**场景人流识别**

|  |  |
| --- | --- |
| 组 别： | 第二小组 |
| 小组成员： | 1211002008 黄俊达 |
|  | 1211002001 白家运 |
|  | 1211002011 雷万年 |
|  | 1211006001 果 航 |

## 研究现状

人群计数一直是计算机视觉中热门的研究领域之一，主要是在图像或者人群中准确地统计人群数量。尤其在近几年，随着全球各地重大踩踏事故的频发，人群计数有着更为重要的意义。通过人群计数，可以通过管控降低人群密度，减小踩踏事件发生的概率。

近年来，随着深度学习技术的不断发展，有着越来越多的人群计数算法被不断提出并持续升级优化，有如基于卷积神经网络(CNN)的人群计数算法、多阵列卷积神经网络(Multi-scale CNN,即MSCNN)、Point-Query Quadtree for Crowd Counting(PET)、CSRNet、ic-CNN等。

CNN是一种深度学习的算法，其基本原理是通过多层的非线性变换来提取图像中的特征。与传统的计算机视觉方法相比，CNN能够自动学习和提取图像中的特征，而不需要手动设计和选择特征，此外，CNN还具有强大的鲁棒性和泛化能力，能够处理各种复杂场景下的图像。基于CNN 的人群计数算法通常可以分为以下几个步骤：数据预处理、模型构建和训练、预测和评估。但是，基于CNN的人群计数算法也存在一些缺点，如数据标注成本高、模型训练时间长、对大规模和密集的人群估计效果不佳。

PET通过点查询四叉树结构，允许数据相关的分裂，能够自适应地生成查询点以解决密集人群预测问题。PET还采用了渐进式矩形窗口注意力机制，提高了效率。尽管PET在某些方面有所改进，但人群计数的根本性挑战如遮挡处理、极端密集人群的计数准确性等依然存在。与此同时的是，该方法可能增加了模型的复杂性，需要更多的研究和实验来验证其广泛适用性。

MSCNN通过不同尺度的卷积核来提取不同大小的人群特征，增强了模型对不同密度人群的适应性。但与之对应的是，这种多阵列结构的模型通常参数较多，计算量大，可能无法实现实时的人群计数预测，有些低效的分支结构可能未能有效提取特征。

CSRNet模型抛弃了Multi-column框架，使用预训练的VGG16网络和空洞卷积(Dilated Convolution)，能够更有效地获取人头的边缘信息，从而提高计数精度。该模型虽然相比多阵列结构有所简化，但依然存在一定的计算复杂度，可能影响实时性能。

ic-CNN模型采取由低分辨率到高分辨率逐步细化的策略，得到高质量的密度图，从而提升计数的准确性。但是逐步细化的策略可能会增加计算步骤，影响整体速度。

## 问题定义

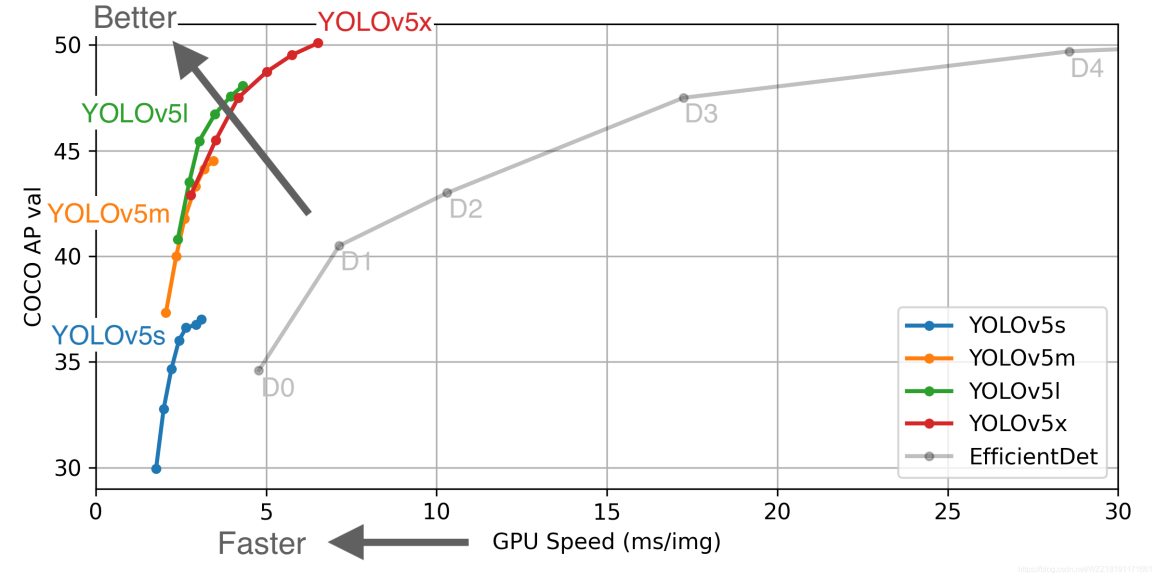
正如前文所提及的，随着深度学习技术的不断发展，人群计数算法在准确性、实时性和适应性都取得了显著进展，但让面临以下问题：

1. 数据标注成本高。尽管人群计数是近年来计算机视觉的热门研究领域之一，其数据集也在不断增多和提高精度，但是当在进行模型训练时，如若不能找到适配的数据集，仍需要根据特点的目标去进行专门的数据标注。这使得人群计算这一研究领域，数据标注成本仍是非常大的。
2. 计算资源需求大。深度学习是需要进行多次训练以达到更佳目的。加之近年提出的新型算法模型，其计算难度与量级无疑是巨大的，并且部分模型如PET还需大量实验和研究才能验证其广泛适用性。
3. 极端密集人群处理困难。这个问题是目前提出的人群计数模型都面临的问题，其仍是难以攻克的难关。在当前画面出现极端密集人群时，模型都会出现漏检、误检、准确率和实时率下降等各方面问题。
4. 遮挡重检准确率低。这同样是目前人群计数模型面临的问题，当人群被遮挡后重新出现时，其模型能否正确分配ID，正确识别同一个人，准确率能否提高，这些都是当前人群计数领域面临的难题。

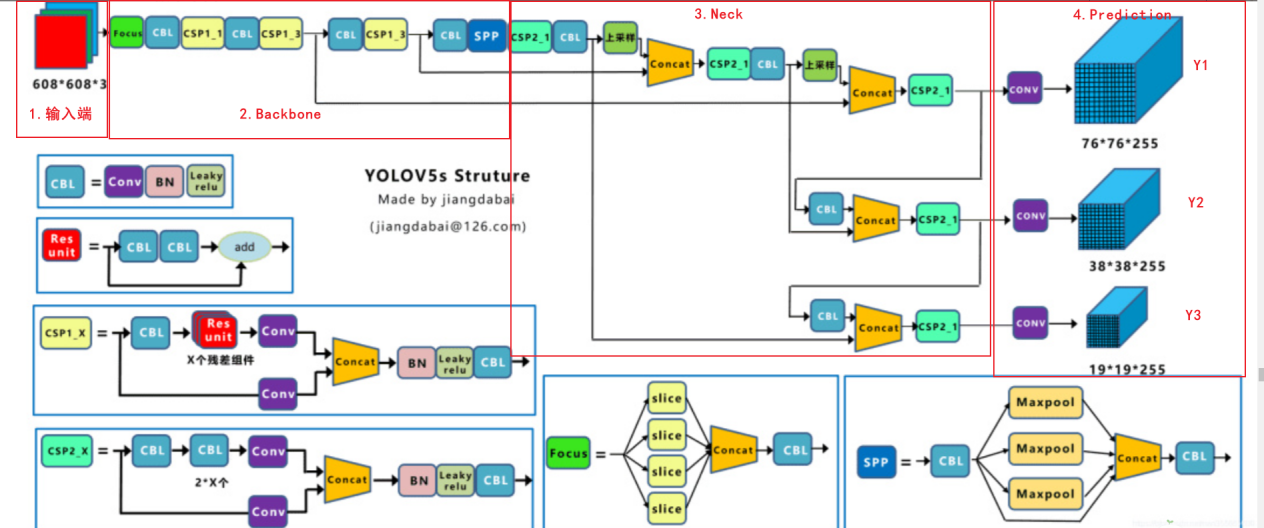
## Yolov5算法

Yolo(You Only Look Once)是一种one-stage目标检测算法，即仅需要 “看” 一次就可以识别出图片中物体的class类别和边界框。Yolov5是由Alexey Bochkovskiy等人在YOLO系列算法的基础上进行改进和优化而开发的，使其性能与精度都得到了极大的提升。

如下图所示是Yolov5算法在COCO数据集上的测试结果。其中，Yolov5s是深度最浅、特征图宽度最小的网络，速度最快，AP精度也最低。其它的三种网络，在Yolov5s基础上，不断加深加宽，AP精度也不断提升，但速度的消耗也在不断增加。



Yolov5的网络结构如下图所示，主要包括Input、Backbone骨干网络、Neck颈部结构和Head头部结构(Prediction)四个模块，分别负责输入图片预处理、特征提取、特征融合、输出检测信息。其中，Backbone模块使用BottleNeckCSP结构从输入图像中提取丰富的信息特征。

其中，

Backbone(BottleNeckCSP结构)：由Focus结构、三组CBL+CSP1\_x和CBL+SPP串行搭建。

CBL：由卷积Conv+批量归一化BN+激活函数Leaky Relu组成。

Focus：对图像进行切片后再Concat。

CSP1\_x：借鉴了CSPNet网络结构，由CBL模块、Res uint模块以及卷积层Concat组成，其中x表示有x个CSP1模块。

CSP2\_x：不再使用Res unit，由卷积层CBL模块Concat组成。

SPP：采用1×1，5×5，9×9，13×13的最大池化的方式，进行多尺度融合。

Res unit：残差组件，输入通过两个CBL后，再与原输入进行add。这是一种常规的残差单元，以使得网络可以提取到更深层的特征，同时避免出现梯度消失或爆炸。

上采样：利用元素复制扩充的方法使得特征图尺寸扩大，例如线性插值。

Concat：张量拼接，会扩充两个张量的维度，实现多尺度特征融合。例如26×26×256和26×26×512两个张量拼接，结果是26×26×768。

Add：张量相加，张量直接相加，不会扩充维度，例如104×104×128和104×104×128相加，结果还是104×104×128。

Y1、Y2、Y3：表示Yolov5三种尺度的输出。

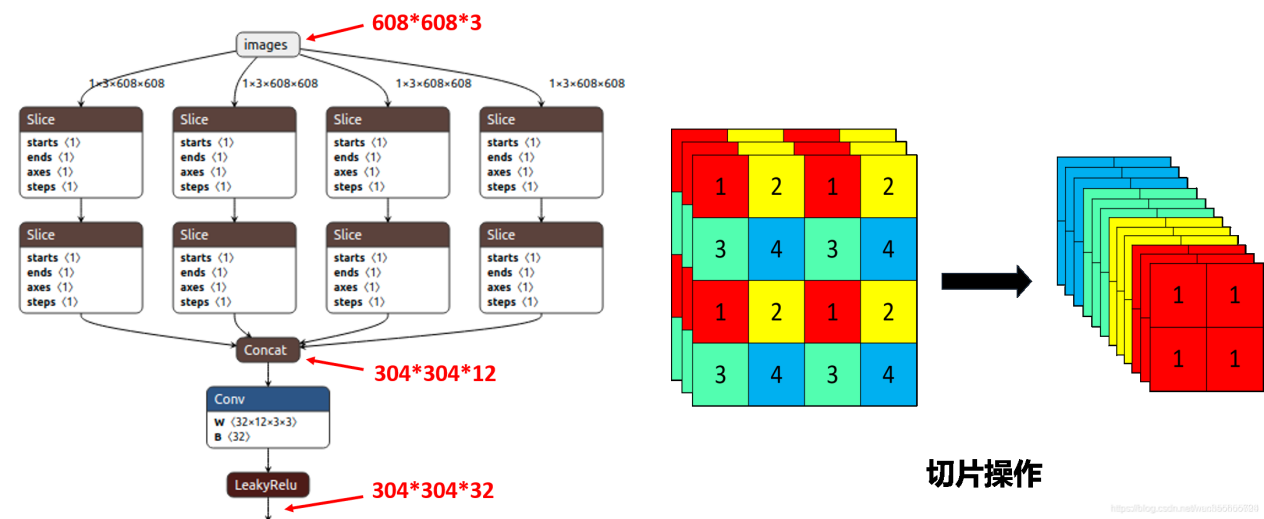
### 输入端

输入端主要对输入图片进行预处理。该网络输入图像大小为608\*608，预处理主要是将输入图像缩放至网络的输入大小，并进行归一化等操作。在网络训练阶段，Yolov5与Yolov4同样使用了Mosaic数据增强操作以提升模型的训练速度和网络精度；并提出了一种自适应锚框计算与自适应图片缩放方法。

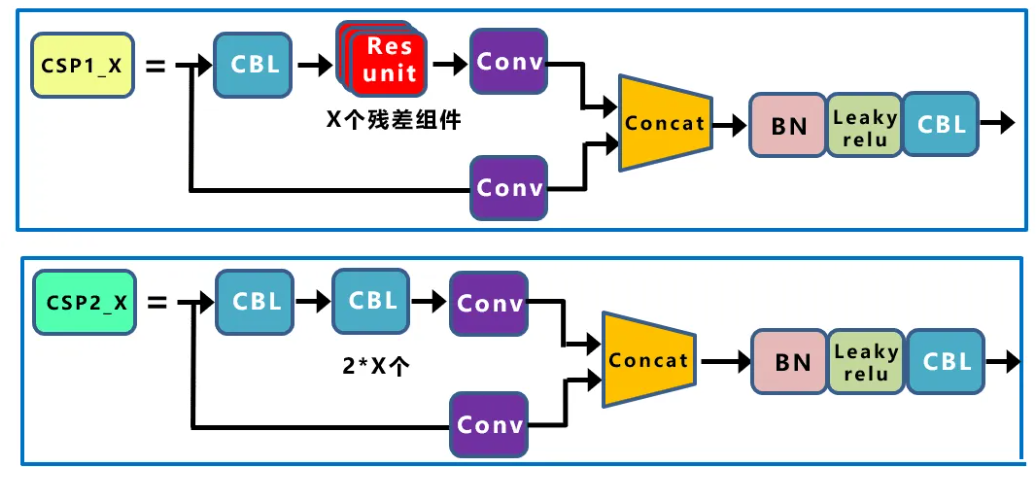
### Backbone骨干网络

Yolov5与Yolov4相比，添加了Focus结构。其中最关键的是切片操作。例如下图所示，在每张图片中每隔一个像素采集一个值，这样可组合为4张互补且无信息丢失的图片，同时将宽高信息集中到通道空间，输入通道扩充4倍，拼接后的图片相对于原图片的RGB三通道模式变为12通道，即4x4x3的图像经切片后变为了2x2x12的特征图。

而在Yolov5s网络结构中，608x608x3的图像输入经Focus结构，进行切片操作，先变为304x304x12的特征图，再经过一次32个卷积核的卷积操作，最终变为304x304x32的特征图。



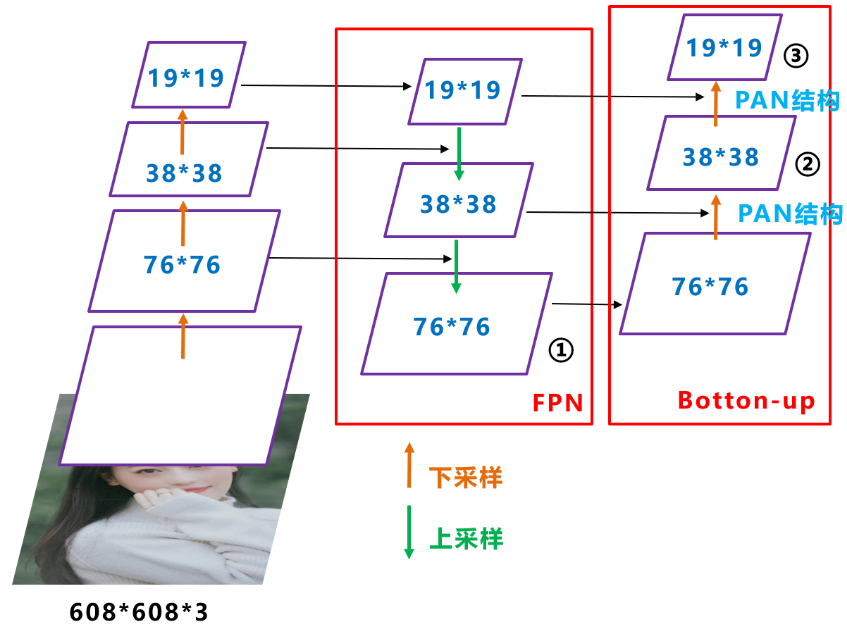
早在Yolov4网络结构中，便借鉴了CSPNet的设计思路，在主干网络中使用了CSP结构。Yolov5则是设计了两种CSP结构，其中CSP1\_x结构应用于Backbone主干网络，另一种CSP2\_x结构应用于Neck中。



**图 13 CSP结构**

### Neck颈部结构

Yolov5更新后的Neck与Yolov4同采用了FPN+PAN结构，如下图所示。



整个网络的输入为608x608，然后经过CSP模块，生成一个76x76大小的特征映射，再经过两次下采样操作之后生成19x19的特征映射。

接着传入FPN结构，依次对19x19、38x38、76x76执行融合操作，即先对比较小的特征映射层执行上采样操作，将其调整成相同大小，然后将两个同等大小的特征映射叠加起来。通过FPN操作可以将19x19大小的特征映射调整为76x76大小，这样不仅提升了特征映射的大小，可以更好的解决检测中尺度问题，而且增加了网络的深度，提升了网络的鲁棒性。

最后将其传入PAN结构，PANet网络的PAN结构是将两个相同大小的特征映射执行按位加操作，YOLOv5与Yolov4中同样使用Concat操作来代替它。经过两个PAN结构，可将76x76大小的特征映射重新调整为19x19大小，这样可以在一定程度上提升该算法的目标定位能力。

FPN层自顶向下可以捕获强语义特征，而PAN则通过自底向上传达强定位特征，通过组合这两个模块，可以很好的完成目标定位的功能。

### Prediction

Yolov5延用了Yolov3和Yolov4的多尺度融合检测方法，与Yolov4采用CIOU\_Loss作为Bounding box损失函数不同的是，Yolov5中采用GIOU\_Loss作为Bounding box的损失函数。

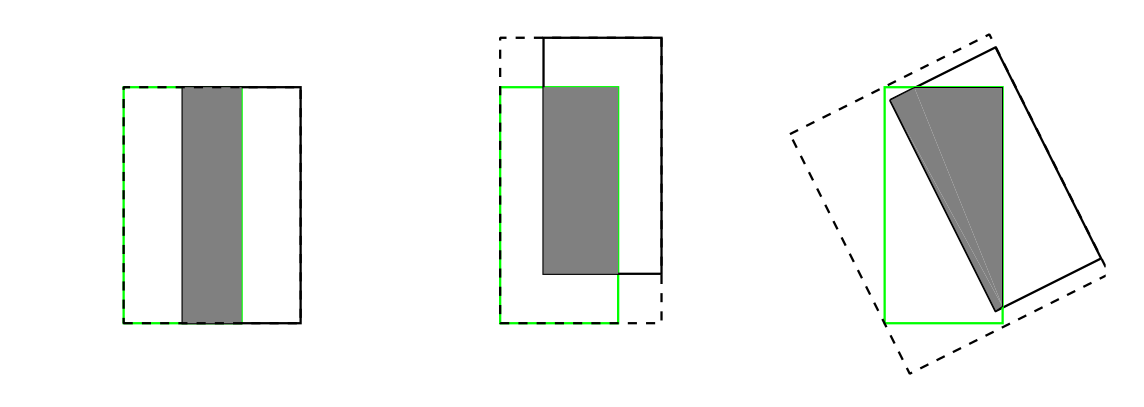
IoU即为交并比，用于度量目标检测中预测框与真实框的重叠程度。计算公式如下：

显而易见，IoU的值越高也说明预测框与真实框重合程度越高，模型预测越准确。反之，IoU越低模型性能越差。

IoU Loss计算公式如下：

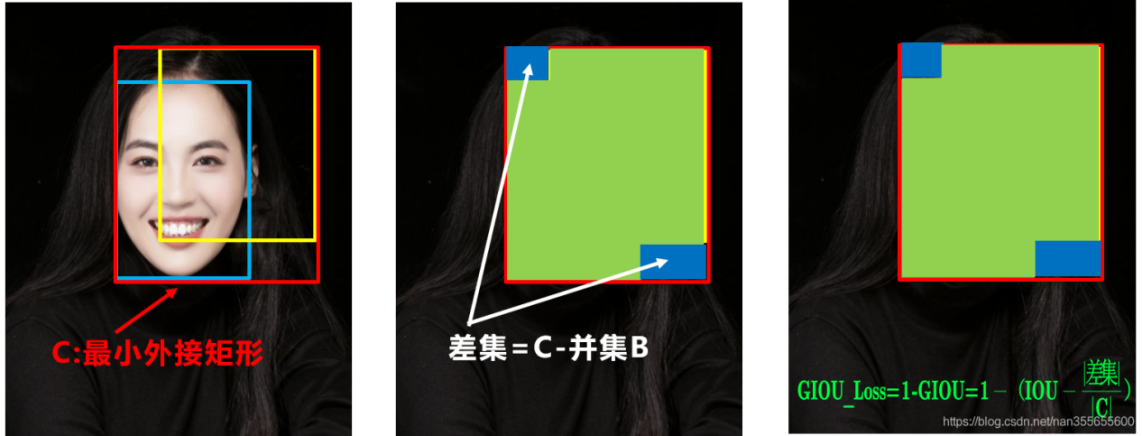
由下图可以看出，如果两个目标没有重叠，IoU将会为0，并且不会反应两个目标之间的距离，在这种无重叠目标的情况下，如果IoU用作于损失函数，梯度为0，无法优化。

并且，IoU无法区分两个对象之间不同的对齐方式。更通俗的理解就是，不同方向上有相同交叉级别的两个重叠对象的IoU会完全相等。

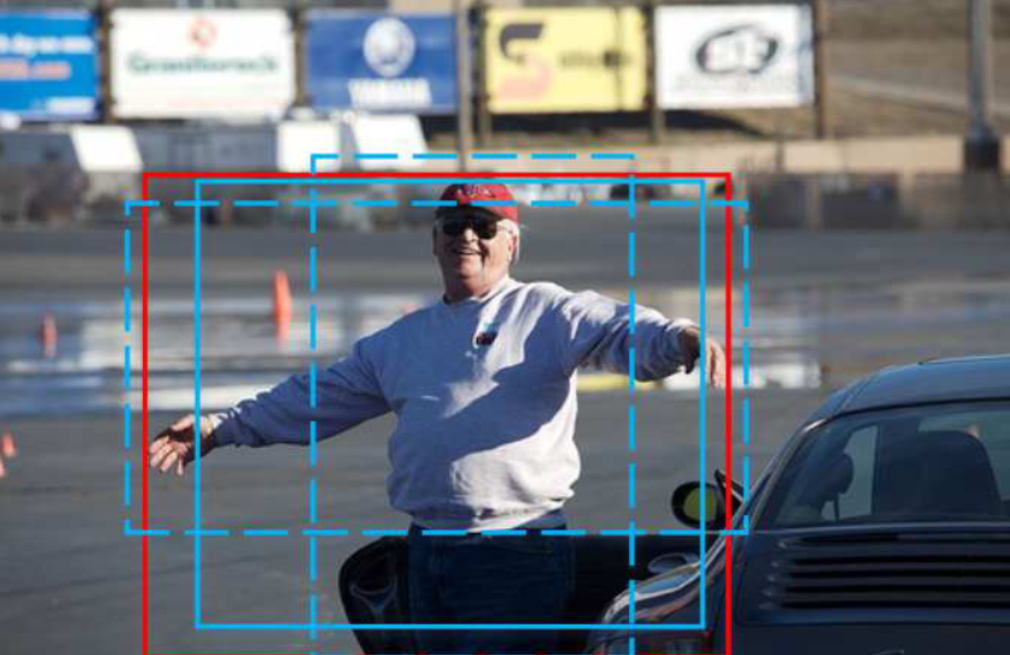


GIoU加入了C检测框（C检测框是包含了检测框和真实框的最小矩形框），这样就可以解决检测框和真实框没有重叠的问题。但是当检测框和真实框之间出现包含的现象的时候GIoU就和IoU loss是同样的效果了。GIoU Loss计算如下：

其中，C为能包含预测框和真实框的最小box。



在目标检测的后处理过程中，针对很多目标框的筛选，通常需要进行NMS操作。Yolov4在DIOU\_Loss的基础上采用DIOU\_nms的方式，而Yolov5中仍然采用加权NMS的方式。



加权NMS通过分类置信度和IOU对同类物体所有的边框坐标进行加权平均，并归一化。其中加权对象包括目标物自身和IOU>阈值的相邻框。

假定所有的Box来自相同的物体，通过考虑非极大结果充分考虑了目标的信息，提出了如下的非极大权重（Non-Maximum Weighting）：

## Yolov5配置问题及解决

1. Python版本、CUDA版本或其他依赖库版本不兼容。

确保安装了正确版本的Python（通常是Python 3.7或更高版本）。

安装与CUDA版本兼容的PyTorch版本。

使用pip安装所有必要的依赖库，例如：

1. 数据集格式不正确或路径设置错误。

（1）确保数据集按照YOLOv5的要求格式化，通常是JSON或YOLO格式。

（2）检查数据集路径是否正确设置在data/hyp.scratch.py文件中。

3、训练时出现内存不足或显存溢出。

（1）减少批量大小（batch size）或图像大小（img\_size）。

（2）使用torch.cuda.empty\_cache()释放未使用的缓存显存。

（3）确保使用的是最新版本的CUDA。

4、模型性能不佳，如准确率低和召回率低。

（1）检查数据集是否平衡，避免类别不平衡问题。

（2）调整模型超参数，如学习率、训练周期等。

（3）尝试使用不同的模型架构或预训练权重。

5、训练速度慢。

（1）使用预训练权重开始训练，以减少训练时间。

（2）使用分布式训练或多GPU训练来加速。