Cluster-based Pseudo-labeling for Semi-Supervised LiDAR Semantic Segmentation

Supplementary Material

1. Social impact

Our research on semi-supervised LiDAR semantic segmentation addresses a crucial challenge in the field of computer vision, contributing to the broader goal of enhancing the efficiency and affordability of autonomous systems. By developing methods that allow LiDAR segmentation models to learn from limited labeled data alongside abundant unlabeled data, we pave the way for more cost-effective and sustainable deployment of LiDAR-based technologies.

The positive social impact of our work is two-fold. Firstly, our approach reduces the dependency on extensively annotated datasets, lowering the economic burden associated with manual data labeling. Secondly, traditional supervised methods often require substantial human resources and time to label vast datasets, contributing to increased energy consumption and carbon footprints. Our semi-supervised approach mitigates these environmental concerns, aligning with global efforts to develop sustainable and eco-friendly technologies.

2. Algorithm

We show the overall framework in Alg. 1. During training, a raw point cloud frame is first divided into ground clusters $\mathcal{G}$ and foreground clusters $\mathcal{F}$. Then we obtain point-level labels $\mathcal{Y}$ and cluster-level labels $\bar{\mathcal{Y}}$ for each cluster using ground truth through the label transfer strategy. Finally, we update ground net θ_g and foreground net θ_f using both cluster level loss $\mathcal{L}_{cluster}$ and point level loss $\mathcal{L}_{point}$. For pseudo-labeling process, we use θ_g and θ_f to generate point-level predictions $\mathbf{L}$ and cluster-level predictions $\bar{\mathbf{L}}$ for ground and foreground clusters, respectively. We use $\bar{\mathbf{L}}$ to generate category masks $\mathbf{M}$, which are combined with $\mathbf{L}$ to create the final pseudo labels. These pseudo labels can be directly used as ground truth for training the segmentation network.

Algorithm 1: Cluster-based Pseudo-labeling

Input: Labeled point cloud frames $\mathcal{X}^l = \{\mathcal{X}^l_i, \mathcal{Y}^l_i\}_{i=1}^{N_l}$; Unlabeled point cloud frames $\mathcal{X}^u = \{\mathcal{X}^u_i\}_{i=1}^{N_u}$; Classification confidence threshold τ; Ground net θ_g and foreground net θ_f; Max iteration I.

Output: Pseudo labels for unlabeled point cloud frames $\hat{\mathcal{Y}} = \{\hat{\mathcal{Y}}_i\}_{i=1}^{N_u}$.

```
// Cluster-based training.
for $iter = 1, 2, \cdots, I$ do
    for $i = 1, 2, \cdots, N_l$ do
        Generate $\mathcal{G}$ and $\mathcal{F}$ from $\mathcal{X}^l_i$;
        $\mathcal{Y}^{g/f}, \bar{\mathcal{Y}}^{g/f} \leftarrow LabelTransfer(\mathcal{G}, \mathcal{F}, \mathcal{Y}^l_i)$;
        $\mathbf{L}^g, \bar{\mathbf{L}}^g \leftarrow \theta_g(\mathcal{G})$;
        $\mathbf{L}^f, \bar{\mathbf{L}}^f \leftarrow \theta_f(\mathcal{F})$;
        Compute $\mathcal{L}^{g/f}_{point}(\mathbf{L}^{g/f}, \mathcal{Y}^{g/f})$ by Eq. (1);
        Compute $\mathcal{L}^{g/f}_{cluster}(\bar{\mathbf{L}}^{g/f}, \bar{\mathcal{Y}}^{g/f})$ by Eq. (2);
        $\mathcal{L}^{g/f} \leftarrow \mathcal{L}^{g/f}_{point} + \mathcal{L}^{g/f}_{cluster}$;
        Update model $\theta_{g/f}$ by loss $\mathcal{L}^{g/f}$;
    end
end
// Cluster-based pseudo-labeling.
for $i = 1, 2, \cdots, N_u$ do
    Generate $\mathcal{G}$ and $\mathcal{F}$ from $\mathcal{X}^u_i$;
    $\mathbf{L}^g, \bar{\mathbf{L}}^g \leftarrow \theta_g(\mathcal{G})$;
    $\mathbf{L}^f, \bar{\mathbf{L}}^f \leftarrow \theta_f(\mathcal{F})$;
    for $\mathbf{L}^{g/f}_k, \bar{\mathbf{L}}^{g/f}_k \in \mathbf{L}^{g/f}, \bar{\mathbf{L}}^{g/f}$ do
        Compute category mask $\mathbf{M}^{g/f}_k$ by Eq. (4);
        Compute pseudo labels $\hat{\mathcal{Y}_k}^{g/f}$ by Eq. (5);
    end
    $\hat{\mathcal{Y}}_i \leftarrow \{\hat{\mathcal{Y}}^g_1, ..., \hat{\mathcal{Y}}^g_s, \hat{\mathcal{Y}}^f_1, ..., \hat{\mathcal{Y}}^f_m\}$
end
```

3. Implementation Details

3.1. Datasets

SemanticKITTI. SemanticKITTI [1] stands as a comprehensive autonomous driving dataset derived from LiDAR acquisitions within the renowned KITTI Vision Odometry Benchmark. Captured in Karlsruhe, Germany, this dataset comprises LiDAR point clouds gathered by a 64-beam LiDAR sensor, featuring annotations across 19 semantic classes at the point level. It encompasses 22 LiDAR point cloud sequences, categorized into a train set (sequences 00 to 10, with sequence 08 allocated for validation) and a test set (sequences 11 to 21). Following [5, 9, 10, 13, 14], we exclusively utilize the training set for both training and validation, without employing the test set.

nuScenes. nuScenes [2] is another widely used large-scale LiDAR segmentation dataset in academia. It includes 1,000 driving scenes, each lasting 20 seconds, captured through a 32-beam LiDAR sensor in Boston and Singapore. It comprises a total of 40,000 LiDAR scans. Our approach adheres to the official train and validation sample splits, with the

Table A1. Category definitions of SemanticKITTI [1].

Role	car	bicy	moto	truck	bus	ped	b.cyc	m.cyc	road	park	walk	o.gro	build	fence	veg	trunk	terr	pole	sign
ground	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗	✗	✗	✗	✓	✗	✗
in ground bank	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗
foreground	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓	✓	✓	✗	✓	✓
in foreground bank	✗	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✓	✗	✓	✓

Table A2. Category definitions of nuScenes [2].

Role	barr	bicy	bus	car	const	moto	ped	cone	trail	truck	driv	othe	walk	terr	manm	veg
ground	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✓	✗	✗
in ground bank	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗
foreground	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓	✓
in foreground bank	✗	✓	✓	✗	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗

training set encompassing 28,130 scans and the validation set including 6,019 scans. We follow the official label mapping and use 16 semantic classes in our experiments.

ScribbleKITTI. ScribbleKITTI [12] is an extension of the popular SemanticKITTI [1] dataset. Instead of dense point-level annotations, it provides scribble-based annotations in the training set, which are sparse and less detailed. The other settings are the same as SemanticKITTI [1].

3.2. Training details

Model configuration. For our labeling network, we utilized a lightweight version of MinkNet [3]. In comparison to the original paper's network, Minknet18, which has layers $(2, 2, 2, 2, 2, 2, 2, 2)$, our network has fewer layers with $(2, 2, 2, 2, 1, 1, 1, 1)$. Additionally, we add a simple global average pooling layer and a linear layer for the classification task. All other configurations remain consistent with the original paper.

For our segmentation networks, we employ three common network architectures: BEV-based PolarNet [15], voxel-based MinkNet [3] and range-based FIDNet [16]. For PolarNet, we refer to the PolarNet [15] for more details on the model architecture and other related configurations. For MinkNet [3], we utilized the MinkNet18 as our segmentation network, maintaining the same configurations as detailed in the original paper. For FIDNet [16], to ensure a fair comparison, we directly implement the configuration following Lasermix [5] and IT2 [6].

Training configuration. All methods are implemented using PyTorch [8] on 6× NVIDIA GeForce RTX3090 GPUs with 24GB RAM. We utilize AdamW [7] as the optimizer and apply the OneCycle learning rate scheduler [11]. The labeling networks utilize a maximum learning rate of 0.1 and undergo a 50-epoch training process. We employ a batch size of 8 for SemanticKITTI [1] and 16 for nuScenes [2]. For all segmentation networks, the maximum learning rate is set to 0.01 and the training process spans 40 epochs. For MinkNet, the batch size is 4 for SemanticKITTI [1] and 8 for nuScenes [2]. For PolarNet and FIDNet, the batch size is 2 for SemanticKITTI [1] and 4 for nuScenes [2].

Category definition. Tab. A1 and Tab. A2 show the category definitions and the usage of each category in cluster bank of SemanticKITTI [1] and nuScenes [2], respectively.

Cluster bank details. During the training phase, for each frame of point cloud data, we first decompose it into foreground clusters and ground clusters (as described in § 3.2). Using the ground truth, clusters containing multiple categories are further decomposed into pure clusters. We maintain a first-in-first-out (FIFO) queue for each category listed in Tab. A1 and Tab. A2, and these pure clusters are stored in the corresponding queues. The maximum capacity of each queue is set to 1000. For each point cloud frame, we randomly select one cluster from each queue and combine these clusters with the foreground and ground clusters decomposed from the frame. These are then fed into the pseudo-labeling network for training. The cluster bank is not used during the inference phase.

Semi-supervised training framework. For each labeled data frame, we randomly select a frame of unlabeled data with pseudo-labels generated by `CLASS` and mix them using LaserMix [5]. We set $\alpha = 2, \phi = 1$ where α represents azimuth and ϕ represents the inclination direction. Please refer to Tab. 4 in LaserMix [5] paper for more details. On top of that, we add class-wise mixing. Specifically, we first extract points from the unlabeled data corresponding to the minority categories (as listed in Tab A1 and Tab. A2, line "in foreground bank") based on the pseudo-labels generated by `CLASS`. We then create two augmented copies of these extracted points by rotating them 120° and 240° along the z-axis, respectively. Finally, we concatenate these three sets of points with the point cloud generated by LaserMix [5]. The mixed data is used for training the downstream segmentation backbones.

Other configurations. We employ random jittering, scaling,

Table A3. Cluster accuracy and point IoU across SemanticKITTI and nuScenes datasets.

	SemanticKITTI	nuScenes
Foreground Cluster Acc	94.4	89.7
Ground Cluster Acc	88.9	85.8
Foreground Point IoU	86.7	83.1
Ground Point IoU	86.1	83.2

Table A4. Statistics of generated clusters on nuScenes `train`.

Statistics per cluster	Ground	Foreground	All
1 category	62.4%	85.3%	74.0%
2 categories	33.5%	14.1%	23.7 %
3 categories	4.1%	0.6%	2.3%
$>$ 3 categories	0.0%	0.0%	0.0%
average # of categories	1.4	1.2	1.3

flipping, and rotating as data augmentations for all raw point cloud frames in training phase. These exact data augmentation methods are also applied to clusters retrieved from the pure cluster bank. The minimum number of points per cluster is set to 50, and the confidence threshold τ for the classification task is set to 0.5.

4. Additional results.

Cluster statistics. We calculate the cluster accuracy and point IoU across SemanticKITTI [1] and nuScenes [2] in tab. A3. It can be found that clustering errors are rare in both datasets. We also present the statistics of generated clusters on nuScenes [2] in Tab. A4. Noticeably, clusters in nuScenes [2] exhibit higher purity compared to SemanticKITTI [1]. This disparity arises from nuScenes [2] employing a 32-beam LiDAR, whereas SemanticKITTI [1] uses a 64-beam LiDAR. Consequently, a single frame in nuScenes [2] contains significantly fewer points than in SemanticKITTI [1], resulting in sparser point distribution. This sparsity facilitates easier separation of different objects using HDBSCAN [4].

Analysis of classification threshold. Fig. A1 provides results with different classification threshold $\tau \in \{0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99\}$. We directly train the segmentation network using pseudo-labels generated by `CLASS` without employing the semi-supervised training framework. When τ is too low, classification predictions might erroneously output categories that do not exist within the cluster. This can lead to a reduction in the constraint imposed by the category mask on point-level predictions. When τ is too high, some challenging categories (*e.g.*, bicycle, motorcycle) might be incorrectly filtered out due to insufficient confidence. This can result in a decrease in the network's predictions for these categories. Given the minor changes in results around the 0.5 threshold, we opt for simplicity and set τ at 0.5 across all settings, despite achieving the best results with a threshold of 0.6.

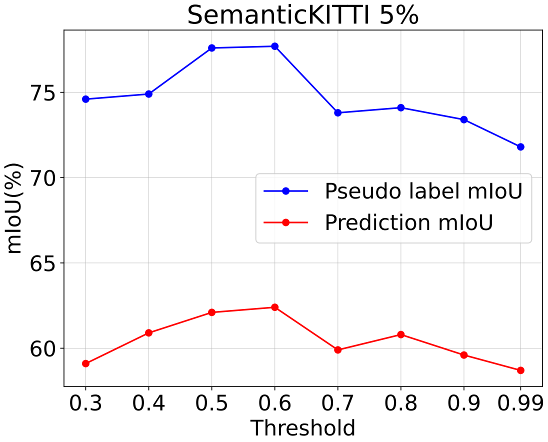

Figure A1. Analysis results on different confidence thresholds τ used in cluster-based pseudo-labeling process.

Table A5. Segmentation results of different pillar size on SemanticKITTI [1] `val` set using MinkNet [3] with 5% labeled frames. To save computational resources, we did not use $\theta_f \neq \theta_g$.

pillar size	3*3	4*4	5*5	6*6	7*7
mIoU	59.2	59.6	60.4	58.9	58.3

Analysis of pillar size. Tab. A5 provide the segmentation results of different pillar size on SemanticKITTI [1] `val` set using MinkNet [3] with 5% labeled frames. We directly train the segmentation network using pseudo-labels generated by `CLASS` without employing the semi-supervised training framework. To save computational resources and speed up training, we did not use $\theta_f \neq \theta_g$. When the pillar size is too small, some non-ground plane points may be incorrectly fitted(*e.g.*, some flat car roofs.), while when it's too big, some uneven ground points cannot be correctly fitted. These errors in plane fitting can both lead to a slight decline in model performance. Moreover, a small pillar size increases the number of ground clusters, thereby extending training and inference time.

Analysis of minimum cluster size. In §3.2, we filter out clusters with too few points(<50 points) since these clusters may disrupt the subsequent training. We analyze the effects of HDBSCAN's min cluster size on cluster purity and final performance in Tab. A6. Experimental setup is the same as Tab. A5. While smaller min cluster size can improve cluster purity at the risk of over-segmentation, and larger sizes degrade purity, `CLASS` shows consistent performance across this spectrum. This robustness confirms its insensitivity to hyperparameter variations and effectiveness in handling impure clusters.

Table A6. Cluster purity and segmentation results of different min cluster size on SemanticKITTI [1] `val` set using MinkNet [3] with 5% labeled frames. To save computational resources, we did not use $\theta_f \neq \theta_g$.

min cluster size	10	30	50	70	90
avg. # of classes	1.47	1.54	1.60	1.66	1.79
mIoU	58.6	59.9	60.4	60.7	60.1

Table A7. Inference latency and parameter count for different components of CLASS.

component	inference time (ms)	parameters (MB)
Plane fitting	95.8	0
HDBSCAN	127.9	0
Total	223.7	0
Cluster-based Pseudo-labeling	40.6	16.3

Latency and parameters of CLASS. Tab. A7 shows the average inference latency per frame and parameter count for different components of CLASS on SemanticKITTI [1] with a single NVIDIA GeForce RTX 3090. CLASS has a lower latency in pseudo-labeling stage due to the small network parameter size. The cluster generation stage is performed on CPU, resulting in longer latency. We will consider employing more advanced cluster generation techniques to optimize the efficiency in the future.

More illustration of cluster bank. Fig. A2 shows the pseudo labels predicted by the networks trained with/without the cluster bank. It can be observed that after the introduction of the cluster bank, the quality of pseudo labels for tail classes such as bicycles and motorcycles is significantly improved. This demonstrates the effectiveness of the cluster bank in alleviating the long-tail distribution issue in point cloud data.

Qualitative results. Fig. A3 visualizes the cluster generation results on the *train* set of SemanticKITTI [1]. It further confirms the purity of the generated clusters. Fig. A4 visualizes pseudo labels produced by networks trained with different CLASS components, each column corresponds to a row in Table. 5 of the ablation studies. These pseudo labels are showcased on the *train* set of SemanticKITTI [1], providing further validation for the efficacy of each component in our approach.

5. Public Resources Used

We acknowledge the use of the following public resources, during the course of this work:

- SemanticKITTI[1] .... CC BY-NC-SA 4.0
- SemanticKITTI-API[2] .... MIT License

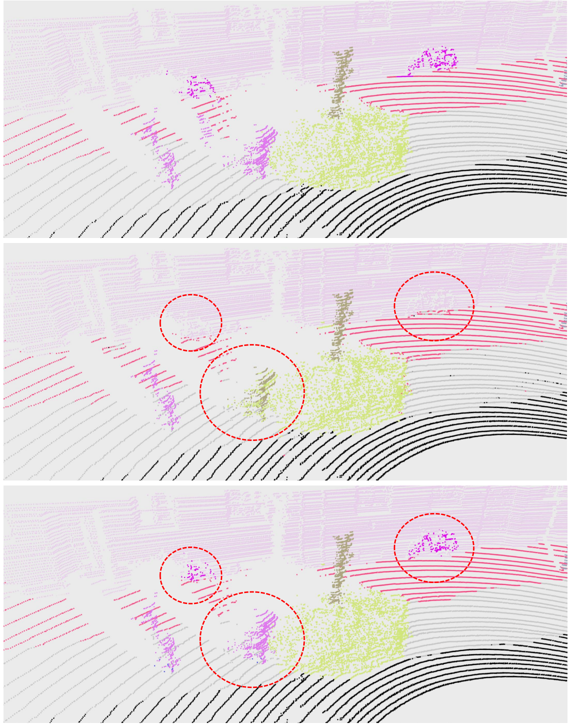

Figure A2. Role of cluster bank: **Top:** Ground truth; **Middle:** Pseudo labels produced by networks trained without cluster bank; **Bottom:** Pseudo labels produced by networks trained with cluster bank.

- nuScenes[3] .... CC BY-NC-SA 4.0
- nuScenes-devkit[4] .... Apache License 2.0
- TorchSparse[5] .... MIT License
- Minkowski Engine[6] .... MIT License
- PolarNet[7] .... BSD 3-Clause License
- FIDNet[8] .... Unknown
- LaserMix[9] .... CC BY-NC-SA 4.0
- OpenPCSeg[10] .... Apache License 2.0
- OpenPCDet[11] .... Apache License 2.0

[1] http://semantic-kitti.org.

[2] https://github.com/PRBonn/semantic-kitti-api.

[3] https://www.nuscenes.org/nuscenes.

[4] https://github.com/nutonomy/nuscenes-devkit.

[5] https://github.com/mit-han-lab/torchsparse.

[6] https://github.com/NVIDIA/MinkowskiEngine.

[7] https://github.com/edwardzhou130/PolarSeg.

[8] https://github.com/placeforyiming/IROS21-FIDNet-SemanticKITTI.

[9] https://github.com/ldkong1205/LaserMix.

[10] https://github.com/PJLab-ADG/OpenPCSeg.

[11] https://github.com/open-mmlab/OpenPCDet.

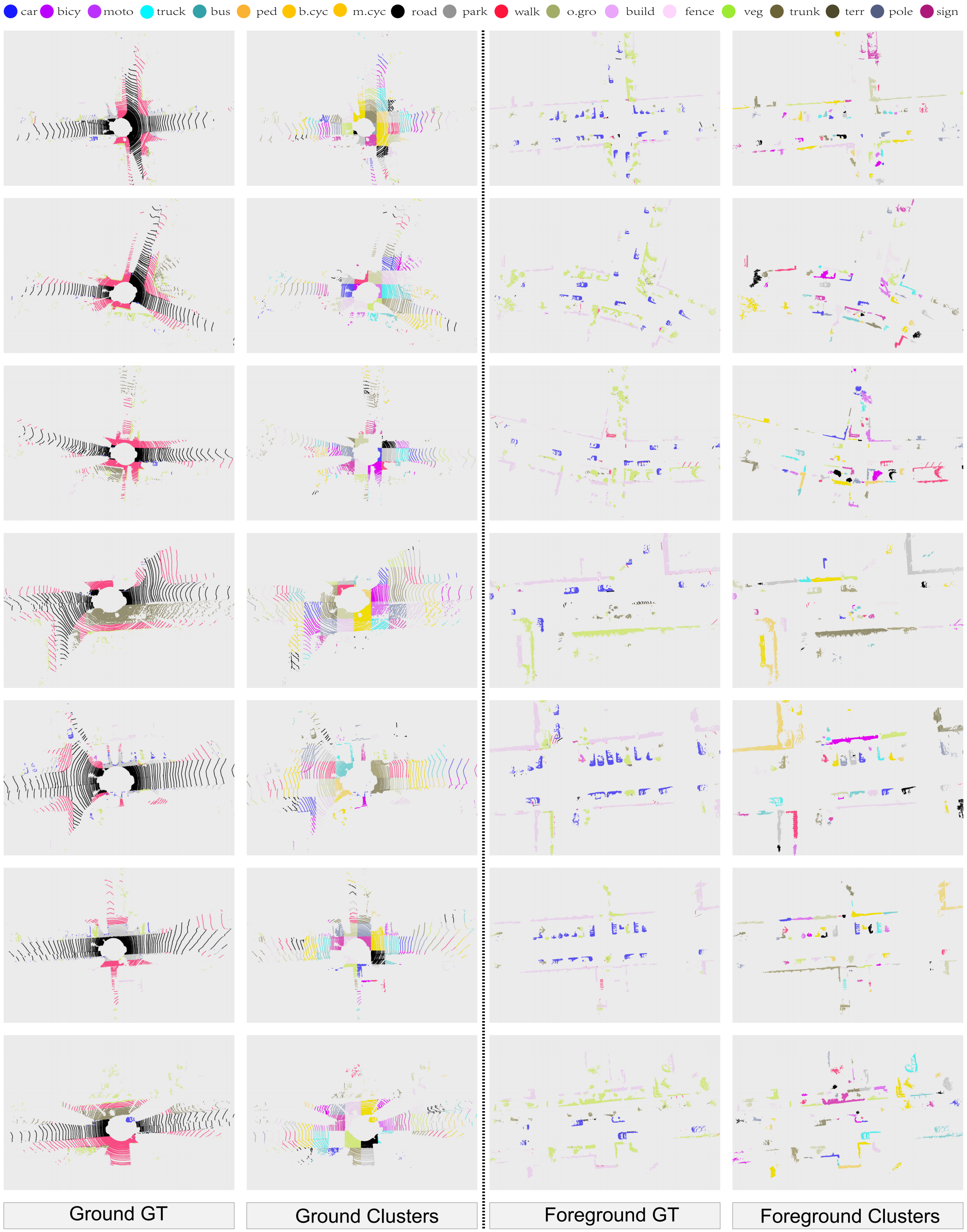

Figure A3. **Visualization of ground and foreground clusters.** The colors in ground and foreground GT represent different semantic categories, while the colors in ground and foreground clusters are used to distinguish different clusters but don't represent category information. Best viewed in color.

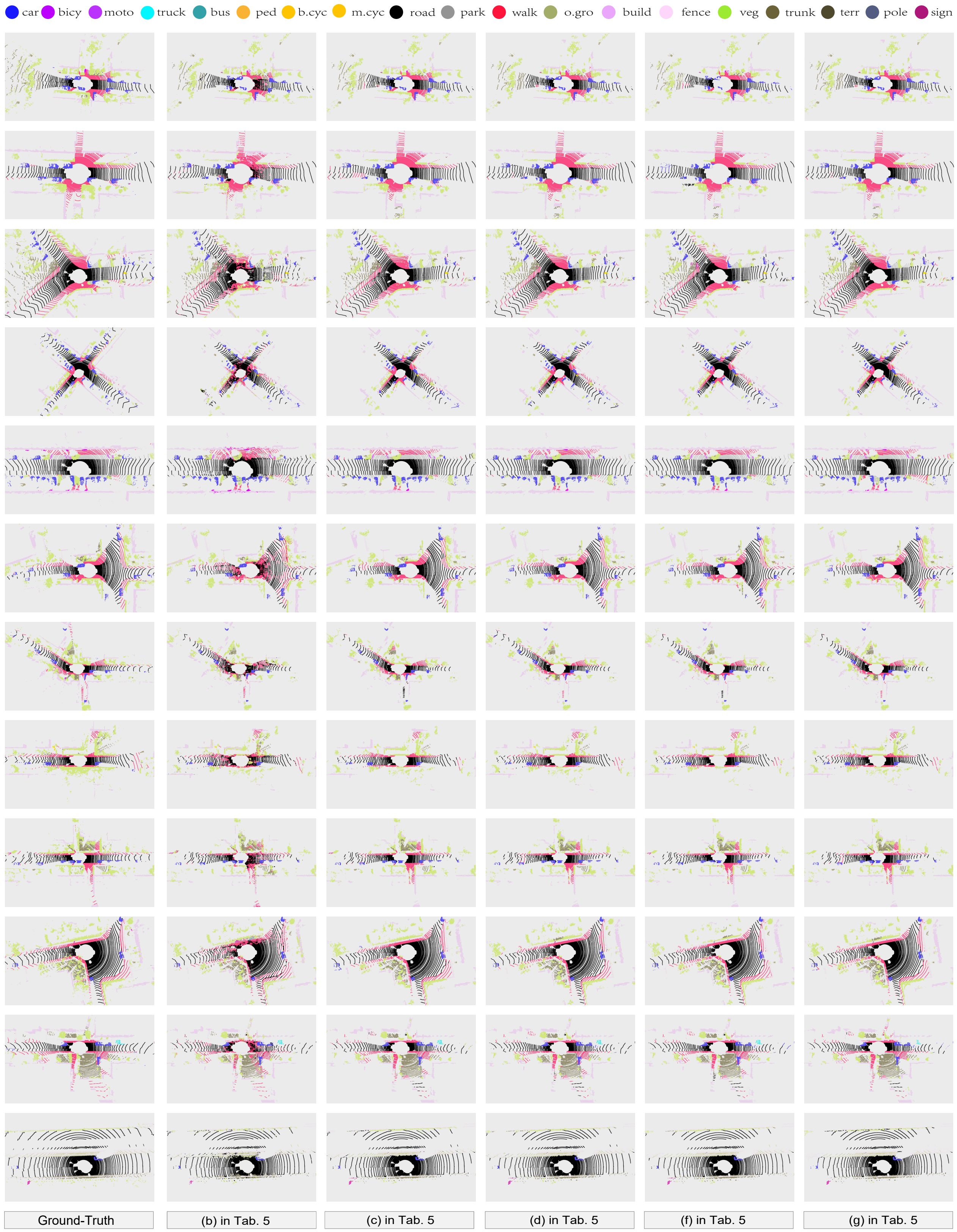

Figure A4. **Visualization of pseudo labels.** From left to right, each column represents the incremental addition of one component of `CLASS`, corresponding to each row in Tab. 5. Best viewed in color.

References

[1] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Juergen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, pages 9297–9307, 2019. 1, 2, 3, 4

[2] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11621–11631, 2020. 1, 2, 3

[3] Christopher B. Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, pages 3075–3084, 2019. 2, 3, 4

[4] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996. 3

[5] Lingdong Kong, Jiawei Ren, Liang Pan, and Ziwei Liu. Lasermix for semi-supervised lidar semantic segmentation. In *CVPR*, pages 21706–21716, 2023. 1, 2

[6] Yuyuan Liu, Yuanhong Chen, Hu Wang, Vasileios Belagiannis, Ian Reid, and Gustavo Carneiro. Ittakestwo: Leveraging peer representations for semi-supervised lidar semantic segmentation. In *ECCV*, pages 81–99. Springer, 2025. 2

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2018. 2

[8] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019. 2

[9] Cristiano Saltori, Fabio Galasso, Giuseppe Fiameni, Nicu Sebe, Elisa Ricci, and Fabio Poiesi. Cosmix: Compositional semantic mix for domain adaptation in 3d lidar segmentation. In *ECCV*, pages 586–602, 2022. 1

[10] Cristiano Saltori, Aljosa Osep, Elisa Ricci, and Laura Leal-Taixé. Walking your lidog: A journey through multiple domains for lidar semantic segmentation. In *ICCV*, pages 196–206, 2023. 1

[11] Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. *arXiv preprint arXiv:1708.07120*, 2017. 2

[12] Ozan Unal, Dengxin Dai, and Luc Van Gool. Scribble-supervised lidar semantic segmentation. In *CVPR*, 2022. 2

[13] Aoran Xiao, Jiaxing Huang, Dayan Guan, Kaiwen Cui, Shijian Lu, and Ling Shao. Polarmix: A general data augmentation technique for lidar point clouds. In *NeurIPS*, pages 11035–11048, 2022. 1

[14] Aoran Xiao, Jiaxing Huang, Dayan Guan, Fangneng Zhan, and Shijian Lu. Transfer learning from synthetic to real lidar point cloud for semantic segmentation. In *AAAI*, pages 2795–2803, 2022. 1

[15] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *CVPR*, pages 9601–9610, 2020. 2

[16] Yiming Zhao, Lin Bai, and Xinming Huang. Fidnet: Lidar point cloud semantic segmentation with fully interpolation decoding. In *IROS*, pages 4453–4458, 2021. 2