SeSame: Simple, Easy 3D Object Detection with Point-Wise Semantics

Hayeon O[1][0009−0007−8197−8017], Chanuk Yang[2][0000−0002−2065−4031], and Kunsoo Huh[2][0000−0002−7179−7841]

[1] Department of Automotive Engineering (Automotive-Computer Convergence)
[2] Department of Automotive Engineering
{gkdus9595,ych901208,khuh2}@hanyang.ac.kr
Hanyang University, Seoul, 04763, Republic of Korea

Abstract. In autonomous driving, 3D object detection provides more precise information for downstream tasks, including path planning and motion estimation, compared to 2D object detection. In this paper, we propose SeSame: a method aimed at enhancing semantic information in existing LiDAR-only based 3D object detection. This addresses the limitation of existing 3D detectors, which primarily focus on object presence and classification, thus lacking in capturing relationships between elemental units that constitute the data, akin to semantic segmentation. Experiments demonstrate the effectiveness of our method with performance improvements on the KITTI object detection benchmark. Our code is available at https://github.com/HAMA-DL-dev/SeSame

Keywords: autonomous driving · LiDAR semantic segmentation · 3D object detection.

1 Introduction

In autonomous driving, the perception system is the initial step in identifying the surrounding environment for autonomous driving. Therefore, various tasks for recognizing driving situations using cameras and LiDAR have been proposed. Notably, 2D object detection based on image from camera has achieved accuracy surpassing human identification capabilities due to the rapid advancement of deep learning and ongoing research. However, it has limitations in conveying height and shape of objects. On the other hand, according to [6,7], 3D object detection provides a spatial representation, which can be expressed in a Bird's Eye View (BEV), where the BEV plane can serve as the global coordinate system for downstream tasks. For 3D object detection in autonomous driving, image-based methods estimate depth and generate pseudo-LiDAR point clouds[10]. Nevertheless, this approach has lower accuracy compared to LiDAR-only 3D object detection. To overcome this, other fusion methods [13,14,15,16,17,18,21] have been proposed. But the multi-modality method presents challenges in integrating and synchronizing data, which have different representations and are defined

in different coordinate systems. In contrast, LiDAR-only based 3D object detection has the advantage of fully leveraging the geometric information provided by point cloud. Furthermore, LiDAR semantic segmentation[30,33] has demonstrated the ability to extract semantic information from point cloud. This led to a question:

> "what if we can extract semantics from point cloud and leverage them in existing 3D detectors?"

Therefore, inspired by the question and [17], this paper proposes a method to leverage semantics from point-wise semantic segmentation for LiDAR-only 3D object detection, aiming to minimize loss and distortion on geometric information and preserve it. This method consists of three components, where point-wise semantic segmentation provides supplementary semantic features for 3D detector. In summary, the contributions of this paper are as follows:

1. This paper proposes a simple and easy method for 3D object detectors by leveraging segmented point cloud from point-wise semantic segmentation. To our best knowledge, it is the first to leverage semantic segmentation to object detection.
2. Unlike existing methods that rely on calibration, this approach does not require any calibration, thus minimizing loss and distortion in extracting and incorporating semantics.
3. The proposed method outperforms multi-modality approach for all classes, and the reference model and the baseline detector for car on KITTI object detection benchmark[37].

2 Related Works

2.1 LiDAR Semantic Segmentation

Semantic segmentation is essential for fully understanding the driving scenario and contextual information in autonomous driving. This can be categorized into 2D semantic segmentation at the pixel level and 3D semantic segmentation at the point level[8]. They involve multi-class classification on the pixels and points that constitute the image and point cloud, respectively. Originating from the PointNet series [28,29], methods have been proposed to extract semantics from the point cloud. Inspired by [23,24,25], SqueezeSeg[30] and PointSeg[31] use spherical projection to convert point cloud into range-view image, apply 2D semantic segmentation, and then map the result back to the point cloud. However, spherical projection has the drawback of causing loss or distortion of semantic information from the camera and geometric information from the point cloud. Z.Zhuang et al. [34] address this limitation with two-stream network and perspective projection. Nevertheless, similar to Fig. 1, this method is sensitive to the calibration, thus it can significantly affect the results[20]. On the other hand, inspired by [24], Y.Zhang et al.[32] propose BEV projection method to project point cloud onto

a 2D grid using polar coordinate. However, this method also cannot fully preserve geometric information because it projects 3D point clouds into 2D space. Cylinder3D[35], instead of other projection methods, converts point clouds defined in Cartesian coordinate into Cylindrical coordinate, considering the sparsity of point cloud. Its asymmetrical residual block considers the characteristic of many objects in autonomous driving scenes being cuboid-shaped. And the dimension-decomposition considers various contexts that point clouds represent in outdoor spaces with different densities. To reflect these high-rank contexts, they are divided according to context, enabling a LiDAR semantic segmentation that fully understands the contexts with low computation cost [36]. For this reason, we select Cylinder3D[35] to extract the semantics for 3D detectors.

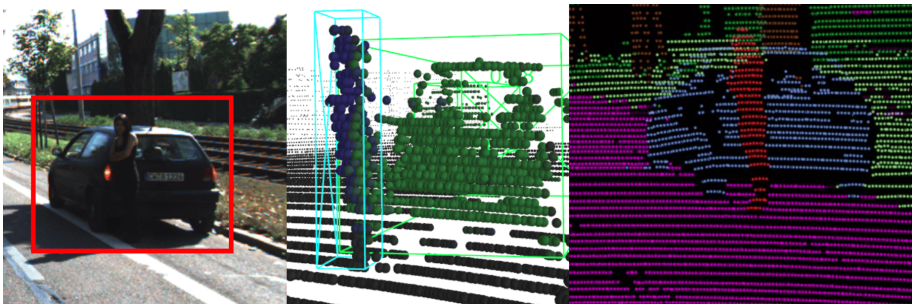

Fig. 1. (left) This depicts a scenario in which two objects, pedestrian and car, are overlapping, causing occlusion. **(center)** For the pixel-wise segmentation[26] projected onto the point cloud using perspective projection, miss-segmentation occurs, where some semantics of pedestrian (blue) are incorrectly classified as car (green). **(right)** On the other hand, it may be seen that point-wise semantic segmentation has higher accuracy.

2.2 3D Object Detection with Image

This can be divided into mono, stereo, and multi-view image methods depending on whether they use single or multiple images. Wang et al.[10] and D.Park et al.[11] proposes a method that extracts depth from an image and converts it into a point cloud through perspective projection, subsequently using it as input for existing 3D detectors. BEVFormer[12] extracts BEV features from multi-view images. Its spatial cross-attention between multi-camera views connects the same objects across different camera views and extract spatial information as a grid-shaped BEV query. However, due to the fundamental characteristics of image, inaccurate estimated depth can interfere with detection performance. Furthermore, this method can suffer from efficiency issues compared to LiDAR-based detection as concatenating multiple images increases the input dimension, leading to higher computational cost and lower performance.

Fig. 2. (left) Due to the error-sensitive hard association based on the LiDAR-camera calibration matrix, some points were not projected onto reflective surfaces. **(right)** Additionally, some image features did not correspond to the point cloud. it is stated that less than 5% of image features match with point clouds (for a 32-channel LiDAR scanner)[21]

.

2.3 3D Object Detection with Images and Point Cloud

Based on above, multi-modal 3D object detection has been proposed. In [17], pixel-wise semantic feature from 2D semantic segmentation is concatenated with raw point cloud and applied to 3D object detection, aiming to supplement semantic feature lacking in raw point cloud. Z.Liu et al.[21] proposed a method where both features are represented in a unified representation on BEV and performs both 3D object detection and BEV map segmentation based on fused features. AVOD[14] and MV3D[13] aggregate features extracted from images and point cloud to generate 3D proposals. CLOCs[16] aims to improve detection performance by fusing 2D bounding boxes obtained from 2D detector and 3D bounding boxes obtained from 3D detector. However, these multi-modal methods present additional challenges, as they require the consideration of various representations defined in different domains as a unified representation and demand synchronization for practical implementation. They also require a calibration matrix between LiDAR and camera for the unified representation. Even a slight error in this matrix can significantly impact performance as mentioned in [20]. As shown in Fig. 2, semantic loss may occur due to an insufficient number of point clouds corresponding to the pixels .

2.4 3D Object Detection with Point Cloud

PointRCNN [3] is a two-stage detector consisting of region proposal, classification, and box regression. It employs PointNet++ [29] as the backbone network to extract point-wise feature vectors through an encoder-decoder structure. These vectors are used for 3D proposal generation and foreground point segmentation, producing semantic features and foreground masks, respectively. After merging these with the raw point cloud, region pooling extracts semantic features and local spatial points. On the other hand, SECOND[4] is one-stage detector which

utilizes voxel grouping for point cloud on a 3D grid. Its sparse convolution considers only non-zero data and optimizes convolution operations through fast rule generation for repetitive patterns. This optimization significantly improves the speed of both training and inference. Similarly, PointPillars[5] encodes the point cloud into a pseudo-image by discretizing it into a 2D grid structure, paired with the corresponding feature map. It allows the network to leverage standard 2D convolutional layers, resulting in highly efficient computations. However, these methods primarily focus on detecting the presence of objects within the scene, which limits their ability to capture contextual relationships between points. Consequently, LiDAR-only based 3D object detection suffers from inconsistencies between the localization and classification of 3D bounding boxes [18].

3 Method

3.1 Extract Semantics from Point Cloud

The point-wise semantic segmentation used in this paper requires a point cloud as input and outputs per-point class labels. Unlike pixel-wise semantic segmentation, it offers distinct advantages. In pixel-wise semantic segmentation, semantic features extracted from images are fused with the point cloud via a camera-LiDAR calibration matrix. Any slight error in this matrix can lead to inaccurate projections of the 2D semantic features onto the point cloud. Moreover, in autonomous driving and robotics, cameras and LiDAR often operate at different frequencies, introducing latency issues. In contrast, point-wise semantic segmentation relies solely on the point cloud, eliminating the need for calibration and synchronization between data from different frames. For these reasons, we selected point-wise semantic segmentation to extract semantic information directly from the point cloud. Among the available models, we chose Cylinder3D[35] by Zhu Xinge et al., which utilizes cylindrical coordinates to preserve the geometric information of the point cloud while effectively handling its sparsity.

3.2 Aggregate the Semantics

Motivation. In the referenced paper[17], semantic features from stereo images were matched and then concatenated with the point cloud. These features were mapped to three classes at the pixel-wise level, derived from a 2D semantic segmentation[27]. However, this matching relied on perspective projection, which can result in geometric loss and distortion, as previously discussed. Therefore, we employ semantics directly extracted from the point cloud, thereby preserving the 3D geometric information without the projection.

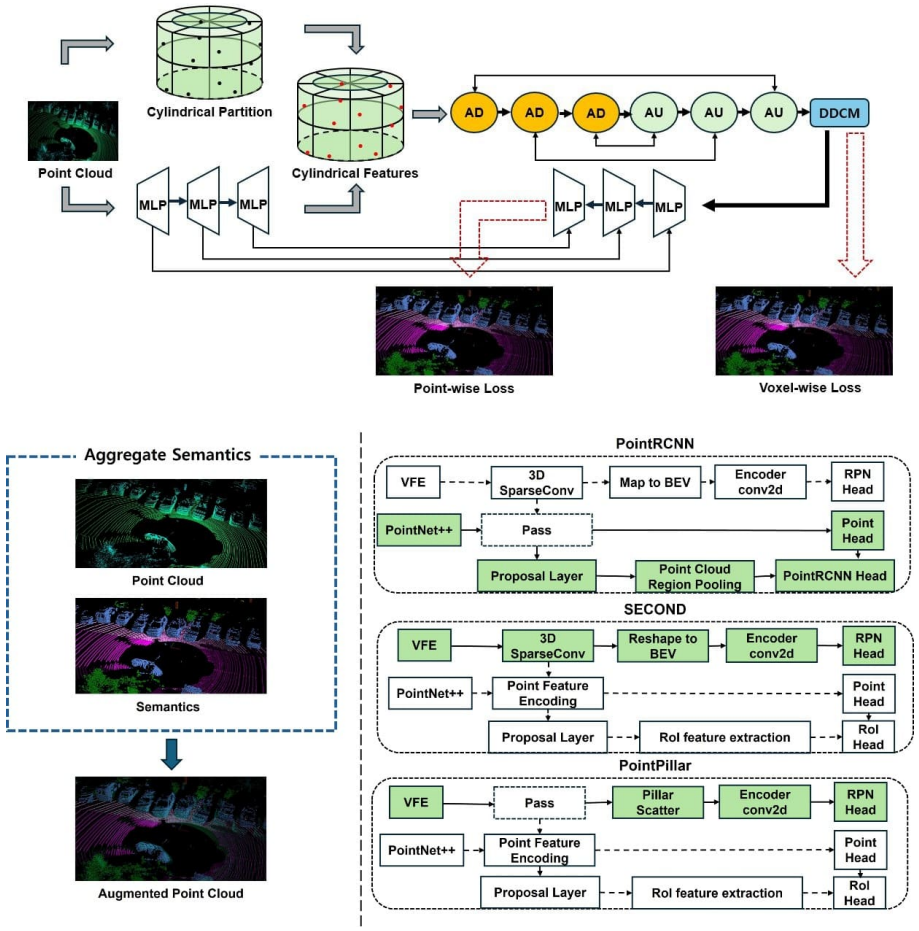

Fig. 3. (up) overall architecture of point-wise semantic segmentation[35] in this paper. It returns per-point semantics. **(down-left)** The input point clouds are augmented with the semantics. **(down-right)** 3D detectors with various feature extractors; point, voxel, pillar. OpenPCDet[40] supports various models within a single framework, with the green sections indicating the components used by each model.

Proposed Method. As shown in Algorithm. 1, the semantics from previous step include per-point class, indicating which class it belongs to. This corresponds to the semantic label map of the SemanticKITTI[38], which is the training dataset for the pretrained model from the previous step. Therefore, class mapping is required for the detector training on the KITTI 3D object detection dataset. Among the semantics, only the classes corresponding to the KITTI label map are assigned, followed by one-hot encoding. After aggregating with the raw

point cloud, a new point cloud is generated as follows : $[x_i, y_i, z_i, r_i, \text{sem.}]$, where i is index number, x, y, z are coordinates of a point in Cartesian coordinates, and r is intensity. Here, sem. represent which class it belongs to; unlabeled, car, pedestrian, and cyclist, respectively. The resulting point cloud with semantic annotations serves as the input for the feature extractor of each detector.

Algorithm 1 Pseudocode of Aggregate Semantics

Input:
map: returns index corresponding to the class
point cloud $(P) : \{p_i = (x_i, y_i, z_i, r_i) \mid i = 1, \ldots, n\} \in \mathbb{R}^{N \times 4}$ with N points
semantics: generated from LiDAR sem. seg. $\in \mathbb{R}^{N \times C}$ with C classes
Output:
augmented point cloud $(P_L) : \{p_i{}^{\mathrm{L}} = (x_i, y_i, z_i, r_i, sem.) \mid i = 1, \ldots, n\} \in \mathbb{R}^{N \times (C+4)}$

```
def augment_points (point_cloud, map, semantics) :
   label_one_hot = np żeros((points.shape[0], len(map)))
   for index, point in point_clouds:
     if semantics[i] in map:
       label_one_hot[index][map[cls]] = 1.0
   label_one_hot = torch.from_numpy(label_one_hot).float()
   augmented_point_cloud = np.concatenate((points,label_one_hot),axis=1)
   return augmented_point_cloud
```

3.3 LiDAR Detectors

To evaluate the proposed method, we selected existing 3D detectors that take point cloud as input. As mentioned in [17], These networks differ in their approaches to encoding the point cloud. [3] employs raw input points without the need for separate voxelization to extract point-wise features. In contrast, [4,5] voxelizes the input point cloud into voxels and pillars, respectively. Consequently, their 3D feature extraction backbone (Voxel Feature Encoding, VFE) generates 3D features, which are subsequently mapped to Bird's Eye View (BEV) and utilized as input for the 2D backbone. Finally, each network's head predicts bounding boxes based on the features extracted from their respective backbones.

3.4 Loss Functions

LiDAR Semantic Segmentation. For network optimization, weighted cross-entropy loss is employed for both voxel-wise and point-wise losses to maximize per-point accuracy and Intersection over Union (IoU) scores across all classes. Additionally, Lovasz-softmax loss [41] is utilized for point-wise loss to guide the training process. They share the same weight. So, the total loss is a summation of them.

$$L_{total} = L_{voxel} + L_{point} \tag{1}$$

3D object detection.

– **SeSame+point.** The loss function is defined in two stages:
In stage 1, focal loss addresses class-imbalance because background points, obtained through foreground segmentation, are more numerous than foreground points. For bin-based classification in 3D box generation, cross-entropy loss is used for the x and z axes and orientation, while Smooth L1 loss is used for regression based on residual values for the y axis and object dimensions (h, w, l).

$$L_{\text{reg}} = \text{smooth}_{L1} \sum_{v_i \in \{\Delta y, h, w, l\}} (v_i, v_i^*) \tag{2}$$

where Δy represents the residual values for the y axis, and h, w, l are the object dimensions. In stage 2, a positive label is assigned if the IoU is 0.55 or higher; otherwise, it is considered a negative label. Only proposals with positive labels are refined, and cross-entropy loss is calculated. So the total loss function is then a combination of these:

$$L_{\text{total}} = L_{\text{focal}} + L_{\text{cls}} + L_{\text{reg}} + L_{\text{refine}} \tag{3}$$

– **SeSame+voxel and SeSame+pillar.** Ground truth and anchor boxes are defined as $(x, y, z, w, h, l, \theta)$. Localization loss is computed using Smooth L1 loss to evaluate the residual between the two bounding boxes. The probability associated with each anchor is utilized as an argument in the focal loss, which is employed to calculate the classification loss. Furthermore, directional loss is incorporated into these two losses to derive the final loss.

$$L_{\text{total}} = L_{\text{loc}} + L_{\text{cls}} \tag{4}$$

4 Experiment

Training Details. SeSame+point, +voxel, and +pillar are trained with learning rates of 0.01, 0.003, and 0.003, and batch sizes of 8, 16, and 16, respectively. All three detectors share the same Adam as optimizer and OneCycleLR as scheduler[1], with a weight decay rate of 0.01 and momentum of 0.9, and are trained for 80 epochs with a single TITAN RTX GPU.

SemanticKITTI. Cylinder3D was trained on the SemanticKITTI dataset, which consists of point clouds classified into 28 classes. They are merged into a total of 19 classes by combining classes based on different moving statuses and ignoring classes with too few points. The label data is `uint32_t`, where the upper half represents the instance label and the lower half represents the semantic

label. In this paper, only the lower half containing semantic labels was used. The 19 classes were mapped to three classes defined in the KITTI object detection benchmark dataset: car, pedestrian, and cyclist.

KITTI 3D Object Detection Evaluation. The dataset contains 7481 training samples and 7518 testing samples, consisting of images and point clouds. All existing detectors[3,4,5] use the dataset, but differ in data split. In this paper, following general split method, the given training dataset was further split into 3712 training and 3769 validation samples for experiment. The train and val samples obtained through this split do not overlap[9]. Before the splitting, pretrained Cylinder3D performs inference on this dataset, and the resulting label mapped to the KITTI dataset is concatenated with training sample, thus forming the training dataset. The same approach was applied to the test split, and the results can be seen in Tab. 1 and 2. The evaluation metric is Average Precision (AP), with IoU thresholds of 0.7 for cars and 0.5 for pedestrians and cyclists, respectively. The three classes are divided into easy, moderate, and hard based on detection difficulty, where occlusion and truncation increase from the former to the latter.

Data Augmentation. Due to the limited number of training samples in the dataset, common data augmentation techniques applied in existing detectors are incorporated. Following the approach initiated from [2], a database is first created from the given training dataset. Random selection is made from this database for training, involving data augmentation such as random flipping along the x-axis, rotation around the z-axis and the origin within the range of $[-\pi/4, \pi/4]$, and random scaling within the range of $[0.95, 1.05]$. These augmentations are implemented for both the parameters of individual bounding boxes and the point sets within those boxes, with configurations tailored for each detector. To fairly evaluate the effectiveness of the proposed method, common settings are referenced and applied uniformly. If the 3D bounding boxes and point clouds generated by this technique result in physically implausible scenarios, such as collision due to overlapping bounding boxes, the original data is reverted.

4.1 Overall Results

Quantitative Results The results are based on the evaluation detection metrics which are: BEV, 3D as shown in Tab. 1 and 2. The modalities are LiDAR(L), and camera(C). The proposed method incorporating the existing detectors outperforms both the camera-only method and the multi-modality method for car. This result is evident in both metrics, which signifies the consistency of the results. For [5], increasing per box data augmentation leads to a further degradation in performance for pedestrians. This was mentioned in [5] and also observed in our result. Comparative analysis of the ablation studies based on modality, the implementation of the proposed method, and references to our approach is provided in Sec. 4.2.

Table 1. Results on the KITTI test 3D object detection benchmark

Method	modality	Car			Pedestrian			Cyclist		
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
DD3D[11]	C (mono)	23.22	16.34	14.20	13.91	9.30	8.05	2.39	1.52	1.31
Pseudo-LiDAR[10] + AVOD[14]	C (stereo)	55.4	37.2	31.4	N/A	N/A	N/A	N/A	N/A	N/A
MV3D[13]	L + C	71.09	62.35	55.12	N/A	N/A	N/A	N/A	N/A	N/A
AVOD-FPN[14]	L + C	73.59	65.78	58.38	38.28	31.51	26.98	60.11	44.90	38.80
PI-RCNN[19]	L + C	84.37	74.82	70.03	N/A	N/A	N/A	N/A	N/A	N/A
F-PointNet[15]	L + C	82.19	69.79	60.59	50.53	42.15	38.08	72.27	56.12	49.01
PointRCNN[3]	L (point)	**85.94**	75.76	68.32	49.43	41.78	38.63	73.93	**59.60**	**53.59**
SECOND[4]	L (voxel)	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90
PointPillars[5]	L (pillar)	79.05	74.99	68.30	**52.08**	**43.53**	**41.49**	**75.78**	59.07	52.92
SeSame + point **(Ours)**	L (point)	85.25	**76.83**	**71.60**	42.29	35.34	33.02	69.55	54.56	48.34
SeSame + voxel **(Ours)**	L (voxel)	81.51	75.05	70.53	46.53	37.37	33.56	70.97	54.36	48.66
SeSame + pillar **(Ours)**	L (pillar)	83.88	73.85	68.65	37.61	31.00	28.86	64.55	51.74	46.13

Table 2. Results on the KITTI test BEV detection benchmark

Method	modality	Car			Pedestrian			Cyclist		
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
DD3D [11]	C (mono)	30.98	22.56	20.03	15.90	10.85	8.05	3.20	1.99	1.79
Pseudo-LiDAR [10] + AVOD [14]	C(stereo)	66.8	47.2	40.3	N/A	N/A	N/A	N/A	N/A	N/A
MV3D [13]	L + C	86.02	76.90	68.49	N/A	N/A	N/A	N/A	N/A	N/A
AVOD-FPN [14]	L + C	88.53	83.79	77.90	**58.75**	**51.05**	**47.54**	68.06	57.48	50.77
PI-RCNN [19]	L + C	91.44	85.81	81.00	N/A	N/A	N/A	N/A	N/A	N/A
F-PointNet [15]	L + C	91.17	84.67	74.77	57.13	49.57	45.48	77.26	61.37	53.78
PointRCNN [3]	L(point)	**92.13**	87.39	82.72	54.77	46.13	42.84	**82.56**	**67.24**	**60.28**
SECOND [4]	L(voxel)	89.39	83.77	78.59	55.99	45.02	40.93	76.50	56.05	49.45
PointPillars [5]	L(pillar)	90.07	86.56	82.81	57.60	48.64	45.78	79.90	62.73	55.58
SeSame + point (Ours)	L(point)	90.84	**87.49**	**83.77**	48.25	41.22	39.18	75.73	61.70	55.27
SeSame + voxel (Ours)	L(voxel)	89.86	85.62	80.95	50.12	41.59	37.79	76.95	59.36	53.14
SeSame + pillar (Ours)	L(pillar)	90.61	86.88	81.93	44.21	37.31	35.17	72.22	60.21	53.67

Qualitative Results As shown in Fig. 4 and Fig. 5, SeSame+point detects cars more than the other approaches, which corresponded to the actual cars present in the scene. While SeSame+voxel recognizes pedestrian better than the other two approaches, this finding is consistent with the quantitative results presented in Tab. 1 and 2. On the other hand, SeSame+point and SeSame+pillar seem to have false positive and false negative for pedestrian each other. However, they also seem to detect other objects correctly, and especially +pillar detects far away car, which was confirmed to be actual objects upon inspection of the scene. Overall, SeSame+point demonstrates high accuracy in detecting cars, closely aligning with the ground truth, despite the occurrence of false negatives in sparse point regions. SeSame+voxel and +pillar effectively identifies objects even in sparse point cloud, with +voxel identifying cars missed by SeSame+point. These findings are consistent with the quantitative results in the paper, demonstrating the overall result is reliable.

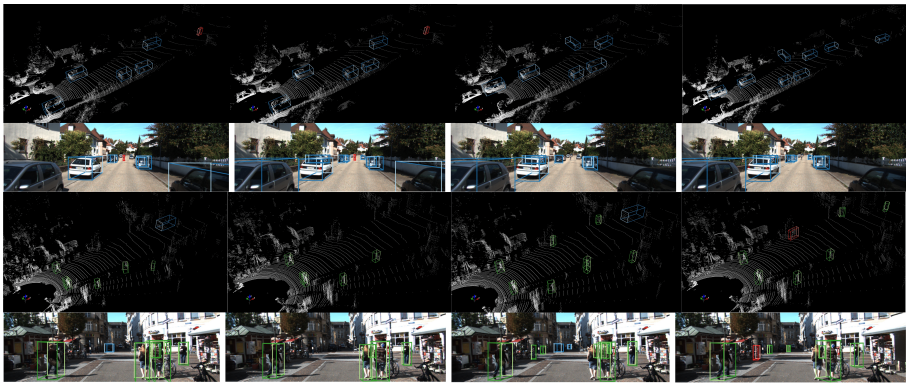

Fig. 4. The leftmost of the four sections represents the ground truth(GT), while the remaining three sections depict predictions from detectors varying with input features of point[3], voxel[4], and pillar[5]. The top figure illustrates a scenario with multiple cars**(blue)** and some cyclists**(red)**, while the bottom figure shows multiple pedestrians**(green)**.

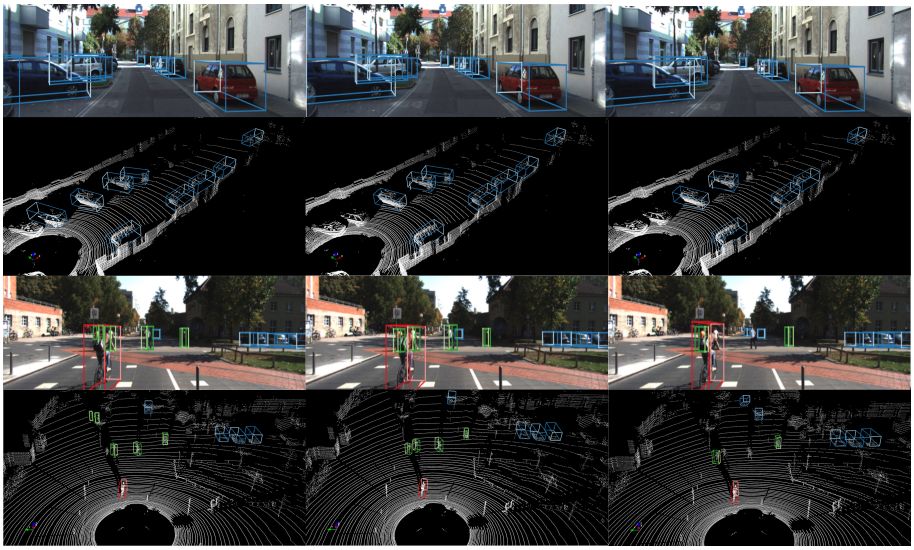

Fig. 5. Qualitative results on the KITTI test set. There are two scenes. For each scene, the results of SeSame+point, +voxel, and +pillar are shown from leftmost to rightmost.

4.2 Ablation Study

Encoding : Label and Score. The label obtained from point-wise semantic segmentation is mapped from SemanticKITTI to KITTI, assigning a score vector composed of 0 or 1 according to one-hot encoding to the point cloud. When comparing the aggregated point cloud with the point cloud passed through softmax for score mapping, the accuracy improved for the score mapping with val split. However, for test split, the accuracy was higher when mapped by one-hot encoding label. Furthermore, score mapping through softmax exhibits discrepancies in performance between the validation split and the test split. This implies that the training was within the range of noise for score mapping, and mapping by label is more reasonable. The corresponding results are included in Tab.1-4 in the Supplementary Material.

Modality : point cloud and multimodal As mentioned earlier in Sec. 2.2, the multimodal methods require calibration, which can distort or lose semantic information from images or geometric information from point clouds as illustrated in Fig. 2. Our method aims to address these challenges and demonstrates superior performance without calibration compared to multimodal methods as shown in Tab. 3. This indicates that the proposed method can be free from calibration-induced distortions and losses, while outperforming multimodal methods.

Table 3. Performance comparison of BEV and 3D object detection with reference method and ours for car on the test split. The IoU threshold is 0.7. The mAP (Mod.) column represents the mean Average Precision at the moderate difficulty level, providing a balanced assessment of model performance.

Method	Modality	mAP	AP_{3D}			AP_{BEV}		
		(Mod.)	Easy	Mod.	Hard	Easy	Mod.	Hard
MV3D[13]	L+C	69.63	71.09	62.35	55.12	86.02	76.90	68.49
AVOD-FPN[14]	L+C	74.86	73.59	65.78	58.38	88.53	83.79	77.90
PI-RCNN[19]	L+C	80.32	84.37	74.82	70.03	91.44	85.81	81.00
F-PointNet[15]	L+C	77.23	82.19	69.79	60.59	91.17	84.67	74.77
Painted PointRCNN[17]	L+C	79.91	82.11	71.70	67.08	**92.45**	**88.11**	83.36
SeSame+Point **(Ours)**	L (point)	**82.16**	**85.25**	**76.83**	**71.60**	90.84	87.49	**83.77**

Evaluation of the Method: Pre- and Post-Application Analysis. When comparing the detectors that employ our method to their baselines, there are performance improvements across all the detectors for car. Notably, SECOND exhibits the most substantial enhancement, as shown in Tab. 4. For SeSame+voxel, performance gains are observed not only for car but also for cyclist. But [5] did not show gains in identifying pedestrian and cyclist. We initially assumed that this was due to the conversion of 3D features into 2D features. However, this

hypothesis was rejected as the conversion implemented in SECOND[4] did not lead to any performance degradation. In conclusion, this can be attributed to the stochastic sampling in [5]. The sampling filter out or eliminate points containing semantic information, particularly when the ground truth object consists of only a few points. As a result, performance degradation may occur.

Table 4. We compared the performance of the detectors before and after the application of the proposed method using the KITTI object detection benchmark for car, with an Intersection over Union (IoU) threshold of 0.7. The detectors that employed our method exhibited performance improvements.

Method	Modality	mAP	$\mathbf{AP}_{3D}$			$\mathbf{AP}_{BEV}$		
		(Hard)	Easy	Mod.	Hard	Easy	Mod.	Hard.
PointRCNN [3]	L(point)	76.71	86.96	75.64	70.70	92.13	87.39	82.72
SeSame+Point **(Ours)**	L (point)	**77.69**	85.25	76.83	71.60	90.84	87.49	83.77
Improvement		0.98	-1.71	**1.19**	**0.9**	-1.29	**0.1**	**1.05**
SECOND [4]	L (voxel)	72.08	83.13	73.66	66.20	88.07	79.37	77.95
SeSame+Voxel **(Ours)**	L (voxel)	**75.74**	81.51	75.05	70.53	89.86	85.62	80.95
Improvement		3.66	-1.62	**1.39**	**4.33**	**1.79**	**6.25**	**3.00**
PointPillars [5]	L (pillar)	74.07	79.05	74.99	68.30	88.35	86.10	79.83
SeSame+Pillar **(Ours)**	L (pillar)	**75.29**	83.88	73.85	68.65	90.61	86.88	81.93
Improvement		1.22	**4.83**	-1.14	**0.35**	**2.26**	**0.78**	**2.10**

Comparison between Reference Method and Ours. There is only one result of [17] on test split, painted PointRCNN, so we compared it with SeSame+point as shown in Tab. 3. Our method demonstrates an improvement in performance for car. However, PointPainting[17] outperforms our method in detecting pedestrian and cyclist. We hypothesize that the results are attributed to the performance differences between pixel-wise and point-wise semantic segmentation across classes. [17] extracts semantic information using DeepLabv3+[27], trained on Cityscapes[39], achieving IoU of 87.95 and 78.88 for pedestrian and cyclist classes, respectively. In contrast, the point-wise semantic segmentation[35] used in this paper achieves IoU of 73.90 and 65.80 for pedestrian and cyclist, respectively. We can see Tab. 5 supports this hypothesis. This indicates that 3D object detection leveraging semantics is dependent on the performance of semantic segmentation. Therefore, while LiDAR-based semantic segmentation can provide reliable semantics for classes where it performs well, it may lead to performance degradation for classes where it does not. This leads us to focus on future work in Sec. 5.

Table 5. Performance comparison of 3D object detection with baseline, reference method, and ours on KITTI val split.

Method	Modality	Car			Pedestrian			Cyclist		
		Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
PointRCNN[3]	L(point)	86.75	76.05	74.30	63.29	58.32	51.59	83.68	66.67	61.92
Painted PointRCNN[17]	L+C	88.38	77.74	76.76	**69.38**	**61.67**	**54.58**	85.21	71.62	66.98
SeSame + point **(ours)**	L(point)	**88.76**	**78.35**	**77.43**	62.83	55.30	51.35	**87.80**	**72.74**	**67.00**
SECOND[4]	L(voxel)	86.85	76.64	74.41	67.79	59.84	52.38	84.92	64.89	58.59
Painted SECOND[17]	L+C	87.15	76.66	74.75	**68.57**	**60.93**	**54.01**	**85.61**	**66.44**	**64.15**
SeSame + voxel **(ours)**	L(voxel)	**88.35**	**78.55**	**77.27**	56.02	52.37	48.34	81.34	65.97	61.16
PointPillars[5]	L(pillar)	**87.22**	**76.95**	73.52	65.37	60.66	56.51	**82.29**	63.26	59.82
Painted PointPillars[17]	L+C	86.26	76.77	70.25	**71.50**	**66.15**	**61.03**	79.12	**64.18**	**60.79**
SeSame + pillar **(ours)**	L(pillar)	85.65	75.94	**73.85**	53.43	48.43	44.69	77.85	61.45	58.32

5 Conclusion

In this paper, we leveraged semantics from point-wise semantic segmentation for LiDAR-only 3D object detection and assessed its potential impact. The proposed approach was evaluated through performance improvements of the baseline model on the KITTI 3D object detection benchmark. Additionally, we demonstrated superior performance compared to multimodal methods that rely on calibration, which can lead to a loss of semantic information from images. Nevertheless, both our method and the reference model require annotated data for semantic segmentation, which incurs labeling costs. Furthermore, the performance of point-wise semantic segmentation remains insufficient for certain classes. In conclusion, our future research will focus on self-supervised multimodal semantic segmentation as a pretext task for 3D object detection, aiming to leverage both modalities without the calibration and data annotation.

References

1. Smith, L. A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay. (2018)
2. Zhou, Y. & Tuzel, O. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2018,6)
3. Shi, S., Wang, X. & Li, H. PointRCNN: 3D object proposal generation and detection from point cloud. *Proceedings Of The IEEE Computer Society Conference On Computer Vision And Pattern Recognition*. **2019-June** pp. 770-779 (2019)
4. Yan, Y., Mao, Y. & Li, B. SECOND : Sparsely embedded convolutional detection. *Sensors (Switzerland)*. **18**, 1-17 (2018)
5. Lang, A., Vora, S., Caesar, H., Zhou, L., Yang, J. & Beijbom, O. PointPillars: Fast encoders for object detection from point clouds. *Proceedings Of The IEEE Computer Society Conference On Computer Vision And Pattern Recognition*. **2019-June** pp. 12689-12697 (2019)
6. Singh, A. & Bankiti, V. Surround-View Vision-based 3D Detection for Autonomous Driving: A Survey. (2023), http://arxiv.org/abs/2302.06650

7. Ma, Y., Wang, T., Bai, X., Yang, H., Hou, Y., Wang, Y., Qiao, Y., Yang, R., Manocha, D. & Zhu, X. Vision-Centric BEV Perception: A Survey. (2022,8), http://arxiv.org/abs/2208.02797
8. Hao, S., Zhou, Y. & Guo, Y. A Brief Survey on Semantic Segmentation with Deep Learning. *Neurocomputing*. **406** pp. 302-321 (2020)
9. Chen, X., Kundu, K., Zhu, Y., Berneshawi, A., Ma, H., Fidler, S. & Urtasun, R. 3D Object Proposals for Accurate Object Class Detection. *Advances In Neural Information Processing Systems*. **28** (2015)
10. Wang, Y., Chao, W., Garg, D., Hariharan, B., Campbell, M. & Weinberger, K. Pseudo-lidar from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving. *Proceedings Of The IEEE Computer Society Conference On Computer Vision And Pattern Recognition*. **2019-June** pp. 8437-8445 (2019)
11. Park, D., Ambruş, R., Guizilini, V., Li, J. & Gaidon, A. DD3D : Is Pseudo-Lidar needed for Monocular 3D Object detection?. *Proceedings Of The IEEE International Conference On Computer Vision*. pp. 3122-3132 (2021)
12. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y. & Dai, J. BEVFormer: Learning Bird's-Eye-View Representation from Multi-camera Images via Spatiotemporal Transformers. *Computer Vision – ECCV 2022*. pp. 1-18 (2022)
13. Chen, X., Ma, H., Wan, J., Li, B. & Xia, T. Multi-View 3D Object Detection Network for Autonomous Driving. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2017,7)
14. Ku, J., Mozifian, M., Lee, J., Harakeh, A. & Waslander, S. Joint 3D Proposal Generation and Object Detection from View Aggregation. *2018 IEEE/RSJ International Conference On Intelligent Robots And Systems (IROS)*. pp. 1-8 (2018)
15. Qi, C., Liu, W., Wu, C., Su, H. & Guibas, L. Frustum PointNets for 3D Object Detection From RGB-D Data. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2018,6)
16. Pang, S., Morris, D. & Radha, H. CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection. *IEEE International Conference On Intelligent Robots And Systems*. pp. 10386-10393 (2020)
17. Vora, S., Lang, A., Helou, B. & Beijbom, O. PointPainting: Sequential fusion for 3D object detection. *Proceedings Of The IEEE Computer Society Conference On Computer Vision And Pattern Recognition*. pp. 4603-4611 (2020)
18. Huang, T., Liu, Z., Chen, X. & Bai, X. EPNet: Enhancing Point Features with Image Semantics for 3D Object Detection. *Computer Vision – ECCV 2020*. pp. 35-52 (2020)
19. Xie, L., Xiang, C., Yu, Z., Xu, G., Yang, Z., Cai, D. & He, X. PI-RCNN: An efficient multi-sensor 3D object detector with point-based attentive cont-conv fusion module. *Proceedings Of The AAAI Conference On Artificial Intelligence*. **34**, 12460-12467 (2020)
20. Bai, X., Hu, Z., Zhu, X., Huang, Q., Chen, Y., Fu, H. & Tai, C. TransFusion: Robust LiDAR-Camera Fusion for 3D Object Detection With Transformers. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition (CVPR)*. pp. 1090-1099 (2022,6)
21. Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D. & Han, S. BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird's-Eye View Representation. *2023 IEEE International Conference On Robotics And Automation (ICRA)*. pp. 2774-2781 (2023)
22. Lin, T., Dollar, P., Girshick, R., He, K., Hariharan, B. & Belongie, S. Feature Pyramid Networks for Object Detection. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2017,7)

23. Long, J., Shelhamer, E. & Darrell, T. Fully Convolutional Networks for Semantic Segmentation. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2015,6)
24. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing And Computer-Assisted Intervention – MICCAI 2015*. pp. 234-241 (2015)
25. Iandola, F., Han, S., Moskewicz, M., Ashraf, K., Dally, W. & Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. (2016)
26. Chen, L., Papandreou, G., Schroff, F. & Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. *CoRR*. **abs/1706.05587** (2017), http://arxiv.org/abs/1706.05587
27. Chen, L., Zhu, Y., Papandreou, G., Schroff, F. & Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. *Proceedings Of The European Conference On Computer Vision (ECCV)*. (2018,9)
28. Qi, C., Su, H., Mo, K. & Guibas, L. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2017,7)
29. Qi, C., Yi, L., Su, H. & Guibas, L. PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances In Neural Information Processing Systems*. **2017-Decem** pp. 5100-5109 (2017)
30. Wu, B., Wan, A., Yue, X. & Keutzer, K. SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud. *2018 IEEE International Conference On Robotics And Automation (ICRA)*. pp. 1887-1893 (2018)
31. Wang, Y., Shi, T., Yun, P., Tai, L. & Liu, M. PointSeg: Real-Time Semantic Segmentation Based on 3D LiDAR Point Cloud. (2018)
32. Zhang, Y., Zhou, Z., David, P., Yue, X., Xi, Z., Gong, B. & Foroosh, H. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition (CVPR)*. (2020,6)
33. Aksoy, E., Baci, S. & Cavdar, S. SalsaNet: Fast Road and Vehicle Segmentation in LiDAR Point Clouds for Autonomous Driving. *2020 IEEE Intelligent Vehicles Symposium (IV)*. pp. 926-932 (2020)
34. Zhuang, Z., Li, R., Jia, K., Wang, Q., Li, Y. & Tan, M. Perception-Aware Multi-Sensor Fusion for 3D LiDAR Semantic Segmentation. *Proceedings Of The IEEE/CVF International Conference On Computer Vision (ICCV)*. pp. 16280-16290 (2021,10)
35. Zhu, X., Zhou, H., Wang, T., Hong, F., Ma, Y., Li, W., Li, H. & Lin, D. Cylindrical and Asymmetrical 3D Convolution Networks for LiDAR Segmentation. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition (CVPR)*. pp. 9939-9948 (2021,6)
36. Chen, W., Zhu, X., Sun, R., He, J., Li, R., Shen, X. & Yu, B. Tensor Low-Rank Reconstruction for Semantic Segmentation. *Computer Vision – ECCV 2020*. pp. 52-69 (2020)
37. Geiger, A., Lenz, P. & Urtasun, R. Are we ready for autonomous driving? The KITTI vision benchmark suite. *2012 IEEE Conference On Computer Vision And Pattern Recognition*. pp. 3354-3361 (2012)
38. Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C. & Gall, J. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR

Sequences. *Proceedings Of The IEEE/CVF International Conference On Computer Vision (ICCV)*. (2019,10)
39. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S. & Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2016,6)
40. Team, O. OpenPCDet: An Open-source Toolbox for 3D Object Detection from Point Clouds. (https://github.com/open-mmlab/OpenPCDet,2020)
41. Berman, M., Triki, A. & Blaschko, M. The Lovász-Softmax Loss: A Tractable Surrogate for the Optimization of the Intersection-Over-Union Measure in Neural Networks. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition (CVPR)*. (2018,6)