

# 基于三维立方体损失的点云目标检测算法

葛旭阳<sup>1</sup>, 黄尚锋<sup>2</sup>, 蔡国榕<sup>2</sup>

(1. 集美大学理学院, 福建 厦门 361021; 2. 集美大学计算机工程学院, 福建 厦门 361021)

[摘要] 常用的优化回归边界框参数的损失并不等价于最大化 IoU 指标, 并且 IoU 作为回归损失, 在边界框不重叠的情况下进行优化是不可行的。为了解决这个问题, 在 IoU 基础上加入外接框的计算, 将所得结果(3D\_CGIoU)作为损失纳入到目前主流的三维目标检测框架中, 并在数据集 KITTI 和 ScanNet 上进行实验。实验结果显示检测的平均精度得到了提升, 表明该方法是有用的。

[关键词] 目标检测; IoU; 3D\_CGIoU; 度量; 损失

[中图分类号] TN 911.73

## Point Cloudobject Detection Algorithm Based on 3D Cube Loss

GE Xuyang<sup>1</sup>, HUANG Shangfeng<sup>2</sup>, CAI Guorong<sup>2</sup>

(1. School of Science, Jimei University, Xiamen 361021, China;

2. College of Computer Engineering, Jimei University, Xiamen 361021, China)

**Abstract:** Optimizing the commonly used losses for regressing the parameters of a bounding box is not equivalent to maximizing the IoU index. And IoU as a regression loss, it is not feasible to optimize when boundary boxes do not overlap. In order to solve the above problems, the calculation of the circumscribed box is added to the basis of IoU(3D\_CGIoU) as a new metric and new loss. Incorporating 3D\_CGIoU as a loss into the state-of-the-art 3D object detection framework, and experiments were conducted on the KITTI dataset and the ScanNet dataset respectively. The improvement of the average accuracy (AP) of the detection shows the effectiveness of the method.

**Keywords:** object detection; IoU; 3D\_CGIoU; metric; loss

## 0 引言

三维目标检测在自动驾驶<sup>[1]</sup>、机器人<sup>[2]</sup>等领域有着广泛的应用。通过估计空间目标的3D位置, 自动驾驶的车辆或机器人可以更准确地预判和规划自己的行为 and 路径, 避免碰撞和违规。目前, 三维目标检测按照数据类型可分为三类: 单目图像、多视图图像、点云目标检测。基于单视图的方法, 如GS3D<sup>[3]</sup>使用单目摄像头完成三维目标检测; 基于多视图的方法, 如Chen等<sup>[4]</sup>从不同视图的图像中得到的视差来获得深度图完成三维目标检测; 基于点云的方法, 如PointRCNN<sup>[5]</sup>、VoteNet<sup>[6]</sup>从点云获取目标信息完成三维目标检测。

目前基于点云的目标检测方法, 想获得更好的检测性能有三种方法: 一是用更好的主干网; 二是

[收稿日期] 2020-04-16

[基金项目] 国家自然科学基金项目(41971424, 61701191); 福建省教育厅科技项目(JAT190321, JAT190318, JAT190315)

[作者简介] 葛旭阳(1994—), 男, 硕士生, 从事点云目标检测、语义分割研究。通信作者: 蔡国榕(1979—), 男, 副教授, 硕导, 从事模式识别、计算机视觉、三维重建等的研究。E-mail: guorongcai.jmu@gmail.com

设计更好的策略提取更好的特征; 三是用并集上的交集 (Intersection over Union, IoU) 计算的度量损失来替代传统的边界框回归损失, 如  $l_1$  范数、 $l_2$  范数等。边界框回归是 2D/3D 视觉任务中一个最基础的模块, 不论是实例分割、目标跟踪, 还是目标检测, 都依赖于对边界框进行回归, 以获得准确的定位效果。因此, 选择更好的边界框回归损失, 有利于提高目标检测网络的性能。

IoU 又被称为 Jaccard 索引, 常用来表示两个任意形状目标的相似性, 在目标检测的边界框回归中起到重要的作用。在 anchor-based<sup>[7]</sup> 的方法中, IoU 不仅可以用来确定正样本和负样本, 还可以用来评价输出框和真实框的距离, 或者说预测框的准确性。IoU 还有一个好的特性, 就是尺度不变性。由于这个良好的属性, 所以在语义分割<sup>[8]</sup>、目标检测<sup>[9]</sup> 和跟踪<sup>[10]</sup> 等任务中的性能测量均采用 IoU 作为测量指标。

$l_n$  范数类型损失的一个较好的局部最优解可能并非 IoU 的局部最优解; 而且与 IoU 不同的是,  $l_n$  范数不具有尺度不变性, 相同重叠程度的边界框的损失值不一定相同。另外, 一些边界框的表示方法, 由于没有对不同类型的表示参数进行正则处理, 当表示参数变多时, 或在问题中添加更多维度时, 复杂性会增加。为了缓解上述问题, 目前最先进的对象检测器引入了锚盒 (anchor box)<sup>[11]</sup> 的概念, 他们还定义了一个非线性的表示<sup>[12-13]</sup> 来补偿尺度的变化。即使进行了这些手工更改, 优化回归损失和最大化 IoU 值之间仍然存在差距。

Rezatofighi 等<sup>[14]</sup> 探索了轴对齐矩形之间的 IoU, 此时 IoU 有解析解, 并且可反向传播, 意味着, IoU 可以直接用作目标函数进行优化。而优化 IoU 目标函数与优化某个损失函数之间, 显然选择优化 IoU 目标函数, 可以提高 IoU 指标的强相关。但是这其中也存在问题: 如果两个框不相交, 那么就无法衡量两个框的距离; 同时 IoU 的值为 0, 其梯度也将为 0, 当计算梯度的时候就无法使用优化器进行传播。针对这些问题, 提出了泛化版的 IoU, 称为 GIoU。GIoU 的优点有: a) 沿袭 IoU 能将目标形状属性编码进区域属性; b) 维持 IoU 的尺度不变性; c) 在目标有重叠情况下与 IoU 强相关。

本文在此基础上, 设计了一个适用于三维目标检测边界框回归的度量方法 (3D\_CGIoU), 将 3D\_CGIoU 的值纳入三维目标检测边界框回归的损失中, 并将该方法融入到目前主流的三维目标检测网络中, 如 PointRCNN、VoteNet 和 VoxelNet<sup>[15]</sup> 等, 评估其对检测性能的影响。

## 1 3D\_CGIoU 算法

三维目标检测中, 边界框回归的目标是尽可能准确地获取预测框。目前绝大部分的目标检测都会用 IoU 对候选预测框进行筛选, 多数的目标检测通过  $l_n$  范数来作为度量标准, 这将会存在两个预测框  $l_n$  范数的绝对值相同, 而和真实框的重叠方式却不不同的情况。另外,  $l_n$  范数对物体的尺度变化很敏感, 而 IoU 和 GIoU 具有尺度不变性可以更好地度量预测框的精准度。受到 GIoU 的启发, 本文设计了一个适用于三维目标检测边界框回归的度量和损失 (3D\_CGIoU) 的计算方法。

### 1.1 GIoU 的原理

IoU 是用来比较任意形状 (体积) 之间的相似性, 但是当预测框与真实框完全不重合时, 损失函数梯度为 0。另外, 预测框与真实框可以以不同的方式重叠, 得到相同的 IoU 值, 即 IoU 不能反映两个框之间如何发生重叠。

针对以上存在的问题, Rezatofighi 等<sup>[14]</sup> 设计了适用于二维目标检测的 GIoU。首先, 计算预测框和真实框的最小闭包区域面积 (同时包含了预测框和真实框的最小长方形的面积), 再计算闭包区域中不属于两个框的区域占闭包区域的比重, 最后用 IoU (其值记为  $I_1$ ) 减去这个比重得到 GIoU (其值记为  $I_2$ )。

IoU 和 GIoU 的计算方式如下:

$$I_1 = S_{\text{紫色}} / (S_{\text{紫色}} + S_{\text{绿色}} + S_{\text{蓝色}}), \quad (1)$$

$$I_2 = I_1 - S_{\text{黄色}} / (S_{\text{紫色}} + S_{\text{蓝色}} + S_{\text{绿色}} + S_{\text{黄色}})。 \quad (2)$$

公式 (1) 和公式 (2) 中的  $S$  指代各颜色区域面积, 各颜色区域如图 1 所示。

## 1.2 3D\_CGIoU 的原理

借鉴 GIoU 的方法, 本文设计了一个适用于三维目标检测边界框的度量方法, 该方法在鸟瞰图上以最小凸多边形为外接图形进行计算, 称为 3D\_CGIoU, 其值记为  $I_3$ , 则

$$I_3 = |A \cap B| / |A \cup B| - |C / (A \cup B)| / |C|. \quad (3)$$

其中:  $A$  代表真实框  $B_g$  的体积;  $B$  代表预测框  $B_p$  的体积;  $C$  表示封闭盒  $B_c$  的体积。

为了更好地理解在 3D\_CGIoU 中最小的封闭盒  $B_c$  的形状与计算, 给出了更直观展示的图 2。



绿色和蓝色边框分别表示真实框  $A$  和预测框  $B$ , 紫色表示  $A$  和  $B$  的交集, 红色包围框表示  $A$  和  $B$  的外接最小封闭框  $C$

Green and blue represent ground truth box  $A$  and the predicted bounding box  $B$ , respectively, and purple represents the intersection of them, the red bounding box represents the smallest enclosing box  $A$  of  $B$  and  $C$

图 1 一种二维边界框重叠方式

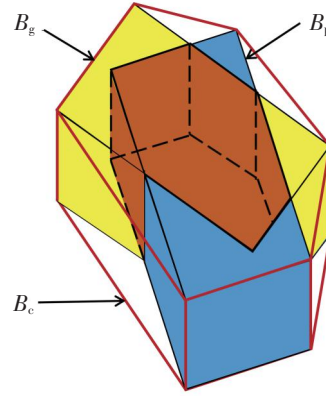
Fig.1 A situation of 2D bounding box overlapping

图 3 中预测框  $B_p$  的中心点、长、宽、高、倾角分别是  $A_0(x_p, y_p, z_p)$ 、 $l_p$ 、 $W_p$ 、 $H_p$ 、 $\theta_p$ ; 真实框  $B_g$  的中心点、长、宽、高、倾角分别是  $B_0(x_g, y_g, z_g)$ 、 $l_g$ 、 $W_g$ 、 $H_g$ 、 $\theta_g$ 。根据这些参数可以得到  $B_p$  和  $B_g$  顶点的坐标为:  $A_i(x_{pi}, y_{pi}, h_{pi})$ ,  $B_i(x_{gi}, y_{gi}, h_{gi})$ 。当  $i = 1, 2, 3, 4$  时,  $h_{pi} = z_p + H_p/2$ ,  $h_{gi} = z_g + H_g/2$ ; 当  $i = 5, 6, 7, 8$  时,  $h_{pi} = z_p - H_p/2$ ,  $h_{gi} = z_g - H_g/2$ 。

由于框的底部均平行于水平面, 因此只要从鸟瞰图中找出两个框的相交截面部分并求出面积  $S_0$ , 就可得到两个框相交部分的体积  $V_0$ :  $V_0 = S_0 H$ , 其中  $H = \min(\max h_p, \max h_g) - \max(\min h_p, \min h_g)$ 。

为了计算相交部分的面积  $S_0$ , 首先, 判断每个顶点是否在另一个矩形内部 (不包含边缘)。

如果没有顶点在另一个矩形内部, 则无相交部分存在; 如果有, 则保存并进行下一步, 计算 2 个矩形各边的交点 (不包含 8 个顶点) 并保存。接着, 将保存的所有点连起来组成一个凸多边形, 这个凸多边形就是相交的部分。将第一个保存的点, 与其他与其不相邻的点依次连接后, 凸多

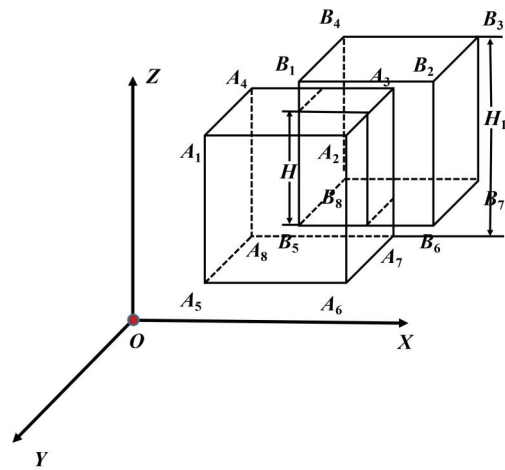


黄色和蓝色分别表示真实框  $B_g$  和预测框  $B_p$ , 棕色表示  $B_g$  和  $B_p$  的交集, 红色包围框表示  $B_g$  和  $B_p$  的外接最小封闭框  $B_c$

Yellow and blue represent ground truth box  $B_g$  and the predicted bounding box  $B_p$ , respectively, and brown represents the intersection of them, the red bounding box represents the smallest enclosing box  $B_c$  of  $B_g$  and  $B_p$

图 2 一种三维边界框重叠情况

Fig.2 A situation of 3D bounding box overlapping



$H$  为两个框交集的高度,  $H_1$  为两个框并集的高度

$H$  is the height of the intersection of them, and  $H_1$  is the height of the union of them

图 3 一种预测框和真实框的位置关系

Fig.3 A positional relation between the predicted bounding box and ground truth box

边形被分割成多个三角形。每个点坐标都可以计算得到, 显然三角形的每一条边长也可求得, 根据海伦公式算出每个三角形面积并求和, 最终得到  $S_0$  (见图 4)。

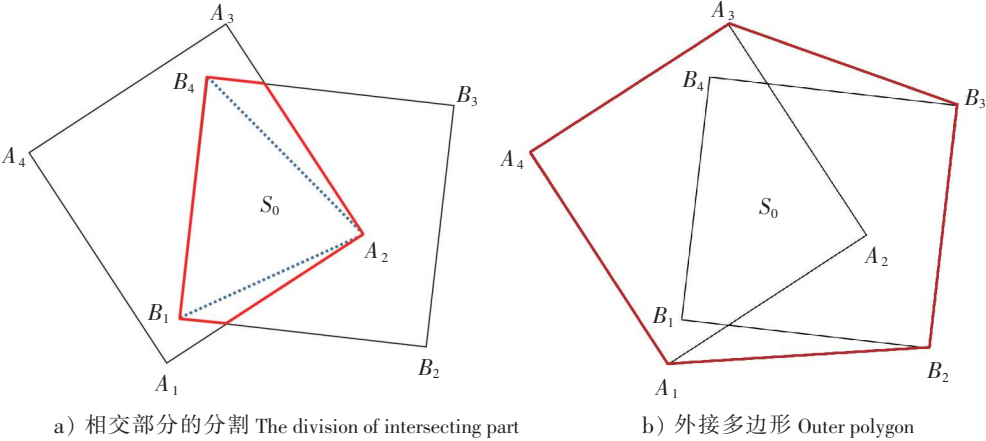


图 4 预测框和真实框的鸟瞰图

Fig.4 The bird's eye view of the predicted bounding box and ground truth box

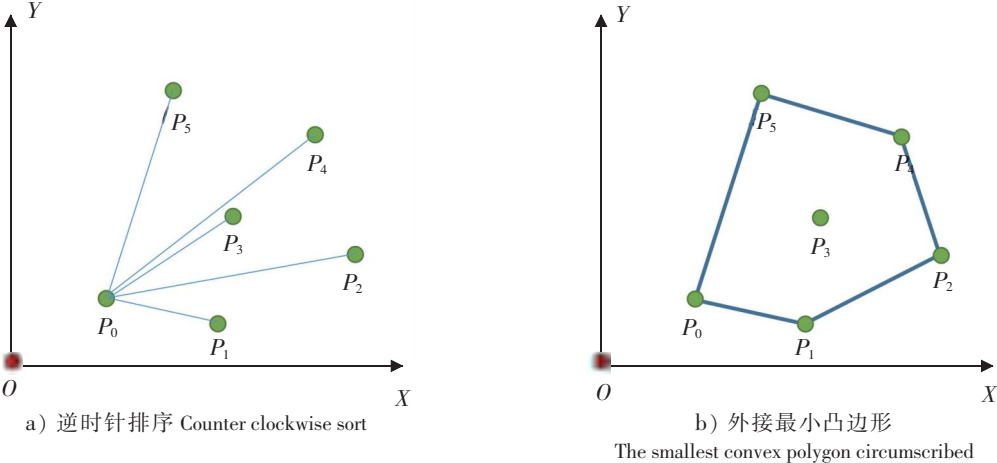
通过鸟瞰图外接框形状 (见图 5) 确定算法, 得到鸟瞰图中 2 个矩形的外接面积最小的凸多边形, 如图 4b 所示多边形  $A_1 B_2 B_3 A_3 A_4$ , 求其面积  $S_1$  的方法同  $S_0$ 。将这个凸多边形作为横截面所形成的棱柱并当作预测框和真实框的外接框, 计算出其体积  $V_1: V_1 = S_1 H_1$ , 其中  $H_1 = \max(\max h_p, \max h_g) - \min(\min h_p, \min h_g)$ 。

将 GIoU 从二维目标检测迁移到三维目标检测中, 并对鸟瞰图的外接框进行优化, 得到 3D\_CGIoU 的值  $I_3$ :

$$I_3 = V_0 / (V_p + V_g - V_0) - (V_1 - (V_p + V_g - V_0)) / V_1。$$

(4)

其中:  $V_p$  为预测框  $B_p$  的体积;  $V_g$  为真实框  $B_g$  的体积。



所有点(除  $P_0$ )通过与  $P_0$  的极角逆时针进行排序; 蓝色边框的凸多边形  $P_0 P_1 P_2 P_3 P_4 P_5$  为所有点的外接最小凸多边形  
All points (except  $P_0$ ) are sorted counterclockwise by polar angle with  $P_0$ ; The convex polygon with blue border  $P_0 P_1 P_2 P_3 P_4 P_5$  is the smallest convex polygon circumscribed at all points

图 5 鸟瞰图外接框算法的示意图

Fig.5 Schematic diagram of algorithm which is the circumscribed box of bird's eye view

1.3 鸟瞰图外接框的算法原理及流程

首先, 要确定预测框和真实框的鸟瞰图外接框的形状及大小。

第一步: 确定鸟瞰图中最左下角的点 (坐标排序:  $x$  最小, 如果  $x$  相同则取  $y$  更小的), 记为  $P_0$ 。这一步需要扫描一遍所有的点, 时间复杂度为  $O(n)$ ,  $n$  为点的总数量。在该算法中,  $n$  为 8, 因此

时间复杂度为  $O(1)$ 。

第二步: 将所有的点按照相对于第一步中得到的点  $P_0$  的极角大小进行排序。当极角相同时, 距离  $P_0$  比较近的排在前面。如图 5a 所示, 排序结果为  $P_1, P_2, P_3, P_4, P_5$ 。

第三步: 用一个栈 (数组) 来保存当前的凸多边形的顶点, 先将  $P_0$  和  $P_1$  依次加入到栈中。按顺序扫描每一个点, 用叉积判断当前点和栈顶头两个点形成的拐向。若顺时针就弹出栈顶元素, 接着继续判断; 否则压入新点  $P_i$ 。

然后, 栈中的点则为外接最小凸多边形的所有顶点, 按照其顺序连接起来即可, 如图 5b 所示。

鸟瞰图外接框的算法流程如下:

输入 鸟瞰图的 8 个顶点坐标, 如图 3 中  $A_1A_2A_3A_4B_1B_2B_3B_4$

输出 鸟瞰图中所有点的外接凸多边形的顶点

步骤 1)  $P_0 \leftarrow$  The bottom left point // 最左下角的点  
 2)  $P_1, P_2, \dots, P_7$  // 其他点与最左下角点之间的极角逆时针进行排序  
 3) let  $S$  be an empty stack // 用来存放最终得到外接凸多边形的顶点  
 4) PUSH( $P_0, S$ ) // 将前三个点作为初始点, 并压入栈  
 5) PUSH( $P_1, S$ )  
 6) PUSH( $P_2, S$ )  
 7) for  $i = 3$  to 7  
 8) while the angle formed by points NEXT-TO-TOP( $S$ ), TOP( $S$ ), and  $P_i$  makes a nonleft turn  
 9) POP( $S$ )  
 10) PUSH( $P_i, S$ )  
 11) return  $S$

#### 1.4 3D\_CGIoU 作为损失的算法流程

3D\_CGIoU 作为边界框回归损失算法流程如下:

输入 预测框  $p = \{x_p, y_p, z_p, l_p, W_p, H_p, \theta_p\}$ , 预测框对应的真实框  $g = \{x_g, y_g, z_g, l_g, W_g, H_g, \theta_g\}$

输出  $L_{3D\_CGIoU}$

步骤 1)  $S_A = l_p \times W_p$  // 预测框  $B_p$  俯视图  $A_1A_2A_3A_4$  的面积  
 2)  $S_B = l_g \times W_g$  // 真实框  $B_g$  俯视图  $B_1B_2B_3B_4$  的面积  
 3)  $S_0$  // 预测框  $B_p$  和真实框  $B_g$  俯视图相交的面积  
 4)  $S_1$  // 预测框  $B_p$  和真实框  $B_g$  俯视图外接最小凸多边形的面积  
 5)  $H$  // 预测框  $B_p$  和真实框  $B_g$  交集的高度  
 6)  $H_1$  // 预测框  $B_p$  和真实框  $B_g$  并集的高度  
 7)  $V_p = S_A \times H_p$  // 预测框  $B_p$  的体积  
 8)  $V_g = S_B \times H_g$  // 真实框  $B_g$  的体积  
 9)  $V_1 = S_1 \times H_1$  // 预测框  $B_p$  和真实框  $B_g$  最小封闭盒的体积  
 10) If  $S_0 \leq 0$ :  
      $V_0 = 0$ ;  
   Else:  
     If  $H \leq 0$ :  
        $V_0 = 0$ ;  
     Else:  
        $V_0 = S_0 \times H$ ;  
 11)  $I_3 = V_0 / V - (V_1 - V) / V_1$ , where  $V = V_p + V_g - V_0$

$$12) L_{3D\_CGIoU} = 1 - I_3$$

## 2 实验与分析

### 2.1 实验设置

本算法在 Linux 系统下,使用 Pytorch 编程,分别在不同数据集和主干网上进行实验。通过 PointRCNN、VoxelNet<sup>[15]</sup>、PointPillars<sup>[16]</sup>在 KITTI 基准数据集<sup>[17]</sup>上的表现,以及 VoteNet 在 ScanNet 数据集<sup>[18]</sup>上的表现,对本文提出的方法进行了评估。KITTI 数据集包含 7481 个训练样本和 7518 个测试样本。由于测试数据集的真实性是不公开的,因此将训练数据集分为训练集(3712)和验证集(3769)<sup>[19]</sup>。ScanNet 数据集一共有 1513 个采集场景数据(每个场景中点云数量都不一样,如果用端到端的网络框架,需要对原始点云进行采样,使每一个场景的点云数量都相同),21 个类别的对象,其中 1201 个场景用于训练,312 个场景用于验证。

实验所采用的每种方法中,都使用原作者提供的默认参数和每个基准上的迭代次数,并严格遵循其训练协议。实验结果中每个方法的检测精度,是网络通过训练后,在验证集上展现的精度。

### 2.2 实验结果与分析

通过多种方法在 KITTI 数据集上的表现,评估 3D\_CGIoU 算法融入主干网后对检测性能的影响,结果如表 1 所示。本实验对场景中的汽车进行检测,IoU 阈值为 0.7,评估指标为平均精度。所有的检测结果都使用官方的 KITTI 评估检测指标进行测量,这些指标包括:鸟瞰图、3D、2D 和平均方向相似度。二维检测是在图像平面上进行的。KITTI 数据集被划分为容易的、中等的和困难的三种难度,而官方的 KITTI 排行榜则根据中等的性能进行排序。

相比于二维目标检测,三维目标检测更具挑战性,因为它对三维包围盒在空间中的定位精度要求更高。从表 1 可以看出,将 3D\_CGIoU 算法纳入到边界框回归损失后的主干网对车辆的检测性能明显都要优于原始主干网。所采用的方法在各个难度上的检测性能均有提升,VoxelNet 提升了 1.89%,VoxelNet 提升了 1.79%,PointRCNN 提升了 1.03%。

表 1 在 KITTI 验证集的汽车类上进行目标检测的性能对比

Tab. 1 Performance comparison of object detection on the car class of KITTI validation set

方法 Method	车辆 Car(IoU = 0.7)		
	易 Easy	中 Moderate	难 Hard
VoxelNet	77.97	61.46	58.85
PointPillars	78.05	72.98	64.56
PointRCNN	82.20	75.00	74.07
VoxelNet + 3D_CGIoU	79.79	63.32	60.84
PointPillars + 3D_CGIoU	80.57	73.42	66.97
PointRCNN + 3D_CGIoU	83.99	75.67	74.70

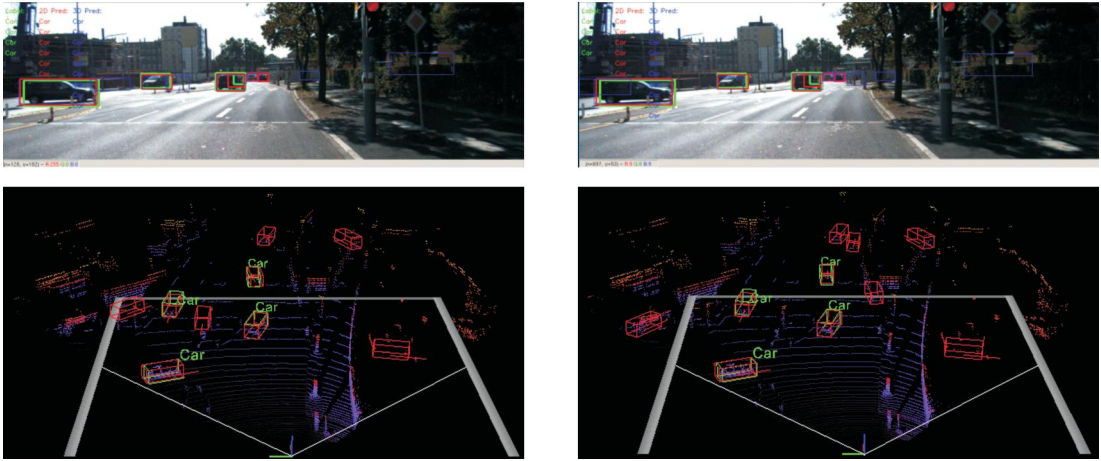
表 2 为 VoteNet 在 ScanNet 数据集上各类目标的平均检测精度,其中 baseline 为原始的 VoteNet 检测结果,ours 为将  $L_{3D\_CGIoU}$  纳入预测框回归损失后的 VoteNet 检测结果。如表 2 所示,ours 的平均精度和平均召回率都要优于 baseline。在 VoteNet 主干网中融入 3D\_CGIoU 算法优化边界框的回归损失,其检测性能在各类别目标上都有提升,所有类别的平均精度提升了近 1.5 个百分点。其中,“柜子”和“冰箱”的检测性能提升差距过大。这两者形状都比较规则,也都是相对较大的物体,且 GIoU 有尺度不变性,因此排除由于大小或形状而导致的精度差异,推测是由于 baseline 在训练过程中由于网络初始化的随机性等问题,导致对“柜子”的表现低于应有的水准。但在本文算法中,“柜子”的检测性能表现较好,而“冰箱”可能与“柜子”的情况恰恰相反,最终使得“柜子”和“冰箱”的精度提升幅度差距过大。



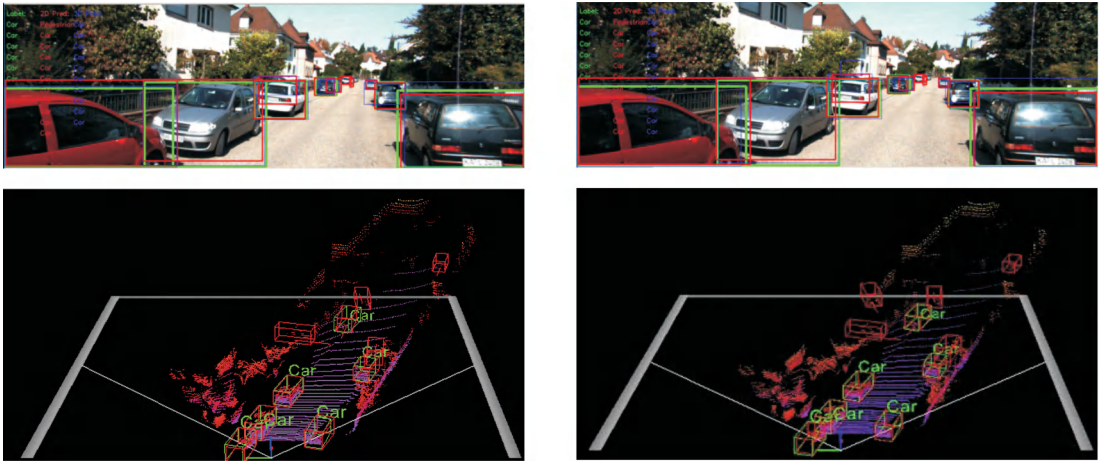
表 2 VoteNet 在 ScanNet 数据集上的平均检测精度  
Tab. 2 Average detection accuracy of VoteNet on ScanNet dataset

目标 Object	Baseline	Ours	目标 Object	Baseline	Ours
柜子 Cabinet	34.81	38.64	办公桌 Desk	64.38	65.28
床 Bed	88.43	90.57	窗帘 Curtain	49.60	50.83
椅子 Chair	87.43	89.67	冰箱 Refrigerator	48.90	49.13
沙发 Sofa	89.21	91.84	浴帘 Shower curtain	69.81	70.68
桌子 Table	58.93	60.12	马桶 Closetool	96.79	97.34
门 Door	43.98	44.85	洗涤槽 Sink	51.74	53.45
窗 Window	37.10	38.09	浴缸 Bathtub	90.72	92.21
书架 Bookshelf	41.51	42.43	垃圾桶 Bin	36.92	37.47
图画 Picture	4.30	5.97	mAP	58.24	59.61
柜台 Counter	53.64	54.34	AR	80.67	81.07

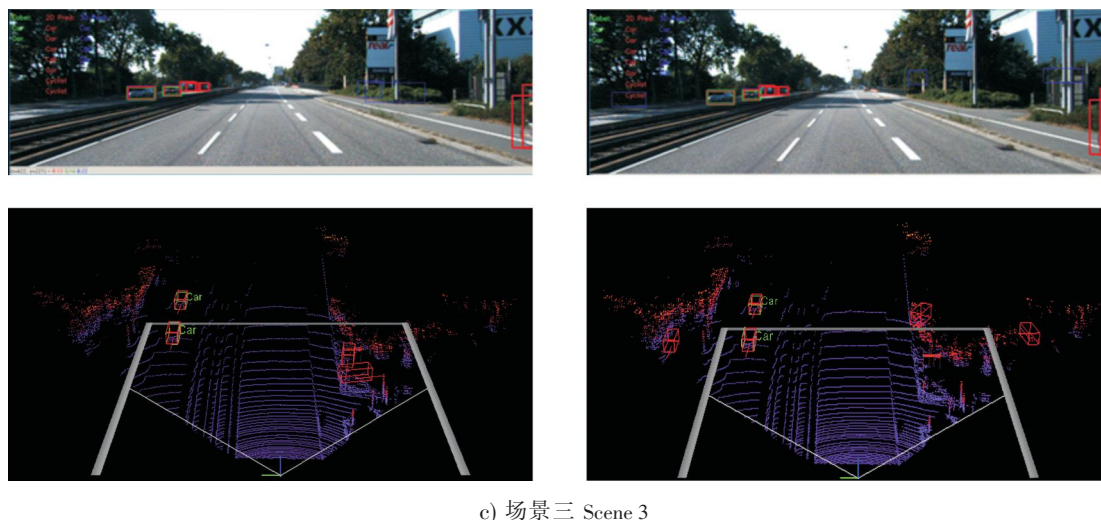
图 6 是 PointRCNN 和 PointRCNN + 3D\_CGIoU 在 KITTI 上的检测结果，可视化其中的三个场景，左侧为原始的 PointRCNN 检测结果可视化，右侧为将  $L_{3D\_CGIoU}$  纳入预测框回归损失后的 PointRCNN + 3D\_CGIoU 检测结果可视化。实物图下方是对应的点云图。实物图中蓝色的是预测框，红色是真实框，绿色是 label；点云图中红色的是预测框，绿色是真实框。实物图中并不是所有的红色框都作为 ground truth (GT)，只有有 label 的框才会在训练中作 GT。



a) 场景一 Scene 1



b) 场景二 Scene 2



c) 场景三 Scene 3

图6 PointRCNN 和 PointRCNN+3D\_CGloU 在 KITTI 上的检测结果可视化

Fig.6 Visualization of detection results of PointRCNN and PointRCNN+3D\_CGloU on KITTI

点云图的左右对比显示, 3D\_CGloU 算法纳入边界框回归损失后, 预测框与真实框重合度有一定的提升, 即得到的预测框的准确率更高。场景一中, 在实物图中可以很明显地看出道路的远处有 2 辆车, 但没有标签; 在点云图中, PointRCNN 只检测出了 1 辆车, 而 PointRCNN + 3D\_CGloU 将 2 辆车都检测了出来。这进一步说明了 3D\_CGloU 的有效性和优越性。

### 3 结论

本文设计了 3D\_CGloU 作为新的三维点云目标检测的评估指标, 对预测框与真实框进行相似性比较。3D\_CGloU 继承了 IoU 的优秀特性且完善了其缺点 (非重叠情况), 因此, 在基于 IoU 作为评估指标的一些三维计算机视觉任务中, 相比于 IoU, 3D\_CGloU 是更好的选择。将 3D\_CGloU 计算损失的方法融入目前主流的三维目标检测算法中, 如 PointRCNN、VoxelNet、PointPillars, 在 KITTI 数据集和 ScanNet 数据集上其检测性能均得到有效提升, 平均精度提高了近 2 个百分点。实验结果表明, 在三维边界框回归的应用中, 可以通过 3D\_CGloU 方法优化回归损失, 从而提高检测的精确度。

### [ 参考文献 ]

- [1] CHEN X, MA H, WAN J, et al. Multi-view 3D object detection network for autonomous driving [C] //2017 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2017: 1907-1915.
- [2] EKVALL S, KRAGIC D, JENSFELT P. Object detection and mapping for service robot tasks [J]. Robotica, 2007, 25 (2): 175-187.
- [3] LI B, OUYANG W, SHENG L, et al. Gs3D: an efficient 3D object detection framework for autonomous driving [C] //2019 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2019: 1019-1028.
- [4] CHEN X, KUNDU K, ZHU Y, et al. 3D object proposals using stereo imagery for accurate object class detection [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 40 (5): 1259-1272.
- [5] SHI S, WANG X, LI H. Pointrcnn: 3D object proposal generation and detection from point cloud [C] //2019 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2019: 770-779.
- [6] LANG A H, VORA S, CAESAR H, et al. Votenet: fast encoders for object detection from point clouds [C] //2019 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2019: 12697-12705.



- [7] CHEN M, WANG Q, LI X. Anchor-based group detection in crowded scenes [C] //2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Piscataway, NJ: IEEE, 2017: 1378-1382.
- [8] CORDTS M, OMRAN M, RAMOS S, et al. The cityscapes dataset for semantic urban scene understanding [C] //2016 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2016: 3213-3223.
- [9] LIN T Y, MAIRE M, BELONGIE S, et al. Microsoft COCO: common objects in context [C] //2014 European Conference on Computer Vision. Cham: Springer, 2014: 740-755. DOI:10.1007/978-3-319-10602-1\_48.
- [10] LEAL-TAIXÉ L, MILAN A, REID I, et al. Motchallenge 2015: towards a benchmark for multi-target tracking [J]. arXiv preprint arXiv: 1504. 01942, 2015.
- [11] REN S, HE K, GIRSHICK R, et al. Faster r-cnn: towards real-time object detection with region proposal networks [C] //Advances in Neural Information Processing Systems. New York: Curran Associates, 2015: 91-99.
- [12] REDMON J, DIVVALA S, GIRSHICK R, et al. You only look once: unified, real-time object detection [C] //2016 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NY: IEEE, 2016: 779-788. DOI:10.1109/CVPR.2016.91.
- [13] GIRSHICK R, DONAHUE J, DARRELL T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation [C] //2014 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2014: 580-587.
- [14] REZATOFIGHI H, TSOI N, GWAK J Y, et al. Generalized intersection over union: a metric and a loss for bounding box regression [C] //2019 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NY: IEEE, 2019: 658-666.
- [15] ZHOU Y, TUZEL O. Voxelnet: end-to-end learning for point cloud based 3D object detection [C] //2018 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2018: 4490-4499.
- [16] LANG A H, VORA S, CAESAR H, et al. Pointpillars: fast encoders for object detection from point clouds [C] //2019 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NY: IEEE, 2019: 12697-12705.
- [17] GEIGER A, LENZ P, URTASUN R. Are we ready for autonomous driving? the kitti vision benchmark suite [C] //2012 IEEE Conference on Computer Vision and Pattern Recognition. Piscataway, NY: IEEE, 2012: 3354-3361.
- [18] DAI A, CHANG A X, SAVVA M, et al. Scannet: richly-annotated 3D reconstructions of indoor scenes [C] //2017 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Piscataray, NY: IEEE, 2017: 5828-5839.
- [19] CHEN X, KUNDU K, ZHU Y, et al. 3D object proposals for accurate object class detection [C] //Advances in Neural Information Processing Systems. New York: Curran Associates, 2015: 424-432.

(责任编辑 朱雪莲 英文审校 黄振坤)