一文读懂PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection

原创 hughlee815 2020-06-23 17:05:33 315 收藏 1

文章标签: 算法 人工智能 3d 计算机视觉

为了致敬*一文读懂Faster RCNN*，本文取名如此。希望能帮助大家以轻松的姿态理解这一霸榜kitti大半年的3D目标检测算法。

0. 立论依据

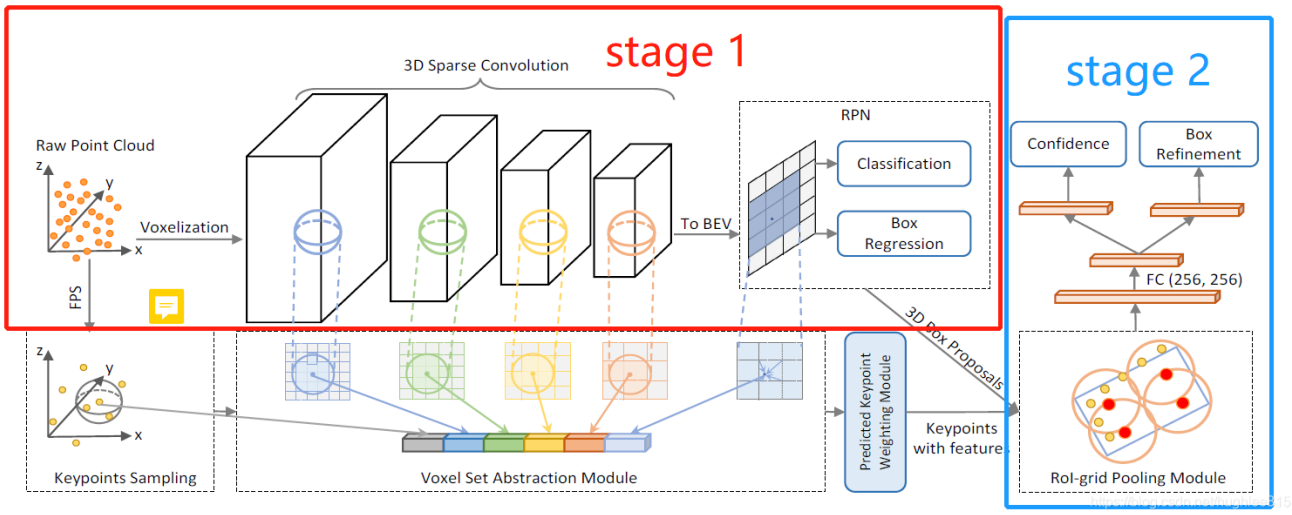
按照点云的前处理方法划分，通常3D目标检测可以分为基于体素和基于点的方法。文中提到二者有其各自的特点：基于体素的方法能有效地进行region proposal，但是感受野受到卷积核大小的限制。而基于点的方法恰好能够获得灵活的感受野，更捕获更精确的上下文信息。（？？此处迷惑，即使是基于体素的方法的感受野难道不是随着卷积层数的增加而逐渐扩大的吗？？讲真，大家都说二个方法有其各自特点，但是目前我还没有见过谁把二者特点真正给讲清楚的）

anyway，作者希望能够将二者结合。

那么如何结合呢？一个直觉的想法是，第一阶段用voxel-base方法得到proposal，再用point-based的方法对proposal中的所有点进行第二阶段的refine。但是因为这样非常占显存，作者认为不可行。后面的消融实验也证明了这样做的效果并不如pvrcnn好。

那么作者究竟是怎么做的呢？

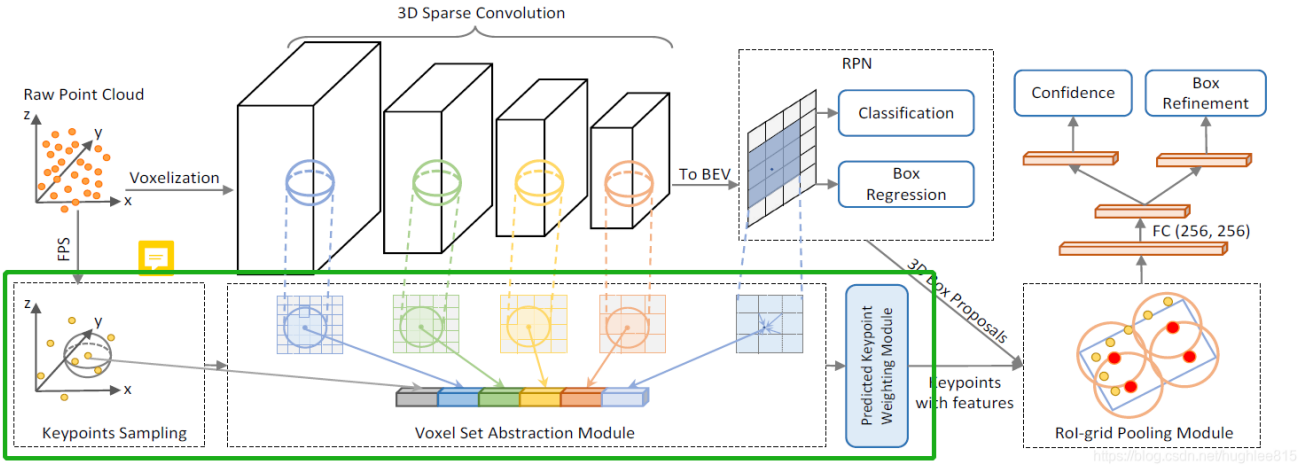
1. stage1：region proposal



如上图所示，pvrcnn是一个两阶段检测算法。stage1采用常规的voxel-based的方法得到proposal。具体来说，以kitti数据集为例（作者还在waymo数据集上进行了实验），将原始点云空间（x:(0，70.4)；y:(-40，40)；z:(-3，1))划分为(0.05m,

0.05m, 0.1m)大小的体素，再用VFE(参见论文voxelnet）提取每个体素的特征，再使用3D稀疏卷积（参见论文second）进行下采样，对特征升维。文中分别为1×，2×,4×, 8×倍下采样，维度分别为16, 32, 64, 64（下采样必然需要更高维的特征向量来承载相同的信息量）。
搞明白了卷积怎么回事，再来看看图1中的***to BEV***是如何实现的。前面提到z方向上整个空间的高度是4m，体素高度为0.1米，所以z方向上共有40个体素。由于最高进行了8倍下采样，所以第4层卷积后z方向有5个体素，每个体素的特征向量为64维。***to BEV**所进行的操作就是把这*5*个体素合并为*1*个修长的体素，特征向量进行拼接，变为*64*5=320维。这样后续的region proposal过程就能进行快捷的2D卷积了。
stage1后续得到RoI的操作和2D检测就很像了，此处不再赘述。

2. stage 2：refine
如前文所述，得到RoI后，作者并不是把ROI中的每个点特征用来进一步的refine操作，而是基于关键点的特征来进行refine。那么关键点的特征是哪里来的呢？这就是下图中绿框部分的工作。



由上图可以看出，作者首先使用最远点采样算法（关于点云的各种采样算法的特点及实现可以参看我的另一篇博文）从原始点云中采样了一部分点，文中称作关键点（keypoint），并用这部分关键点来代表点云的全貌。再将每一个关键点找到在“特征体”（就是图中4个白色的立方体）中对应的位置。（ps：具体是怎么对应的我没搞清楚，文中貌似也没有说。难道是根据坐标的比例关系？）
anyway, 找到对应的点之后，就开始利用pointnet++中的set abstraction模块提取周围某邻域球体内的特征了。并将在4个“特征体”中提取到的特征拼接起来，作为这个关键点的特征。作者认为这样还不够，于是他把set abstraction模块在采样到的关键点和鸟瞰图上再操作一遍，就得到6个尺度的特征，并把他们concatenate起来，代表这个关键点最终的特征。后面的消融实验证明作者加入这两个尺度的特征对提升精度确实是用一定帮助的。

上面说的只是对一个关键点的操作，其他关键点完全一样。最终，我们可以得到每个关键点的特征。

经过stage1我们得到了RoI, 经过刚刚的关键点特征提取我们得到了每个关键点的特征。好了，现在我们终于可以进行refine了。不对，等等，还有一个Predicted Keypoint Weighting模块。它的作用主要是想降低不是前景点的关键点特征对refine阶段的影响。它的实现也很简单，见下图。通过训练两层MPL来使得模型能够区分哪些是前景点，哪些是背景点，并对背景点赋予较小的权重。

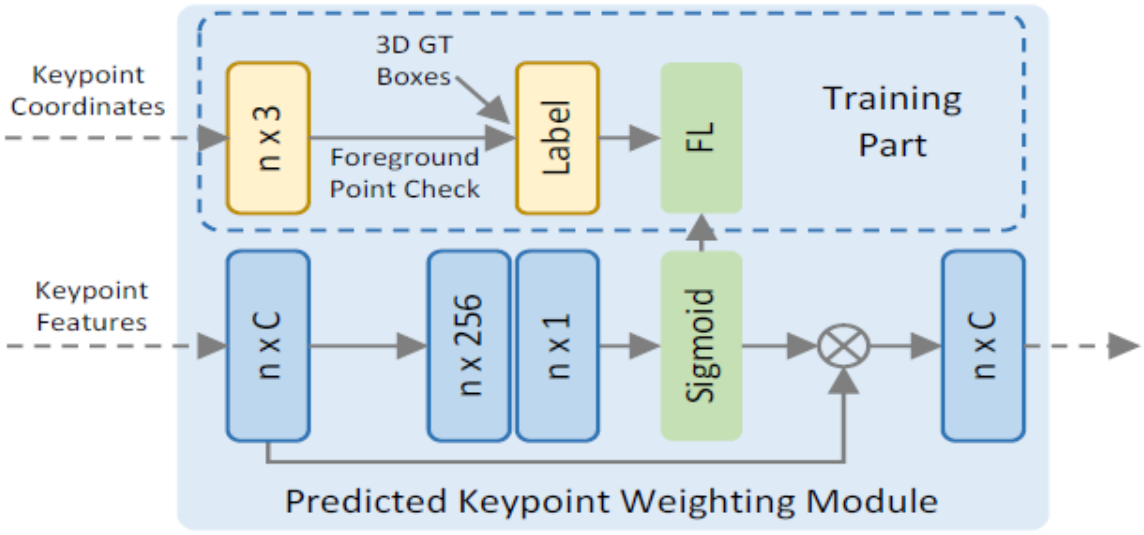
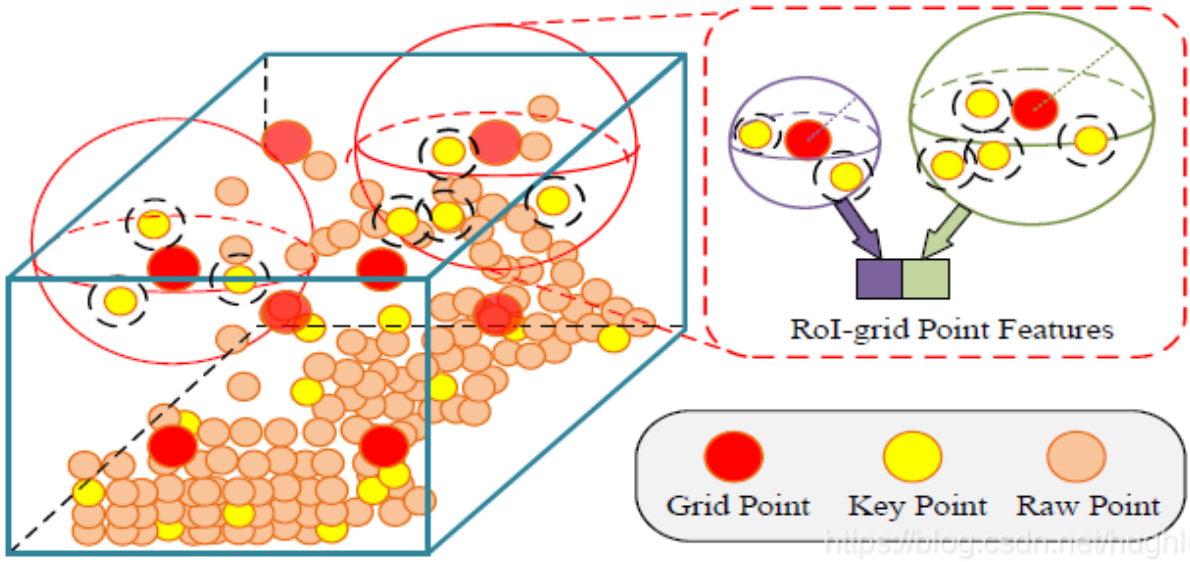


Figure 3. Illustration of Predicted Keypoint Weighting module.

发现没有？其实到目前为止，作者已经将voxel-based方法和point-based方法结合起来了。但是，精彩还在继续，作者在refine阶段又将两个方法结合了一遍。
在第一阶段得到的RoI中，作者随机生成了6 * 6 * 6个格子点（grid point），如下图。注意只看左边和右下角就好了，右上角可以先不管。



注意上图中的key point和raw point都是点云中真实存在的点，而grid point是生成的参考点。其实refine阶段raw point并没有什么用，因为它的特征已经被“吸收”进关键点了，作者在途中画出来只是让你知道有这么个玩意儿。当然grid point更不提供特征，他只是用来标记位置的。
具体来说，作者以gird point为球心，以某一设定值为半径画球，对包括在其中的关键点再次进行set abstraction操作，得到更高级的特征。这样做有一个好处就是，在画球的过程中，有可能将RoI之外的点包括进来，从而提供更丰富的语义信息，帮助模型更好的回归。这样重复6 * 6 * 6次，就能得到6 * 6 * 6个特征向量。现在大家应该能明白上图中右上角的含义了。
得到整个RoI的特征后，就可以进行confidence和regression预测了。值得注意的是，此处的confidence预测作者采用了一篇ECCV2018论文（Acquisition of Localization Confidence for
Accurate Object Detection）的思想，不是直接预测概率，而是预测iou。

到这里，整个模型就介绍完毕了，最后的结果大家应该都清楚，效果一骑绝尘。但是据说被电子科大的新研究超越了？
anyway，最后表达一下我个人的看法：

1. 其实除了point-based和voxel-based的方法还有graph-based的方法。是不是可以排列组合一下，融合多种方法？那按这个道理是不是point-voxel-graph-based的方法最无敌，hhh。可能速度会很慢吧，是不是可以考虑把graph-based的方法作为辅助网络呢，就像阿里达摩院的sa-ssd那样。
2. ……