ISE-RCNN: Image Semantics Enhancement Network for Robust 3D Object Detection

Xiaofeng Wang[1,2], **Yun Ye**[3], **Guan Huang**[3], **Xingang Wang**[1]

[1] Institute of Automation, Chinese Academy of Sciences {wangxiaofeng2020, xingang.wang}@ia.ac.cn
[2] School of Artificial Intelligence, University of Chinese Academy of Science
[3] PhiGent Robotics {yun.ye, guan.huang}@phigent.ai

Abstract

Recently, 3D object detectors combine LiDAR and camera information to generate precise detection results for autonomous driving. However, combining information from separate modalities to produce robust results is non-trivial. To address this problem, we propose ISE-RCNN, an *Image Semantics Enhancement*(ISE) network for robust 3D object detection. Specifically, to better exploit image semantic features, we project 3D proposals to 2D image plane, and then extract semantic embeddings leveraging CLIP, an image encoder trained on hundreds of million Internet images with the supervision of languages. The semantic embeddings are fused with LiDAR structure features by an attention module to refine the 3D boxes. Besides, we adopt database sampler augmentation consistently in point cloud and image domain, which helps our 3D detector learn rich structural and semantic accordance. Our method achieves new state-of-the-art performance on the widely used KITTI dataset(ISE-RCNN ranks 1st among previous published works on the pedestrian and cyclist leaderboards). And extensive experiment results show our detector captures robust and consistent semantic features in partial-occluded and noisy conditions. Code is at this repo.

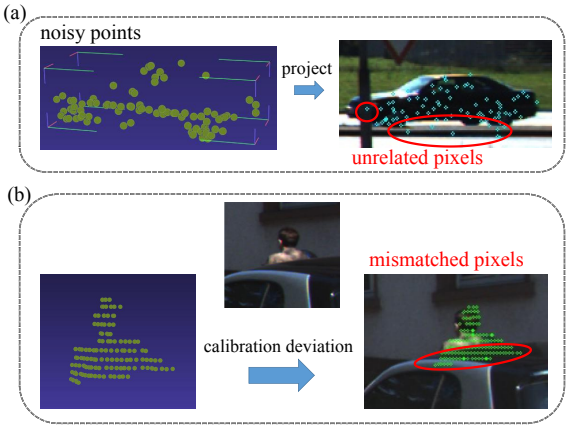

Figure 1: Noisy condition and occlusion condition that hinder point clouds properly align with image pixels. (a)When project noisy car points to the image plane, some points match with unrelated pixels from road and lamppost. (b)The pedestrian is occluded by car, and parts of point clouds are projected onto car surface due to calibration matrix deviation.

1 Introduction

3D object detection relying on LiDAR points has gained rapid progress in autonomous driving, this is due to in-depth mining structure information from points(Yang et al. 2019b; Shi, Wang, and Li 2019; Shi et al. 2020a; Yang et al. 2020) or voxels(Yan, Mao, and Li 2018; He et al. 2020; Liu et al. 2020; Zheng et al. 2021; Deng et al. 2021). However, LiDAR sensor is sensitive to the external environment and tends to make noise points. These noise points make extracting fine-grained structure information difficult, thus posing a latent safety risk to the autonomous driving system.

Intuitively, combining camera and LiDAR information can make more robust and precise detection results. Existing two modality methods (Liang et al. 2018, 2019; Yoo et al. 2020; Vora et al. 2020) use point-wise matching strategy, *e.g,* project point clouds to image plane and augment 3D points with image features such as pixel colors, 2D semantic segmentation scores or high-level semantic features. These methods outperform their LiDAR-only counterparts on the main benchmark dataset. Nevertheless, in these methods, 3D point clouds can be augmented with unrelated image information due to LiDAR sensor noises(Figure 1(a)). Additionally, when objects overlap with each other in camera view, 3D point clouds can be augmented with mismatched information from other objects as a result of calibration matrix deviation(Figure 1(b)). These common cases hinder two modality detectors from generating robust 3D boxes.

Aiming at generating robust and accurate detection results under noisy and occlusion conditions, we propose ISE-RCNN, an *Image Semantics Enhancement* network for robust 3D object detection. Our core idea is, compared with the point-wise matching strategy, the region-wise matching strategy captures more stable semantic features in noisy cases. Besides, an image encoder with enhanced semantic knowledge is required to extract consistent features even under object occlusion scenes(*i.e.,* object is at edge of image patch or is partially visible).

To this end, ISE-RCNN firstly generates coarse 3D bounding boxes using voxel-based 3D detector. These 3D boxes are projected into the image plane to obtain corresponding 2D boxes. Subsequently, we crop image patches

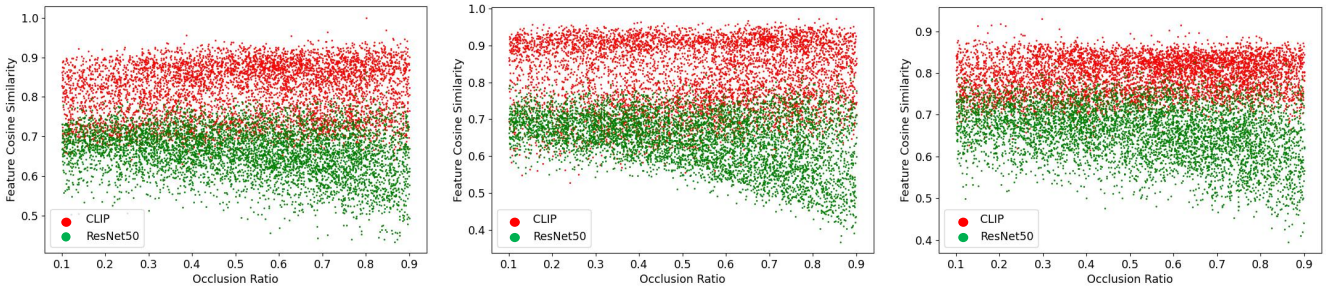

Figure 2: We crop image patches using ground truth boxes in KITTI dataset, then extract semantic embeddings using image encoders. Compared with ResNet50 backbone pre-trained on ImageNet, CLIP extracts consistent and stable features in diverse occlusion levels. The horizontal axis Occlusion Ratio is defined as the maximum intersection area over bounding box area. The vertical axis is defined as cosine similarity of feature embeddings. Figure(a)(b)(c) refer to car, pedestrian and cyclist respectively. More details about this figure is described in Section 4.2.

and feed them into CLIP (Radford et al. 2021) to get region-wise rich semantic information. CLIP is an image encoder which trained on hundreds of millions of image-text pairs scratched from the website. Due to the image diversity and rich semantics come from natural language descriptions, CLIP learns vital and consistent semantic information of various objects under numerous scenes. We find such an image encoder can extract accordant semantic embeddings in cases of object occlusion. As shown in Figure 2, CLIP extracts consistent and stable features compared with ResNet(He et al. 2016).

In the second stage, image semantic embeddings are fused with LiDAR structure information using an attention module to refine the 3D boxes. Furthermore, inspired by MoCa(Zhang, Wang, and Change Loy 2020), we extend the popular database sampling 3D data augmentation strategy to image plane, that is, randomly cut ground truth object points from training dataset and paste them into current scene. Correspondingly, object instances from the image database should be cropped and pasted into the image plane. Such data augmentation strengthens interlinks between two modalities.

Contributions of our work are summarized as follows: (i) Our designed *Image Semantics Enhancement* network extracts robust and consistent semantic information from candidate boxes. Semantic features serve as complementary to LiDAR structure information. Experiment results show our method alleviates the problem that 3D detector produce implausible results under noisy and occluded conditions. (ii) As far as we know, this is the first time CLIP(Radford et al. 2021) is used in the 3D detection domain. Experiments show that, compared with ImageNet(Deng et al. 2009) pre-trained convolution models, CLIP with richer semantic knowledge can boost 3D detector performance. (iii) Without bells and whistles, our proposed ISE-RCNN achieves state-of-the-art performance on the popular KITTI dataset. Particularly, on the pedestrian and cyclist leaderboards, our method ranks 1st among previous published works.

2 Related Work

There are generally two types of 3D detectors: (i) One modality LiDAR-based 3D detectors capture fine-grained structure information from point clouds and produce accurate detection boxes. (ii) Two modality 3D detectors leverage camera as a complementary to LiDAR, generating more reliable results.

One modality methods. One modality methods extract structure features from either raw points or voxels. PointNet(Qi et al. 2017a) uses sampling and grouping operation alternatively to abstract point representations. PointNet++ (Qi et al. 2017b) extent the abstraction operation to learn local embeddings with different contextual scales. Based on these methods, PointRCNN(Shi, Wang, and Li 2019) introduces a 3D region proposal network, where proposal features are abstracted by point cloud RoI pooling. Then 3DSSD(Yang et al. 2020) proposes a one stage anchor-free framework for 3D detection, its point sampling operation considers both geometrical distance and feature distance. Different from these methods that directly extract information from raw points, VoxelNet(Zhou and Tuzel 2018) groups points into voxels and employs 3D CNN to aggregate spatial features. SECOND(Yan, Mao, and Li 2018) introduces sparse 3D convolution to speed up VoxelNet. PointPillars(Lang et al. 2019) further accelerate detection pipeline by dividing points into pillars. PV-RCNN(Shi et al. 2020a) designs voxel set abstraction to leverage both points position information and voxels spatial context. VoxelRCNN (Deng et al. 2021) states precise point position is not necessary, and it fully exploits voxel features to boosts 3D detection performance.

Two modality methods. Two modality methods augmented 3D points with image pixels, semantic scores or high-level features extracted by convolution blocks. MV3D(Chen et al. 2017) utilizes 2D convolution blocks to extract information from LiDAR Bird Eye View(BEV), Front View(FV) and image separately, then fuses these features within BEV proposal regions. AVOD(Ku et al. 2018) fuses multi-modality information before BEV pro-

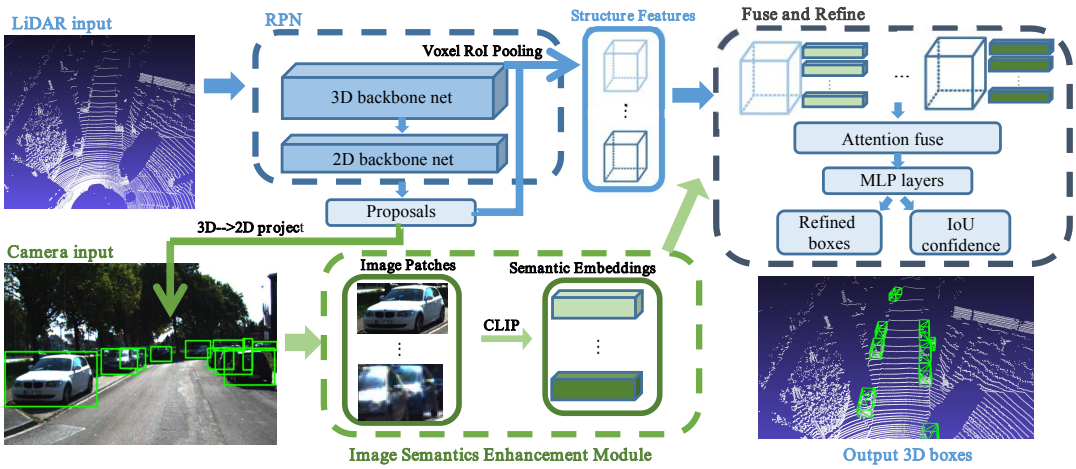

Figure 3: An overview of ISE-RCNN. The input point clouds are grouped into voxels, then we feed voxels into 3D backbone and 2D backbone in sequence to obtain coarse 3D box proposals. Subsequently, 3D proposals are used to focus on RoI of both voxel space and camera view. In voxel space, we employ voxel RoI Pooling to capture fine-grained structure features. In camera view, we project 3D proposals onto image plane and crop corresponding patches. These patches are extracted by CLIP to abstract semantic embeddings. Finally, an attention module combines structure with semantic information to get refined 3D boxes.

posal network, and this results in high recall proposals. ContFuse(Liang et al. 2018) performs multi-modality information fusion at diverse convolution map scales. MMF(Liang et al. 2019) uses multi-task(*e.g.*, 2D detection, depth completion, road estimation) to assist 3D detector. PointPainting(Vora et al. 2020) produces semantic segmentation scores per pixel by a pre-trained 2D segmentation network, then 3D points are projected onto image plane and augmented with semantic scores.

CLIP applications. CLIP(Radford et al. 2021) is an image encoder trained on hundreds of million image-text pairs, it improves generality under few-seen cases and a zero-shot CLIP outperforms ResNet50(He et al. 2016) on 16 datasets, including ImageNet(Deng et al. 2009). DietNeRF(Jain, Tancik, and Abbeel 2021) uses CLIP to format a semantic consistency loss under multiple observation poses, thus improves 3D reconstruction accuracy and even generates plausible completion results of unseen regions. This motivates us that CLIP extracts consistent semantic information under partially visible conditions.

3 ISE-RCNN Design

In this section, We detail the pipeline of ISE-RCNN, its framework is shown in Figure 3, which consists of: (i) a voxel-based Region Proposal Network(RPN) to produce coarse detection boxes, (ii) an auxiliary voxel RoI pooling layer to extract fine-grained structure information, (iii) an Image Semantics Enhancement(ISE) network to abstract image semantics, (iv) an attention module to fuse two modality embeddings and generate refined detection results. In the following, we discuss these modules in sequence.

3.1 Voxel-based Region Proposal Network

To exploit LiDAR points and generate detection proposals, we firstly group 3D point clouds into equally spaced voxels. Each voxel initials feature by averaging position and reflectance information of points within voxels. Then we employ 3D backbone network (Yan, Mao, and Li 2018; Shi et al. 2020a; Deng et al. 2021) to capture voxels information. The 3D backbone uses four cascaded sparse convolution blocks to reduce 3D feature size efficiently. Specifically, blocks$\{2,3,4\}$ each 2-times downsamples feature maps, thus the final 3D features are downsampled by 8-times. Subsequently, we concatenate features along z axis and produce a dense 2D BEV feature map. Next, a general 2D RPN(Yan, Mao, and Li 2018) is used to make 3D candidate boxes.

3.2 Voxel Structure Abstraction Branch

Successive downsampling captures high-level structure contexts. However, it blurs precise position information hidden in voxels. We adopt Voxel RoI pooling(Deng et al. 2021) to further extract fine-grained structure information in 3D voxel features. Voxel RoI pooling is an efficient operation for aggregate spatial context in voxels. Specifically, it divides 3D proposals into $6\times6\times6$ regular sub-voxel spaces, each center of sub-voxel groups neighbour voxel features using Mahalanobis distance.

3.3 Image Semantics Enhancement Module

One of the keys to exploiting image features in 3D detection is how to effectively abstract image information. In this subsection, we propose Image Semantics Enhancement module to address the question. Algorithm 1 outlines our procedures.

Here we give details of the ISE algorithm. In the RPN stage, we have homogeneous 3D detection proposals coor-

Algorithm 1: Image Semantics Enhancement

Input:

$P \in \mathbb{R}^{N,8,4}$ is homogeneous 3D proposals coordinates.

$T_{\text{calib}} \in \mathbb{R}^{3,4}$ is the calibration matrix from LiDAR to image plane.

$Q \in \mathbb{R}^{H,W,C}$ is the input image;

Output:

$Q_{\text{embedding}} \in \mathbb{R}^{N,512}$ is the abstracted semantic information of all proposals.

for $p \in P$ **do**
  $p_{\text{image}} = \text{PROJECT}(T_{\text{calib}}, p)$ $\quad p_{\text{image}} \in \mathbb{R}^{8,2}$
  $b_{\text{image}} = \text{BOUNDING}(p_{\text{image}})$ $\quad b_{\text{image}} \in \mathbb{R}^{4}$
  $q_{\text{patch}} = \text{CROP}(Q, b_{\text{image}})$ $\quad q_{\text{patch}} \in \mathbb{R}^{h,w,C}$
  $q_{\text{patch}} = \text{PROCESS}(q_{\text{patch}})$ $\quad q_{\text{patch}} \in \mathbb{R}^{224,224,C}$
  $q_{\text{embedding}} = \text{CLIP}(q_{\text{patch}})$ $\quad q_{\text{embedding}} \in \mathbb{R}^{512}$
end for

dinates $P_{ij} = (x_{ij}, y_{ij}, z_{ij}, 1)$, where xyz is point clouds position, $i \in [1, n]$ denotes index of proposals and $j \in [1, 8]$ denotes the 8 corners of each 3D proposal. These 3D coordinates are projected onto image plane coordinates p_{ij} by calibration matrix:

$$T_{\text{calib}} = T_{(\text{image} \leftarrow \text{camera})} T_{(\text{camera} \leftarrow \text{lidar})}. \tag{1}$$

For each 3D proposal(8 corners) in image plane, we calculate its minimum bounding rectangle(4 corners) as 2D RoIs. Then we crop the corresponding image patches and pre-process them(*e.g.,* resize and normalize). Subsequently, we extract semantic embeddings of these image patches using CLIP(Radford et al. 2021): a pre-trained vision transformer(Dosovitskiy et al. 2021). The output of each image patch is a 512 dimensions semantic embedding. It is worth noting that Algorithm 1 is made for illustration, the projection, bounding and CLIP can be operated in a parallel manner and speed up the pipeline.

3.4 Attention Fuse Module

Another key to using two modality methods is how to fuse LiDAR and image features effectively. As shown in Figure 1, we use voxel RoI pooling to extract LiDAR structure information within $6 \times 6 \times 6$ sub-voxels. These features are denoted as $\boldsymbol{F}^l \in \mathbb{R}^{216 \times C_l}$, where C_l is structure feature dimension. As for image context, we use ISE module to abstract semantics and obtain $\boldsymbol{F}^i \in \mathbb{R}^{C_i}$, similarly, $C_i = 512$ is semantic feature dimension. We then expand $\boldsymbol{F}^i$ and concatenate it with $\boldsymbol{F}^l$. The combined feature is $\boldsymbol{F}^\alpha \in \mathbb{R}^{216 \times (C_l + C_i)}$.

$\boldsymbol{F}^\alpha$ is produced by simple concatenate strategy. However, the concatenation ignores dependencies between two modalities, thus causing interference. As for a more effective fusing mechanism, we resort to perspective-channel attention(Yang et al. 2019a; Li et al. 2021). Specifically, it pools $\boldsymbol{F}^\alpha$ both in point axis(*i.e.,* the first channel) and feature axis(*i.e.,* the second channel) to get $\boldsymbol{F}^{p_1} \in \mathbb{R}^{1 \times (C_l + C_i)}$ and $\boldsymbol{F}^{p_2} \in \mathbb{R}^{216 \times 1}$. Then a weight matrix is generated by multiplication $\boldsymbol{F}^W = \boldsymbol{F}^{p_2} \boldsymbol{F}^{p_1}$. The final fused feature is $\boldsymbol{F}^\beta = \boldsymbol{F}^W \odot \boldsymbol{F}^\alpha$, where $\odot$ is element-wise multiplication. Finally, $\boldsymbol{F}^\beta$ is used by general detect head(Shi et al. 2020a; Deng et al. 2021) to obtain refined boxes.

3.5 Loss Function

Our ISE-RCNN is trained end to end. The loss design follows general two stage 3D detection network(Shi et al. 2020a; Deng et al. 2021), which comprises RPN part and refinement part:

$$\mathcal{L} = \mathcal{L}_{\text{RPN}} + \mathcal{L}_{\text{refine}}. \tag{2}$$

RPN Losses. RPN losses are decomposed into clssification loss and box regression loss. Specifically, we use Focal loss(Lin et al. 2017) for classification, use Smooth-L_1 loss(Liu et al. 2016) for box position and size regression. As for box direction classification, we resort to cross-entropy loss:

$$\begin{aligned} \mathcal{L}_{\text{RPN}} =& \mathcal{L}_{\text{cls}} + \mathbb{1}\left(\mathbf{c} \neq \mathbf{0}\right) \\ & (\mathcal{L}_{\text{box\_pos}} + \mathcal{L}_{\text{box\_size}} + \mathcal{L}_{\text{box\_dir}}), \end{aligned} \tag{3}$$

where $\mathbf{c}$ is the ground truth clssification label, $\mathbb{1}\left(\mathbf{c} \neq \mathbf{0}\right)$ indicates we only regress boxes within foreground region.

Refinement Loss. The second stage refinement loss consists of IoU confidence branch and box refinement branch. We assign IoU confidence to each unrefined box u_i as:

$$\mathcal{T}(t_i, u_i) = \begin{cases} 0 & \text{IoU}((t_i, u_i) < \theta_1 \\ \frac{\text{IoU}((t_i, u_i) - \theta_1}{\theta_2 - \theta_1} & \theta_1 \leq \text{IoU}((t_i, u_i) < \theta_2 \\ 1 & \text{IoU}((t_i, u_i)) \geq \theta_2 \end{cases} \tag{4}$$

where t_i is the ground truth box assigned to u_i, $\text{IoU}(\cdot)$ is the function calculating IoU between two boxes, θ_1 and θ_2 is the lower bound and upper bound threshold. We use binary cross entropy loss for IoU confidence branch. As for box refinement losses, we again split them into position, scale and direction refinement losses. Smooth-L_1 loss is for the former two tasks, and cross-entropy loss is for direction refinement:

$$\begin{aligned} \mathcal{L}_{\text{refine}} =& \mathcal{L}_{\text{IoU}} + \mathbb{1}\left(\mathcal{T} \geq \theta_{\text{reg}}\right) \\ & (\mathcal{L}_{\text{box\_pos}} + \mathcal{L}_{\text{box\_size}} + \mathcal{L}_{\text{box\_dir}}), \end{aligned} \tag{5}$$

where $\mathbb{1}\left(\mathcal{T} \geq \theta_{\text{reg}}\right)$ restricts that we only calculate box regression loss when IoU is greater than a threshold θ_{reg}.

4 Experiment

We evaluate our ISE-RCNN on the popular KITTI(Geiger et al. 2013) dataset, which comprises 7481 training samples and 7518 testing samples. Following the widely used *trainval* split, the training samples are further grouped into 3769 samples of *val* set and 3712 samples of *train* set. We conduct experiment on both *val* set and online test server. The evaluation metric used is Average Precision(AP) with a 3D overlap thereshold of $\{0.7, 0.5, 0.5\}$ for {car, pedestrian, cyclist}.

4.1 Implementation Details

Voxelization. The point clouds are grouped into equally spaced voxels as inputs. Considering KITTI provides only

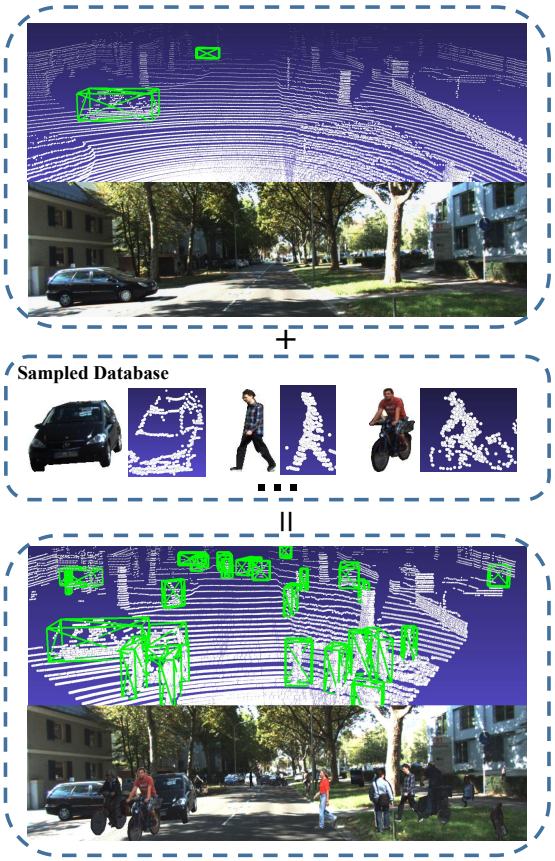

Figure 4: As shown in this figure, the original scene only has two objects and hinders network from learning rich context, so we cut image-point pairs from database and paste them onto road of the current scene. To establish such database, we sample object points from ground truth boxes and get the image instance masks from(Qi et al. 2019).

front-view images and annotations, we limit the point clouds range into $X \in [0, 70.4]\mathrm{m}$, $Y \in [-40, 40]\mathrm{m}$, and $Z \in [-3, 1]\mathrm{m}$. The voxel size for $\{X, Y, Z\}$ axises are $\{0.05, 0.05, 0.1\}\mathrm{m}$.

Network Architecture. Our voxel-based 3D backbone net and 2D region proposal network follow the framework in(Yan, Mao, and Li 2018; Shi et al. 2020a), The 3D backbone comprises four blocks with {16,32,48,64} channels respectively. Blocks{2,3,4} each 2-times downsamples feature map. As for the 2D RPN, there are two convolution blocks with {5,5} layers, and the second block 2-times downsamples the feature map. The voxel RoI pooling parameters follows (Deng et al. 2021).

Training. Our image encoder is set as ViT-B/32, its pretrained model is open-sourced by (Radford et al. 2021). The parameters of image encoder are frozen during training. Other parts of ISE-RCNN are trained with ADAM(Kingma and Ba 2014) optimizer, and learning rate initials as 0.01 with onecycle(Smith 2017) schedule. The IoU confidence thresholds θ_1, θ_2 are set as 0.25 and 0.75, and IoU regression threshold θ_{reg} is 0.55. We select 128 RoIs for refinement(64 samples are positive samples whose IoU$> \theta_{\text{reg}}$ with ground truth boxes).

Augmentation. As for the point clouds data augmentation:(i) We globally rotate the whole scene with an angle sampled from $[-\frac{\pi}{4}, \frac{\pi}{4}]$. (ii) We randomly(50% probability) flip the points along the X axis. (iii) We scale the points with a factor sampled from $[0.95, 1.05]$. As for the multi modality augmentation, we adjust database sampling augmentation(Yan, Mao, and Li 2018) for fair comparison with LiDAR-only methods. Motivated by (Zhang, Wang, and Loy 2021), we consistently cut point-image pairs from database and paste them into curent scene for training, which is illustrated in Figure 4. To address the occlusion problem(*i.e.,* when paste object instances into the image view, boxes may overlap with each other.)(Zhang, Wang, and Loy 2021), we remove such instances whose $\mathrm{OR} > \theta_{\text{block}}$ where the Occlusion Ratio(OR) is defined as the maximum intersection area over bounding box area, and θ_{block} is a mixed value shown in Table 5. The maximum sampled numbers of car, pedestrian and cyclist are empirically set as 12,6,6.

Inference. At the RPN stage, we use non-maximum suppression(NMS) with IoU threshold 0.7, and retrieve 100 proposals for refinement. At the refinement stage, we use NMS with IoU threshold 0.01 to get final detection results.

4.2 Experiment Results on KITTI Dataset

Comparison with State-of-the-Arts. We conduct experiments on both *test* and val set, and results are compared with several state-of-the-arts, including LiDAR-only methods and two-modality methods. As shown in table 1, on the car leaderboardour, ISE-RCNN attains 81.83% moderate AP, which outperforms previous two-modality methods. Particularly, compared with the generalized methods which provides three category results *e.g.,* PV-RCNN(Shi et al. 2020a). Our ISE-RCNN outperforms it by a large margin(0.4%, 2.37% and 5.47% on car, pedestrian and cyclist leaderboards.) It is worth noting that our method ranks 1st among all prior published works on pedestrian and cyclist leaderboards. To fairly compared with former methods, we use 11 recall AP metric for *val* set. Results are shown in Table 2, our ISE-RCNN outperforms recent state-of-the-arts VoxelRCNN(Deng et al. 2021) and 3D-CVF(Yoo et al. 2020) by 0.79% and 5.43% on car moderate AP. Besides, we measure the runtime of ISE-RCNN on a single NVIDIA TITAN RTX GPU and an Intel Xeon E5-2650 CPU, the results are shown in Table 3. ISE-RCNN runs at 4.8 FPS, and PV-RCNN(Shi et al. 2020a) runs at 7.09 FPS on the same platform. It means our method boost 3D detection results on three classes with approximately 2 FPS runtime cost.

Relieve Problem in Object Occlusion Condition. As discussed in Figure 1(b), in occlusion conditions, the LiDAR sensor can hardly capture complete structure information. Beyond that, two modality methods suffer from the problem that points may mismatch with other objects semantic features in image. We next verify that our ISE-RCNN can generate more robust 3D detection results in occlusion conditions. Beginning from the analysis of image semantic information, we use ground truth boxes to crop object image patches. Next, we extract semantic embeddings utiliz-

Method	Modality	Car - 3D Detection			Ped. - 3D Detection			Cyc. - 3D Detection		
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
SECOND(Yan, Mao, and Li 2018)	LiDAR	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90
PointPillars(Lang et al. 2019)	LiDAR	82.58	74.31	68.99	51.45	41.92	38.89	77.10	58.65	51.92
PointRCNN(Shi, Wang, and Li 2019)	LiDAR	86.96	75.64	70.70	47.98	39.37	36.01	74.96	58.81	52.53
STD(Yang et al. 2019b)	LiDAR	87.81	78.49	75.09	**53.29**	42.47	38.35	78.69	61.59	55.30
Part-A^2(Shi et al. 2020b)	LiDAR	84.65	75.96	73.51	53.10	43.35	40.06	79.17	63.52	56.93
PV-RCNN(Shi et al. 2020a)	LiDAR	90.25	81.43	76.82	52.17	43.29	40.29	78.60	63.71	57.65
CIA-SSD(Zheng et al. 2021)	LiDAR	89.59	80.28	72.87	-	-	-	-	-	-
Voxel-RCNN(Deng et al. 2021)	LiDAR	**90.90**	81.62	77.06	-	-	-	-	-	-
Generalized-SIENet(Li et al. 2021)	LiDAR	87.70	81.24	76.79	47.01	40.97	38.88	**83.00**	67.61	60.09
CT3D(Sheng et al. 2021)	LiDAR	87.83	81.77	77.16	-	-	-	-	-	-
SPG(Xu et al. 2021)	LiDAR	90.49	**82.13**	**78.88**	-	-	-	-	-	-
MV3D(Chen et al. 2017)	LiDAR+RGB	74.97	63.63	54.00	-	-	-	-	-	-
ContFuse(Liang et al. 2018)	LiDAR+RGB	83.68	68.78	61.67	-	-	-	-	-	-
AVOD-FPN(Ku et al. 2018)	LiDAR+RGB	83.07	71.76	65.73	50.46	42.27	39.04	63.76	50.55	44.93
F-PointNet(Qi et al. 2018)	LiDAR+RGB	82.19	69.79	60.59	50.53	42.15	38.08	72.27	56.12	49.01
F-ConvNet(Wang and Jia 2019)	LiDAR+RGB	87.36	76.39	66.69	52.16	43.38	38.80	81.98	65.07	56.54
UberATG-MMF(Liang et al. 2019)	LiDAR+RGB	88.40	77.43	70.22	-	-	-	-	-	-
3D-CVF(Yoo et al. 2020)	LiDAR+RGB	89.20	80.05	73.11	-	-	-	-	-	-
CLOCs(Yoo et al. 2020)	LiDAR+RGB	88.94	80.67	77.15	-	-	-	-	-	-
Painted PointRCNN(Vora et al. 2020)	LiDAR+RGB	82.11	71.70	67.08	50.32	40.97	37.87	77.63	63.78	55.89
ISE-RCNN(Ours)	LiDAR+RGB	89.12	81.83	77.29	51.44	**45.66**	**42.43**	82.62	**69.18**	**62.17**

Table 1: Comparison with state-of-the-art methods on KITTI *test* server for car, pedestrian and cyclist. The evaluation metric is the average precision(AP) of 40 sampling recall points.

Method	Modality	Car - 3D Detection		
		Easy	Mod.	Hard
SECOND(Yan, Mao, and Li 2018)	LiDAR	88.61	78.62	77.22
PointRCNN(Shi, Wang, and Li 2019)	LiDAR	88.88	78.63	77.38
PointPillars(Lang et al. 2019)	LiDAR	86.72	76.06	68.91
PV-RCNN(Shi et al. 2020a)	LiDAR	89.35	83.69	78.70
3D-SSD(Yang et al. 2020)	LiDAR	89.71	79.45	78.67
SA-SSD(He et al. 2020)	LiDAR	**90.15**	79.91	78.78
CIA-SSD(Zheng et al. 2021)	LiDAR	90.04	79.81	78.80
Voxel-RCNN(Deng et al. 2021)	LiDAR	89.41	84.52	78.93
MV3D(Chen et al. 2017)	LiDAR+RGB	71.29	62.68	56.56
ContFuse(Liang et al. 2018)	LiDAR+RGB	82.54	66.22	64.04
3D-CVF(Yoo et al. 2020)	LiDAR+RGB	89.67	79.88	78.47
Painted PointRCNN(Vora et al. 2020)	LiDAR+RGB	88.38	77.74	76.76
ISE-RCNN(Ours)	LiDAR+RGB	89.60	**85.31**	**79.04**

Table 2: Comparison with state-of-the-art methods on KITTI *val* set for car. The evaluation metric is the average precision(AP) of 11 sampling recall points for a fair comparison with previous methods(KITTI suggests using recall 40 AP on 08.10.2019.).

Car-3D AP			Car.-BEV AP			FPS(Hz)
Easy	Mod.	Hard	Easy	Mod.	Hard	
92.64	85.63	83.29	95.43	91.46	89.29	4.80

Table 3: ISE-RCNN performance of car with 40 recall points.

ing CLIP and pre-trained ResNet50 backbone respectively. For ResNet50, we delete *fc* layers and extract embeddings from the last pooling layer. Then we calculate cosine similarities between these features(*e.g.,* for CLIP, we firstly abstract the unoccluded object embedding as a *key* sample, then we calculate cosine similarities between other CLIP embeddings and *key* embedding. The same for ResNet50.) As shown in Figure 2, CLIP semantic features display higher consistency than ResNet50 features. Besides, when object occlusion ratio increases, ResNet feature consistency decrease, whereas CLIP features maintain consistency. This observation motivates us that CLIP can boost 3D detection results in occlusion cases. We verify it by evaluating our methods on occluded objects. According to the occlusion-level label provided by KITTI(0:*fully-visible*, 1:*partially-occluded*, 2:*difficult-to-see*), we evaluate our method on *partially-occluded* objects and *difficult-to-see* objects, which follows KITTI moderate and hard label. As shown in Table 4, we integrate ISE module with two competitive baselines: PV-RCNN(Shi et al. 2020a) and VoxelRCNN(Deng et al. 2021). Results show that our CLIP-ISE module comprehensively improves 3D detection performance in occlusion conditions. It is worth noting that our CLIP-ISE module increases AP of *partially-visble* pedestrian by 6%. We argue this is because pedestrian point clouds have a small volume, and their structure information is damaged by occlusion. Although LiDAR-based models capture little information of occluded pedestrians, our CLIP-ISE module captures stable and consistent semantic features and helps to generate robust results. We further replace CLIP-ISE with Res-ISE(with ImageNet pre-trained ResNet50), and the performance dramatically drops, which further proves our design is effective.

Besides, we find that our ISE-RCNN learn rich semantic information from occluded objects in the training phase. As shown in Table 5, we alter the Occlusion Ratio(OR) of multi-modality database sampling augmentation from 0.3 to 0.7, and ISE-RCNN performance increases continually. In contrast, the Res-ISE-RCNN show worse perfor-

Method	Car - 3D		Ped. - 3D		Cyc. - 3D	
	Occ1.	Occ2.	Occ1.	Occ2.	Occ1.	Occ2.
PV-RCNN(Shi et al. 2020a)	84.66	82.65	58.32	53.78	72.35	67.67
PV-RCNN+Res-ISE	83.05	82.52	62.23	57.79	70.70	66.10
PV-RCNN+CLIP-ISE	85.18	83.02	65.11	59.74	73.08	68.52
Improvement to base.	*+0.52*	*+0.37*	*+6.39*	*+5.96*	*+0.73*	*+0.85*
VoxelRCNN(Deng et al. 2021)	85.05	82.73	60.07	55.12	72.43	68.07
VoxelRCNN+Res-ISE	85.04	82.83	65.48	60.170	73.24	68.68
VoxelRCNN+CLIP-ISE	85.11	82.83	66.46	61.63	74.90	70.45
Improvement to base.	*+0.06*	*+0.10*	*+6.39*	*+6.51*	*+2.47*	*+2.38*

Table 4: Our Image Semantic Enhancement(ISE) modules boost 3D detetcion results(recall 40 AP) in occlusion condition(*val* set). Occ1 denotes *partially-occluded* objects, and Occ2 denotes *difficult-to-see* objects. Res-ISE means we use ResNet50 as image encoder in ISE module, and CLIP-ISE means we use CLIP as image encoder.

mance at a high OR. We finally chose a mixed occlusion ratio(randomly sampled from$\{0.3,0.5,0.7\}$) as our setting.

Method	OR=0.3	OR=0.5	OR=0.7	Mixed
Res-ISE-RCNN	85.02	85.28	84.86	85.12
ISE-RCNN	85.08	85.43	85.51	85.63

Table 5: Hyperparameter analysis on Occlussion Ratio(OR) in multi-modality database sampling augmentation. OR is defined as the maximum intersection area over bounding box area. Res-ISE-RCNN means we replace CLIP with ImageNet pre-trained ResNet50. We report metric as Car 3D moderate recall 40 AP(*val* set).

Relieve Problem in Noise Points Condition. As discussed in Figure 1(a), noise point clouds match with unrelated background image information, which hinders detector from generating robust results. To verify our ISE-RCNN relieve this problem. We compare ISE module with competitive two-modality method PointPainting(Vora et al. 2020) on noise points condition. Specifically, in the validation phase, we add random noise to points within ground truth boxes. As shown in Table 6, combining PointPainting with VoxelRCNN generates worse detection results than baseline in the noisy conditions. Contrastly, our ISE module boosts baseline performance consisitenly in noisy cases. We argue that our method captures robust region-wise semantic features, which remedies 3D detetcor with noisy structure information.

Method	Noi.Ratio=0	Noi.Ratio=0.1	Noi.Ratio=0.2
VoxelRCNN(Deng et al. 2021)	85.26	80.77	43.48
Painted-VoxelRCNN	85.43	79.08	34.49
ISE-VoxelRCNN	85.60	81.91	51.41

Table 6: Our ISE module helps baseline generate more robust results than baseline in noisy conditions. Noi. Ratio=0.1 means we randomly shift the position of the points within a radius of 0.1m. We report metric as Car 3D moderate recall 40 AP(*val* set).

4.3 Ablation Study

In this subsection, we discuss how each component contributes to our final design. Our RCNN baseline is a two modality 3D detector which uses pre-trained ResNet50 as image encoder, and two modality faetures are combined by simple concatenation. The results are in Table 7.

Effects of Multi-modality Data Sampling. Following general 3D detector(Yan, Mao, and Li 2018; Shi et al. 2020a; Deng et al. 2021), points sampling augmentation greatly boosts detection performance. However, simply sampling in the points domain hinders two modality methods from learning rich semantic information in the image domain. Consisitently adding image sampling augmention increases 0.71 points of moderate AP.

Effects of Image Semantics Enhancement Module. As discussed in subsection 4.2, Our ISE module effectively help LiDAR-based detector generate more robust results in occlusion and noisy conditions. Replacing ResNet50 with CLIP improves 0.8 points of moderate AP.

Effects of Attention Fuse Module. It is not optimal to simply concatenate two modality embeddings. Using attention module to fuse image semantic features with points structure information attains 0.2 points on moderate AP.

RCNN Baseline	Img. Sampling	ISE Module	Att.-Fuse Module	Car 3D AP(recall 40) Easy	Mod.	Hard
✓				91.89	83.92	82.12
✓	✓			92.32	84.63	82.89
✓	✓	✓		92.47	85.43	83.11
✓	✓	✓	✓	**92.64**	**85.63**	**83.30**

Table 7: Ablation study on our ISE-RCNN design.

5 Conclusion

In this paper, we present ISE-RCNN, an Image Semantics Enhancement network for robust 3D detection. ISE-RCNN firstly generates coarse proposals from point clouds, and we project proposals to the image plane. Then our ISE module extracts semantic features from these image RoIs. Finally, an attention module fuses image semantic features and points structure information to make robust detection results. In the training phase, multi-modality database sampling augmention is applied to reinforce the interlinks between image and points. Our method outperforms all published works on KITTI pedestrian and cyclist leaderboards. Furthermore, extensive experiment results show that our method generate robust detection results under noisy and occluded condistions.

References

Chen, X.; Ma, H.; Wan, J.; Li, B.; and Xia, T. 2017. Multi-view 3d object detection network for autonomous driving. In *CVPR*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *CVPR*.

Deng, J.; Shi, S.; Li, P.; Zhou, W.; Zhang, Y.; and Li, H. 2021. Voxel R-CNN: Towards High Performance Voxel-based 3D Object Detection. In *AAAI*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*.

Geiger, A.; Lenz, P.; Stiller, C.; and Urtasun, R. 2013. Vision meets robotics: The KITTI dataset. *IJRR*.

He, C.; Zeng, H.; Huang, J.; Hua, X.-S.; and Zhang, L. 2020. Structure aware single-stage 3d object detection from point cloud. In *CVPR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Jain, A.; Tancik, M.; and Abbeel, P. 2021. Putting NeRF on a Diet: Semantically Consistent Few-Shot View Synthesis. In *ICCV*.

Kingma, D.; and Ba, J. 2014. Adam: A Method for Stochastic Optimization. *Computer Science*.

Ku, J.; Mozifian, M.; Lee, J.; Harakeh, A.; and Waslander, S. L. 2018. Joint 3d proposal generation and object detection from view aggregation. In *IROS*.

Lang, A. H.; Vora, S.; Caesar, H.; Zhou, L.; Yang, J.; and Beijbom, O. 2019. Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*.

Li, Z.; Yao, Y.; Quan, Z.; Yang, W.; and Xie, J. 2021. SIENet: Spatial Information Enhancement Network for 3D Object Detection from Point Cloud. In *CVPR*.

Liang, M.; Yang, B.; Chen, Y.; Hu, R.; and Urtasun, R. 2019. Multi-task multi-sensor fusion for 3d object detection. In *CVPR*.

Liang, M.; Yang, B.; Wang, S.; and Urtasun, R. 2018. Deep continuous fusion for multi-sensor 3d object detection. In *ECCV*.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *ICCV*.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *ECCV*.

Liu, Z.; Zhao, X.; Huang, T.; Hu, R.; Zhou, Y.; and Bai, X. 2020. Tanet: Robust 3d object detection from point clouds with triple attention. In *AAAI*.

Qi, C. R.; Liu, W.; Wu, C.; Su, H.; and Guibas, L. J. 2018. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*.

Qi, L.; Jiang, L.; Liu, S.; Shen, X.; and Jia, J. 2019. Amodal Instance Segmentation With KINS Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Sheng, H.; Cai, S.; Liu, Y.; Deng, B.; Huang, J.; Hua, X.-S.; and Zhao, M.-J. 2021. Improving 3D Object Detection with Channel-wise Transformer. In *ICCV*.

Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; and Li, H. 2020a. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *CVPR*.

Shi, S.; Wang, X.; and Li, H. 2019. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*.

Shi, S.; Wang, Z.; Shi, J.; Wang, X.; and Li, H. 2020b. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *TPAMI*.

Smith, L. 2017. Cyclical Learning Rates for Training Neural Networks. In *WACV*.

Vora, S.; Lang, A. H.; Helou, B.; and Beijbom, O. 2020. Pointpainting: Sequential fusion for 3d object detection. In *CVPR*.

Wang, Z.; and Jia, K. 2019. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. In *IROS*.

Xu, Q.; Zhou, Y.; Wang, W.; Qi, C. R.; and Anguelov, D. 2021. SPG: Unsupervised Domain Adaptation for 3D Object Detection via Semantic Point Generation. In *ICCV*.

Yan, Y.; Mao, Y.; and Li, B. 2018. Second: Sparsely embedded convolutional detection. *Sensors*.

Yang, B.; Wang, J.; Clark, R.; Hu, Q.; Wang, S.; Markham, A.; and Trigoni, N. 2019a. Learning Object Bounding Boxes for 3D Instance Segmentation on Point Clouds. In *NeurIPS*.

Yang, Z.; Sun, Y.; Liu, S.; and Jia, J. 2020. 3dssd: Point-based 3d single stage object detector. In *CVPR*.

Yang, Z.; Sun, Y.; Liu, S.; Shen, X.; and Jia, J. 2019b. Std: Sparse-to-dense 3d object detector for point cloud. In *ICCV*.

Yoo, J. H.; Kim, Y.; Kim, J.; and Choi, J. W. 2020. 3d-cvf: Generating joint camera and lidar features using cross-view spatial feature fusion for 3d object detection. In *ECCV*.

Zhang, W.; Wang, Z.; and Change Loy, C. 2020. Multi-Modality Cut and Paste for 3D Object Detection.

Zhang, W.; Wang, Z.; and Loy, C. C. 2021. Exploring Data Augmentation for Multi-Modality 3D Object Detection. In *CVPR*.

Zheng, W.; Tang, W.; Chen, S.; Jiang, L.; and Fu, C.-W. 2021. CIA-SSD: Confident IoU-Aware Single-Stage Object Detector From Point Cloud. In *AAAI*.

Zhou, Y.; and Tuzel, O. 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*.