ImpDet: Exploring Implicit Fields for 3D Object Detection

Xuelin Qian
Fudan University
xlqian@fudan.edu.cn

Li Wang
Agora
wangli@agora.io

Yi Zhu*
Amazon Inc.
yzhu25@ucmerced.edu

Li Zhang
Fudan University
lizhangfd@fudan.edu.cn

Yanwei Fu†
Fudan University
yanweifu@fudan.edu.cn

Xiangyang Xue
Fudan University
xyxue@fudan.edu.cn

Supplementary Materials

The supplementary document is organized as follows:

- Sec. A elaborates the structure of backbone network in detail.
- Sec. B describes the formulations of loss objectives and presents the experimental study of coefficients.
- Sec. C provides some insightful discussion of the candidate shifting module.
- Sec. D reports experimental analyses of training configuration and discrepancy.
- Sec. E shows more visualizations of object detection and implicit fields to further prove the efficacy and robustness of our proposed ImpDet.

A. The Structure of Backbone Network

Figure 1 shows the details of our backbone network. (1) *For the point-wise branch*, we first feed raw point clouds $\mathcal{P} = \{x_i, y_i, z_i, r_i\}_{i=1}^{N}$ into a MLP for initial point-wise features $f^{(p_0)} \in \mathbb{R}^{N\times 16}$, where (x_i, y_i, z_i) and r_i mean the coordinates and intensity of point p_i, N is the total number of points. Then, we utilize two stacked VFE layers [7] with filter numbers of 32 and 64 to obtain initial voxel-wise features $f^{(v_0)} \in \mathbb{R}^{N\times 64}$, where each voxel maintains a feature vector for points fall in it. The initial point-wise features with the dimension of 16 are subsequently concatenated with the feature of voxel they fall in. After another MLP layer, we obtain the final point-wise features $f^{(point)} \in \mathbb{R}^{N\times 80}$.

(2) *For the voxel-wise branch*, it consists of five blocks, where the first and the last blocks have one sparse convolution layer, and the three middle blocks contain three layers. Except for the first block, we set the stride of the first layer in each block as 2 to gradually down-sample the features. Specifically, in the last block, we only reduce the height of the feature map by half. To maintain the sparsity, we utilize the submanifold sparse convolution layer in the last two layers of the second, third and fourth blocks. The output channels of five blocks are defined as $64 \rightarrow 32 \rightarrow 64 \rightarrow 64 \rightarrow 128$. Given an initial voxel features $f^{(v_0)}$, we utilize these five blocks to progressively get multi-scale voxel-wise features $f^{(v_1)} \sim f^{(v_5)}$. Similar to [1], we compress the voxel-wise tensor $f^{(v_5)}$ by concatenating features along z-axis, and further apply a FPN [2] structure. By concatenating output features, we get 2D Birds-Eye-View (BEV) map features $f^{(bev)} \in \mathbb{R}^{L\times W\times C}$. Specifically, we adopt 2 convolutional layers and 2 deconvolutional layers as FPN structure.

B. The Details of Loss Objectives

The overall loss functions are composed of six terms, *i.e.*, the candidate shifting loss, the centerness confidence loss, the implicit function loss, the classification loss, the box refinement loss and the direction prediction loss,

$$\begin{aligned}\mathcal{L} = \lambda_1\mathcal{L}_{ofs} + \lambda_2\mathcal{L}_{ctrns} + \lambda_3\mathcal{L}_{imp} \\ + \lambda_4\mathcal{L}_{cls} + \lambda_5\mathcal{L}_{box} + \lambda_6\mathcal{L}_{dir}\end{aligned} \tag{1}$$

where λ_i is the coefficient to balance each term. As illustrated in Fig. 2, we conduct experimental studies to analyze some coefficients, which are commonly important or associated with implicit filed, and empirically set others by default [5, 1]. As observed, we set $\lambda_1 = \lambda_2 = \lambda_4 = 1.0$, $\lambda_3 = \lambda_5 = 2.0$ and $\lambda_6 = 0.2$ for all experiments due to its better performance.

For the first three objectives, we denote the symbols with

*Work done outside Amazon

†Corresponding author. Dr. Fu is also with Fudan ISTBI—ZJNU Algorithm Centre for Brain-inspired Intelligence, Zhejiang Normal University, Jinhua, China

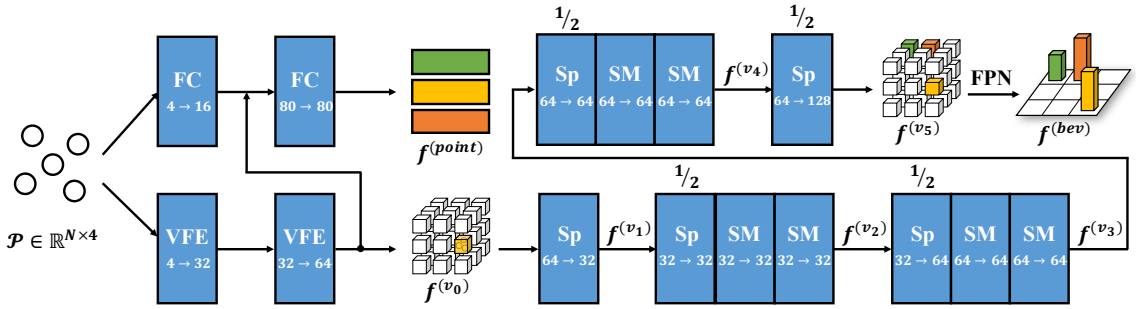


Figure 1. The structure of backbone network. 'SP' and 'SM' denote the sparse convolution layer and the submanifold sparse convolution layer respectively; '$1/2$' means the stride of this layer is 2. The numbers on each rectangle represent the channels of input and output.

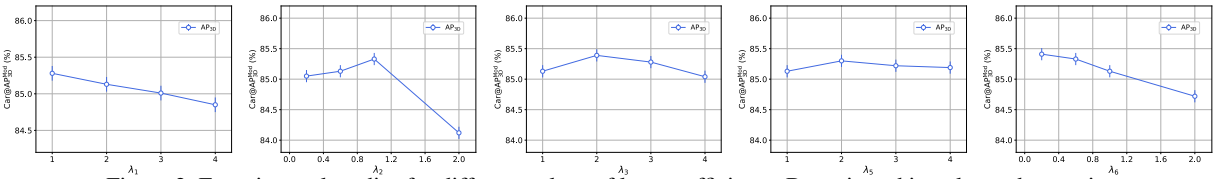


Figure 2. Experimental studies for different values of loss coefficients. Best viewed in color and zoom in.

hat '$\wedge$' as ground truth, each formulation can be defined as,

$$\mathcal{L}_{ofs} = \frac{1}{|\mathfrak{N}_{pixel}|} \sum_{i \in \mathfrak{N}_{pixel}} \mathcal{L}_{smooth_{L1}} \left(p_i^{(ofs)}, \widehat{p_i^{(ofs)}} \right) \tag{2}$$

$$\mathcal{L}_{ctrns} = \frac{1}{|\mathfrak{N}_{pixel}|} \sum_{i=1}^{HW} \mathcal{L}_{focal} \left(s_i^{(ctrns)}, \widehat{s_i^{(ctrns)}} \right) \tag{3}$$

$$\mathcal{L}_{imp} = \frac{1}{|\mathfrak{N}_{center}|} \sum_{i \in \mathfrak{N}_{center}} \mathcal{L}_{BCE} \left(\mathcal{H}_i, \widehat{\mathcal{H}_i} \right) \tag{4}$$

where $\mathfrak{N}_{pixel}$ and $\mathfrak{N}_{center}$ indicate the set of indices of positive pixels/candidate centers if they are inside objects' bounding boxes; '$|\cdot|$' means the cardinality.

We adopt 3D Intersection-over-Union (IoU) between the generated bboxes and corresponding ground-truth boxes for the classification branch. The IoU is further normalize as the training targets to compute the loss,

$$\begin{aligned} \widehat{s_i^{(iou)}} &= \min\left(1,\ \max\left(0,\ 5\text{IoU}_i - 2\right)\right) \\ \mathcal{L}_{cls} &= \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}_{BCE} \left(s_i^{(iou)}, \widehat{s_i^{(iou)}} \right) \end{aligned} \tag{5}$$

where k is the number of sampled top-k candidate centers, $s_i^{(iou)}$ is the predicted classification confidence

For the box refinement branch, we follow [5, 3] to predict the residuals of center location, box size and orientation based on the generated boundary boxes, as

$$\mathcal{L}_{box} = \frac{1}{K} \sum_{i=1}^{K} \mathbb{1}\left(\text{IoU}_i \geq 0.55\right) \mathcal{L}_{smooth_{L1}} \left(b_i, \widehat{b_i} \right) \tag{6}$$

where b_i and $\widehat{b_i}$ are the predicted and ground-truth box residuals. $\mathbb{1}(*)$ denotes the indicator function, its value equals to 1 if $*$ is true, otherwise, returns 0.

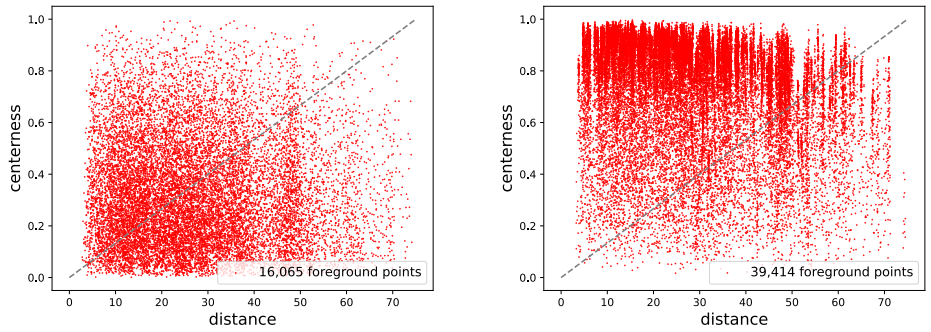
The direction branch outputs a binary value to determine whether the boundary orientation needs to be flipped, so that the range of orientation can be extended from $[0, \pi)$ to $[0, 2\pi)$. The loss is defined as,

$$\mathcal{L}_{dir} = \frac{1}{K} \sum_{i=1}^{K} \mathbb{1}\left(\text{IoU}_i \geq 0.55\right) \mathcal{L}_{cross\_entropy} \left(d_i, \widehat{d_i} \right) \tag{7}$$

where d_i and $\widehat{d_i}$ are the predicted and ground-truth direction.

C. More Discussions of Candidate Shifting

Our proposed ImpDet shows a new perspective that directly benefits from the implicit field learning to achieve more precise 3D object detection. However, performing the implicit function for every potential object in global 3D space would be very computationally expensive, thereby we introduce the candidate shifting module to dramatically reduce the computational costs. It first *shifts* points to build a series of 3D local spaces centered on points, then *samples* points to select local spaces that are most likely to contain



(a) Before point shifting (b) After point shifting

Figure 3. Centerness statistics of foreground points. Best viewed in color and zoom in.

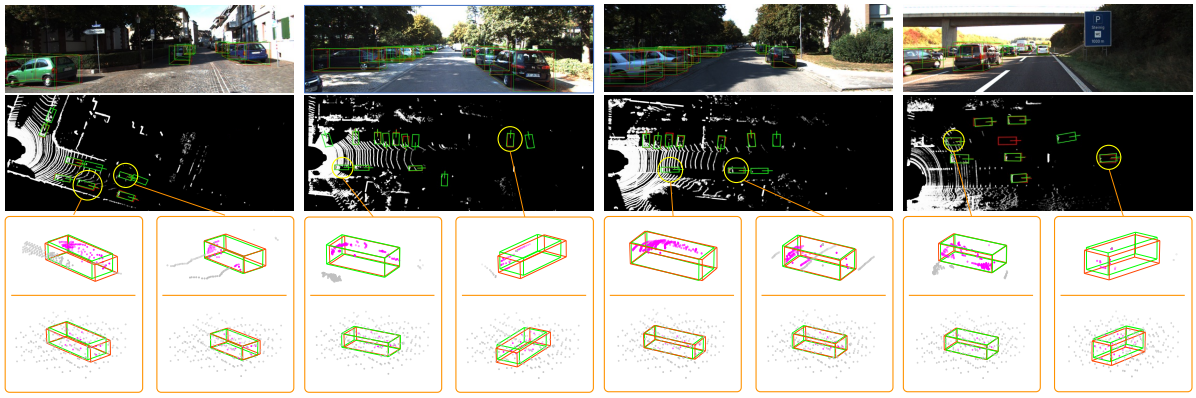


Figure 4. More visualization results on KITTI *val* set. (1) The first and second row: the ground-truth boxes and our predicted boxes are drew in red and green from both front-view and bird-eye-view. (2) The assigned implicit values of sampled raw points and virtual points are shown in the third and fourth row. The classified inside and outside points are represented in gray and pink respectively. Note that green and red boxes in the third and fourth rows mean the generated implicit boundaries and the ground-truth boxes, respectively.

potential objects. An intuition is that each object occupies only a small part of the whole space, and if a local space is centered on an object's center, fewer background distractions and receptive field biases will be introduced during learning. We argue that the proposed candidate shifting module is not our primary contribution but plays a key role in the pipeline. It is different from [6, 4] which shifts positive points to adjust their features and coordinates for better object detection.

To further evaluate the effectiveness of Eq. 2 in the candidate shifting module, (1) we visualize the *centerness* of foreground points of 100 frames randomly chosen from KITTI *val* set before and after shifting. Figure 3 shows that points after shifting are aggregated so that we can successfully build a series of 3D local spaces according to every clusters. (2) For a comparison, we train a same model with *hard score* (0/1) instead of *centerness*, yielding inferior AP_{3D}^{Mod} results (85.12% versus 85.38%) on KITTI *val* set. It clearly suggests that using *centerness* as confidence score is a good choice to measure the quality of the shifted centers for sampling.

D. Analyses of Training Configuration and Variance

We report the details of hyper-parameters setting in the main paper. Interestingly, even with small configurations, our proposed method can also converge efficiently. To verify this, two experiments are conducted by training with half number of virtual points and candidate centers. On KITTI *val* set, we achieve competitive results of 85.22% and 85.30% on Car@AP_{3D}^{Mod}, respectively. Note that both two parameters can significantly affect the computational power during training.

On the other hand, all experimental results in the main paper are achieved with the training seed of 888. To further evaluate the training stability of our proposed ImpDet, we run 5 trials on KITTI with different seeds. The *standard deviation* of performance is only 0.11%, which shows the reliability of our experimental analyses and conclusions.

E. More Visualizations of ImpDet

To better show the effectiveness of our proposed ImpDet framework, we visualize detection results on KITTI *val* split, as illustrated in Fig. 4. Specifically, for each block (left-top, right-top, left-bottom and right-bottom), we show detection results from front-view and bird-eye-view in the first and second rows. The implicit values of sampled raw points and virtual points are shown in the third and fourth row respectively. We especially choose two objects from different distance for better view. Here are several observations. (1) Our implicit boundary generation stage could produce high-quality boxes via the classified inside and outside sampled raw/virtual points. The size and orientation of boundaries are further be refined by occupant aggregation stage significantly. It strongly suggests the effectiveness of our proposed model and shows a novel perspective to generate the bounding box with implicit fields. (2) For close-distance objects, the sampled raw points cover most of the objects' surface. Thanks to well assigned implicit functions, we can easily generate boundary boxes according to the inside raw points. (3) For long-distance objects, the raw points are too sparse to produce a box even with 100% accurate classified points. However, our proposed virtual sampling strategy can effectively fill empty regions. And the learned semantic information can help it to perform implicit function on virtual points, so that we can fit a high-quality boundary according to both inside raw points and virtual points.

More visualizations of implicit fields are shown in Fig. 5. We can see that our implicit function can assign accurate values to inside raw points and virtual points, which are leveraged to generate more robust boundary boxes. The proposed occupant aggregation stage are applied to further refine the boundary box or discriminate it as false positive (*i.e.*, areas only with inside points but without bounding boxes). For some cases where objects have no points due to occlusion, our ImpDet fails on it. We explain that it is difficult to generate a candidate center over a large empty area, and the sampled virtual points cannot learn enough semantic features from their neighbor raw points.

Figure 5. More visualization results on KITTI *val* set. The ground-truth and our predicted boxes are drew in red and green. The assigned inside raw/virtual points are highlighted by purple. Best viewed in color and zoom in.

References

[1] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 1201–1209, 2021.

[2] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[3] Zhenwei Miao, Jikai Chen, Hongyu Pan, Ruiwen Zhang, Kaixuan Liu, Peihan Hao, Jun Zhu, Yang Wang, and Xin Zhan. Pvgnet: A bottom-up one-stage 3d object detector with integrated multi-level features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3279–3288, 2021.

[4] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9277–9286, 2019.

[5] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.

[6] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 11037–11045. IEEE, 2020.

[7] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.