAP_75: 0.3370,
bbox_mAP_s: 0.1680, bbox_mAP_m: 0.3720, bbox_mAP_l: 0.4430, bbox_mAP_copypaste: 0.320 0.489 0.337 0.168 0.372 0.443
```

图 4: DW_miniCOCO

后续在实验室资源足够的时候，我们还是使用了完整的 COCO 数据集，对论文的实验结果进行了验证性的实验。

4.2 与已有开源代码对比

由于论文使用了不同的 backbone 进行实验分析，且不同的 backbone 会带来不同的性能影响，故我们使用了论文中没有提到的 SENet，分析其对性能产生的影响。

```
backbone=dict(
    type = 'SENet',
    block = 'SEBottleneck',
    layers=[3, 8, 36, 3],
    groups = 64,
    reduction = 16,
    frozen_stages=1,
    init_cfg = dict(
        type='Pretrained',
        checkpoint='work_dirs/pre_train/senet154-c7b49a05.pth')
),
```

图 5: 改用 senet 作为 backbone

在训练时，每隔一段时间，随机选择一种尺度进行训练，将能在一定程度上提高检测模型的鲁棒性，且使用小图片测试会更快，但准确率低，而使用尺度大的图片测试速度慢，准确率高，故在输入图片前，我们设置两个尺度代表大小图片，并验证性能是否有提高。

```python
train_pipeline = [
    dict(type='LoadImageFromFile'),
    dict(type='LoadAnnotations', with_bbox=True),
    dict(
        type='Resize',
        img_scale=[(1333, 480), (1333, 960)],
        multiscale_mode='range',
        keep_ratio=True),
    dict(type='RandomFlip', flip_ratio=0.5),
    dict(type='Normalize', **img_norm_cfg),
    dict(type='Pad', size_divisor=32),
    dict(type='DefaultFormatBundle'),
    dict(type='Collect', keys=['img', 'gt_bboxes', 'gt_labels']),
]
```

图 6: 使用多尺度代码

4.3 实验环境搭建

实验需要借助 mmdet 的环境，故需要搭建 MMdetection，可以使用 pip 安装 MMCV，需要在 MMCV 官方网址寻找对应的 cuda 版本以及 torch 版本。下载完成后使用 pip install mmdet 命令即可安装完成。

5 实验结果分析

初始 fcos 与 DW 分析：在没有改变 backbone、图像尺度、以及 box 细化的情况下，使用 resnet50 的 backbone 进行 fcos 以及 DW 的实验，得到 fcos 的 mAP 为 0.365，DW 实验结果为 0.409，由此已经可以验证 DW 相对于从前 fcos 的优越性，实验结果如下：

```
Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.365
Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=1000 ] = 0.558
Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=1000 ] = 0.388
Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = 0.210
Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.402
Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.465
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.535
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=300 ] = 0.535
Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=1000 ] = 0.535
Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=1000 ] = 0.344
Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=1000 ] = 0.580
Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=1000 ] = 0.686

2022-12-03 12:01:46,489 - mmdet - INFO - Exp name: fcos_r50_caffe_fpn_gn-head_1x_coco.py
2022-12-03 12:01:46,490 - mmdet - INFO - Epoch(val) [12][1667]   bbox_mAP: 0.3650, bbox_mA
bbox_mAP_m: 0.4020, bbox_mAP_l: 0.4650, bbox_mAP_copypaste: 0.365 0.558 0.388 0.210 0.402
```

图 7: fcos_r50

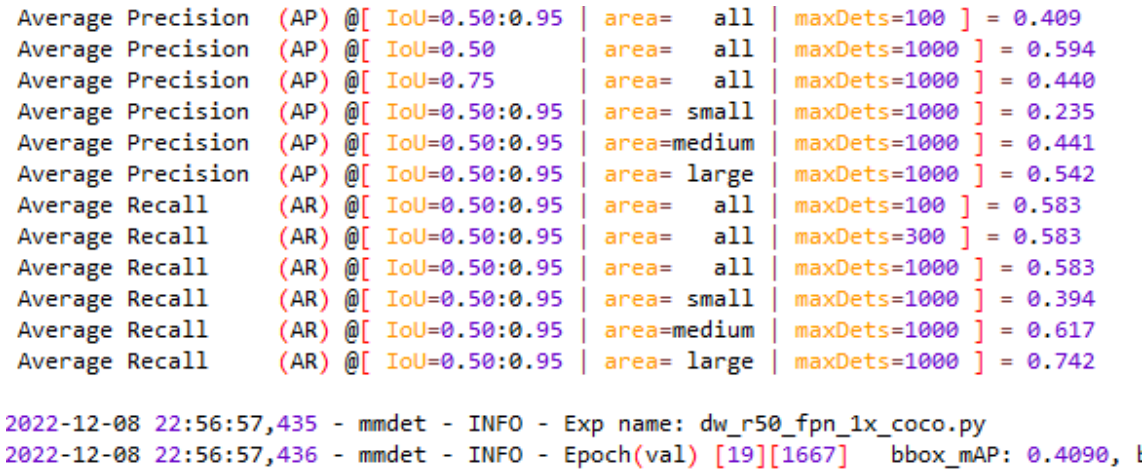

图 8: DW_r50

对 DW 开启多尺度训练：由于多尺度训练针对自身进行改进，我们只对比开启多尺度训练前后的结果，实验得出开启多尺度训练前，DW 的 mAP 为 0.409，开启多尺度训练后，DW 的 map 达到 0.418，实验结果如下：

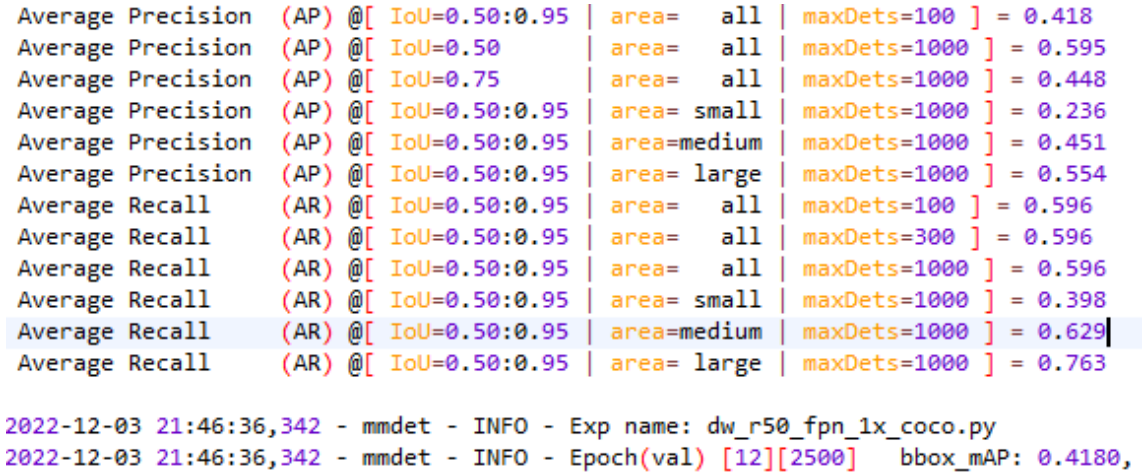

图 9: DW_r50_ 多尺度

DW 使用 senet 作为 backbone: 修改 backbone，使用 senet 作为 backbone，对比改动后的 DW 实验结果性能变化。实验得出，使用 senet 进行实验效果并不理想，用 senet 的 DW 实验 mAP 只有 0.372，相比原先的 0.409 性能还下降了，实验结果如下：

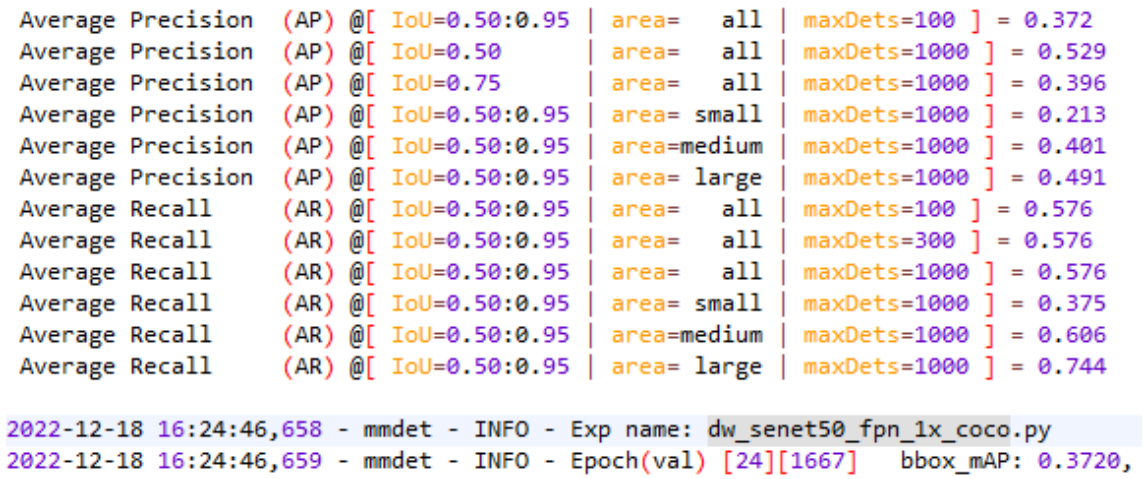

图 10: DW_senet50

统计上述实验所得的结果如下表所示：

表 1: 实验统计结果

Method	mAP	mAP50
fcos	0.365	0.558
DW	0.409	0.594
DW_multiscale	0.418	0.595
DW_senet	0.372	0.529

通过分析，我们可以知道，对正负权重使用不同函数进行设计可以提高传统目标检测的性能，在开启了多尺度训练之后，性能能够得到进一步的提升，但 backbone 并不能随意更改，在原有 resnet50 的情况下使用 resnet101 确实会有性能上的提升，但若只是简单的换成 senet 之类的其他 backbone，则最终的结果甚至比先前的 fcos 方法还差。

6 总结与展望

总结：本文从目标检测的实验流程着手，介绍了目标检测关于标签分配内容的背景以及一些相关的技术。对于目前现有方法存在的问题进行了一定的分析，得出分别设计正负权重函数的意义以及可取性。之后对正负权重的设计分别进行了较为全面的讲解，并将方程详细地列出。最后通过实验验证了这一改进的准确性，并在此基础上进行了一些其他改进。最后对于所有的实验结果进行了统计，并作出进一步分析，得出多尺度训练确实有利于性能的提升，而将 senet 作为 backbone 的方法会导致性能的下降。

展望：本次实验作为研究生入学后的第一个论文复现工作，确实存在较多不足，包括在实验前期数据集的选择，环境的搭建问题；实验中期代码的研读上遇到的问题；实验后期对比实验的设计问题。在整个论文复现过程中，总体上处于一种瞎担心，白忙活的状态，耗费精力的同时还无法顺利推进实验进度。希望未来的研究生活中能多提升这方面的能力。另外，论文中提出使用 box 优化方法，对回归问题进行优化后取得了较好的性能，那和回归问题同等重要的分类问题是不是也有进一步研究的空间。因为这两部分内容都对正负权重有较大的影响，若想继续研究也许可以朝着改进分类过程的方向进行。

参考文献

[1] HE K. Deep residual learning for image recognition[J]. In CVPR, 2016: 770-778.

[2] CARION N. End-to-end object detection with transformers[J]. In European Conference on Computer Vision, Springer, 2020: 213-229.

[3] BOCHKOVSKIY A. Yolov4: Optimal speed and accuracy of object detection[J]. arXiv preprint, 2020, arXiv:2004.10934.

[4] KONG T. Foveabox: Beyound anchor-based object detection[J]. IEEE Transactions on Image Processing, 2020, 29:7389−7398.

[5] GE Z. Ota: Optimal transport assignment for object detection[J]. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021: 303-312.

[6] ZHANG S. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection[J]. In Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition, 2020, pages 9759−9768.

[7] KIM K. Probabilistic anchor assignment with iou prediction for object detection[J]. In Computer Vision−ECCV 2020: 16th European Conference, Glasgow,UK, Springer, 2020, Part XXV 16, pages 355−371.

[8] LI X. Generalized focal loss:Learning qualified and distributed bounding boxes for dense object detection[J]. arXiv preprint, 2020, arXiv:2006.04388.

[9] ZHANG H. Varifocalnet: An iou-aware dense object detector[J]. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pages 8514−8523.

[10] FENG C. Tood: Task-aligned one-stage object detection[J]. In Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pages 3510−3519.

[11] LIU W. Ssd: Single shot multibox detector[J]. In ECCV, Springer, 2016, pages 21−37.

[12] TIAN Z. Fcos:Fully convolutional one-stage object detection[J]. In ICCV, 2019, pages 9627−9636.

[13] CAO Y. Prime sample attention in object detection[J]. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pages 11583−11591.

[14] LIN T Y. Focal loss for dense object detection[J]. In ICCV, 2017, pages 2980−2988.

[15] ZHANG X. Learning to match anchors for visual object detection[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021.

[16] ZHU B. Autoassign: Differentiable label assignment for dense object detection[J]. arXiv preprint, 2020, arXiv:2007.03496.