Supplementary Materials for Single Domain Generalization for LiDAR Semantic Segmentation

Hyeonseong Kim*, Yoonsu Kang*, Changgyoon Oh, and Kuk-Jin Yoon
Visual Intelligence Lab., KAIST, Korea

This supplementary material provides label mapping, implementation details, additional experimental results, and visual results. In particular, the following contents are included in the supplementary material:

- The label mapping for five datasets used in the experiments to the common 10 classes.
- Implementation details of the proposed method and baselines used in the experiments.
- Comparison with Complete&Label [13].
- Hyperparameter analysis of SIFC.
- Additional visual results.

A. Label Mapping

We present the label mapping of each class to the selected 10 common classes for five datasets, SemanticKITTI [1], nuScenes-lidarseg [2], Waymo [10], SemanticPOSS [8], and SynLiDAR [12] in Table 1. To elaborate on our choice of label mapping, we map the {bicyclist, motorcyclist} classes to the {background} because both cyclist classes consist of a rider and transportation, and this might lead to ambiguous predictions. The classes consisting of only humans are mapped to {pedestrian}. For SemanticPOSS, there are no corresponding classes for {motorcycle, truck, other-vehicle, sidewalk}, we only use 5 classes for evaluation. Also, the {trunk, plants} classes are normally mapped to {vegetation}, but those classes in SemanticPOSS actually contain {walkable} points, which is why we integrate {vegetation} to {walkable} only for SemanticPOSS.

B. Implementation Details

For all experiments, we use Adam optimizer [5] with lr=1e-3, β_1=0.9, and β_2=0.999 and batch size of 8. To voxelize the input point clouds, we first clip the data to fit the fixed volume space. We set a different volume space for each source dataset as in Table 2. Then, we voxelize the input using a voxel size of 0.2m. During training, we augment the input point clouds by random rotation along the z-axis, random flipping along the x, y, and x+y axis, arbitrary scaling, and random translation. We do not use intensity values and only use point cloud xyz coordinates.

Proposed Method. We use MinkowskiNet [3] as our backbone network consisting of the encoder Φ_{enc}, decoder Φ_{dec}, and classifier $\mathcal{C}$. Specifically, MinkUnet34 [1] is used where the last *final* layer corresponds to the classifier. The metric learner Ψ consists of 2-layer MLP whose hidden layer dimension is 96 and output layer dimension is 64. The loss weights (λ_1, λ_2) are (1, 10) for the SemanticKITTI, (0.3, 20) for the Waymo, (0.01, 0.1) for nuScenes-lidarseg, and (0.05, 10) for the SynLiDAR. We set threshold τ in SIFC to $\cos 45^\circ = 0.707$ and the augmentation sampling parameters (p_{min}, p_{max}) are (0.3, 0.7) for SemanticKITTI, Waymo, and SynLiDAR. The hyperparameters for nuScenes-lidarseg are $\tau = 0.84$ and (p_{min}, p_{max}) = (0.2, 0.4).

IBN-Net [7]. IBN-Net proposes to use a combination of Instance Normalization (IN) [11] and Batch Normalization [4] to learn appearance invariant features while maintaining content information. Among the variations of the IBN block, we use the IBN-b block, which shows the best performance in the semantic segmentation task. We add IN right after the second conv layer (conv2) and the second, third convolution group (conv3_x and conv4_x) of MinkUNet34.

MLDG [6]. MLDG proposes to utilize the meta-learning scheme for domain generalization that simulates the domain shifts during training. Considering that we have a single source domain, we split the source domain into meta-train and meta-test sets. Depending on how we split the source domain, we implement two variations: MLDG(A) and MLDG(B). At each iteration, MLDG(A) randomly splits the batch into two equal sized meta-train and meta-test sets. On the other hand, MLDG(B) splits the batch using pre-trained model trained on the source domain. The pre-trained model extracts the per-class features for each

*The first two authors contributed equally. In alphabetical order.

[1]We use the official code of Minkowski Engine for the implementation of MinkUNet34. Please refer to the code in https://github.com/NVIDIA/MinkowskiEngine/blob/master/examples/minkunet.py

Table 1. Label mapping for SemanticKITTI [1], nuScenes-lidarseg [2], Waymo [10], SemanticPOSS [8], and SynLiDAR [12].

Common class	SemanticKITTI	nuScenes-lidarseg	Waymo	SemanticPOSS	SynLiDAR
car	car, moving-car	vehicle.car	car	car	car
bicycle	bicycle	vehicle.bicycle	bicycle	bike	bicycle
motorcycle	motorcycle	vehicle.motorcycle	motorcycle	-	motorcycle
truck	truck, moving-truck	vehicle.truck	truck	-	pick-up, truck
other-vehicle	bus, moving-bus, other-vehicle, moving-on-rails, moving-other-vehicle	vehicle.bus.bendy, vehicle.bus.rigid, vehicle.construction, vehicle.trailer	bus, other-vehicle	-	bus, other-vehicle
pedestrian	person, moving-person	human.pedestrian.adult, human.pedestrian.child, human.pedestrian.construction_worker, human.pedestrian.police_officer	pedestrian	1 person, 2+ person	female, male, kid, crowd
drivable-surface	road, parking, lane-marking	flat.driveable_surface	road, lane-marker, other-ground	ground	road, parking
sidewalk	sidewalk	flat.sidewalk	curb, sidewalk	-	sidewalk
walkable	terrain	flat.terrain	walkable	trunk, plants	terrain
vegetation	vegetation, trunk	static.vegetation	vegetation, trunk	-	vegetation, trunk
background	unlabeled, outlier, bicyclist, motorcyclist, other-ground, building, fence, other-structure, pole, traffic-sign, other-object, moving-bicyclist, moving-motorcyclist	noise, animal, human.pedestrian.personal.mobility, human.pedestrian.stroller, human.pedestrian.wheelchair, movable_object.barrier, movable_object.debris, movable_object.pushable_pullable, movable_object.trafficcone, static_object.bicycle_rack, vehicle.emergency.ambulance, vehicle.emergency.police, static.other, vehicle.ego	undefined, motorcyclist, bicyclist, sign, pole, traffic-light, building, construction-cone	rider, traffic-sign, pole, building, garbage-can, cone/stone, fence, unlabeled	unlabeled, other-ground, bicyclist, motorcyclist, building, other-structure, traffic-sign, pole, traffic-cone, fence, garbage-can, table, chair, bench, other-object

Table 2. Volume space used for each dataset.

Dataset	x(m)	y(m)	z(m)
SemanticKITTI	[-50, 50]	[-50, 50]	[-4, 2]
nuScenes-lidarseg	[-50, 50]	[-50, 50]	[-5, 3]
Waymo	[-75, 75]	[-75, 75]	[-4, 2]
SemanticPOSS	[-75, 75]	[-75, 75]	[-4, 4]
SynLiDAR	[-75, 75]	[-75, 75]	[-5, 3]

data in the batch by aggregating the features of the decoder outputs corresponding to each class. Then, we randomly select a LiDAR scan for meta-train set, and find the remaining data whose cosine feature distances are close to the selected one. Data with distant feature distances are assigned to the meta-test set. After obtaining the training sets, we follow the training procedure of [6] described in Algorithm 1. The model parameters θ are first updated using meta-train loss $\mathcal{L}_{train}(\theta)$ on the meta-train set. Then, the updated parameters $\theta' = \theta - \alpha\nabla_\theta\mathcal{L}_{train}$ are virtually evaluated through meta-test loss $\mathcal{L}_{test}(\theta')$ on the meta-test set. The final objective is the combination of both losses, $\mathcal{L}_{train}(\theta) + \beta\mathcal{L}_{test}(\theta - \alpha\nabla_\theta\mathcal{L}_{train})$. Finally, the parameters are updated as follows:

$$\theta = \theta - \gamma\frac{\partial(\mathcal{L}_{train}(\theta) + \beta\mathcal{L}_{test}(\theta - \alpha\nabla_\theta\mathcal{L}_{train}))}{\partial\theta}, \quad (1)$$

where we use $\alpha = 0.005$, $\beta = 1$, and $\gamma = 0.001$.

CoSMIX [9]. We follow the overall training settings of original paper [9] and pre-train the model on the source dataset for 10 epochs with weighted cross-entropy loss used to train our base model. The teacher and student models

Algorithm 1 MLDG Baseline Training Procedure

```
Input: Source domain $S$, minibatch size $B$, meta learning rate $\alpha$, learning rate $\gamma$, loss weight $\beta$, loss function $\mathcal{L}_{sem}$
Initialize: Parameters $\theta$ of model $f$ at iter = 0
1:  while $iter < T$ do
2:      Split $S$ into $S^{tr}$, $S^{te}$ following (A) or (B) setting
3:      Meta-Training:
4:          $\mathcal{L}_{train}(\theta) \leftarrow \mathcal{L}_{sem}(f_\theta(\{\hat{y}_b^{tr}\}_{b=1}^B), \{\tilde{y}_b^{tr}\}_{b=1}^B)$
5:          Update $\theta' \leftarrow \theta - \alpha\nabla_\theta L_{train}$
6:      Meta-Testing:
7:          $\mathcal{L}_{test}(\theta') \leftarrow \mathcal{L}_{sem}(f_{\theta'}(\{\hat{y}_b^{te}\}_{b=1}^B), \{\tilde{y}_b^{te}\}_{b=1}^B)$
8:          Compute $\mathcal{L}_{train}(\theta) + \beta\mathcal{L}_{test}(\theta')$
9:      Update $\theta \leftarrow \theta - \gamma\nabla_\theta(L_{train}(\theta) + \beta L_{test}(\theta'))$
10:     $iter \leftarrow iter + 1$
11: end while
```

used in the adaptation process are initialized with the pre-trained weights. For SemanticKITTI→nuScenes-lidarseg and SemanticKITTI→Waymo, we adapt for 4 epochs, and the Waymo→SemanticKITII, Waymo→nuScenes-lidarseg models are trained for 7 epochs. We follow the original paper and set α=0.5, ζ=0.9, and all the other hyperparameters are the same with the original setting.

C. Comparison with Complete&Label

Since Complete&Label [13] evaluated their method in a DG setting with 2 classes (pedestrian and vehicle), we additionally conduct comparative experiments. Since there is no official code available, we compare our method with the settings used in [13]. In Table 3, we show the quantita-

Table 3. Per-dataset mIoU(%) evaulated on Semantic**K**ITTI (**K**) and **n**uScenes-lidarseg (**N**) compared with the Complete&Label. The Waymo dataset is used as the source domain.

Method	K	N
Base	55.0	42.9
Complete&Label [13]	59.6	49.8
Ours	**79.9**	**68.1**

Table 4. Results of using different sparsity and the threshold τ.

		K	N	W	P	AM	HM
Beam sampling ratio (%)	10-50	58.90	43.79	38.17	44.10	46.24	45.10
	20-60	59.30	43.39	38.63	43.07	46.10	44.94
	30-70	**59.62**	**44.83**	**40.67**	**45.09**	**47.55**	**46.60**
Threshold τ	0.9	59.95	44.02	38.46	43.82	46.56	45.34
	0.7	59.62	**44.83**	40.67	**45.09**	47.55	**46.60**
	0.5	**60.44**	44.24	**41.31**	44.47	**47.62**	**46.60**
	0.3	60.06	44.49	39.11	43.07	46.69	45.50

tive comparison with [13] and report the mIoU of the two classes. The results show that our method has better generalization performance.

D. Hyperparameter Analysis of SIFC

We also conduct a hyperparameter analysis of SIFC using different levels of sparsity and threshold τ. As shown in Table 4, the performance decreases as the augmented domain becomes sparser, as learning with sparse data of less information is challenging. However, it can be seen that the overall performance is robust to the degree of sparsity variation of the augmented domain. Also, as the value of τ decreases, performance improves but begins to decline at τ=0.3. This confirms that aggregating unreliable nearby features has negative effects.

E. Additional Visual Results

We provide additional qualitative comparisons between the baselines and our method in the following figures. Fig. 1, Fig. 2, Fig. 3 shows the segmentation results when trained on SemanticKITTI and tested on nuScenes-lidarseg, Waymo, SemanticPOSS, respectively. And Fig. 4, Fig. 5, Fig. 6 are the segmentation results when trained on Waymo and tested on SemanticKITTI, nuScenes-lidarseg, SemanticPOSS, respectively.

References

[1] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9297–9307, 2019. 1, 2

[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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 1, 2

[3] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 1

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 1

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[6] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. 1, 2

[7] Xingang Pan, Ping Luo, Jianping Shi, and Xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 464–479, 2018. 1

[8] Yancheng Pan, Biao Gao, Jilin Mei, Sibo Geng, Chengkun Li, and Huijing Zhao. Semanticposs: A point cloud dataset with large quantity of dynamic instances. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 687–693. IEEE, 2020. 1, 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 *European Conference on Computer Vision*, pages 586–602. Springer, 2022. 2

[10] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020. 1, 2

[11] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6924–6932, 2017. 1

[12] Aoran Xiao, Jiaxing Huang, Dayan Guan, Fangneng Zhan, and Shijian Lu. Transfer learning from synthetic to real lidar point cloud for semantic segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2795–2803, 2022. 1, 2

[13] Li Yi, Boqing Gong, and Thomas Funkhouser. Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15363–15373, 2021. 1, 2, 3

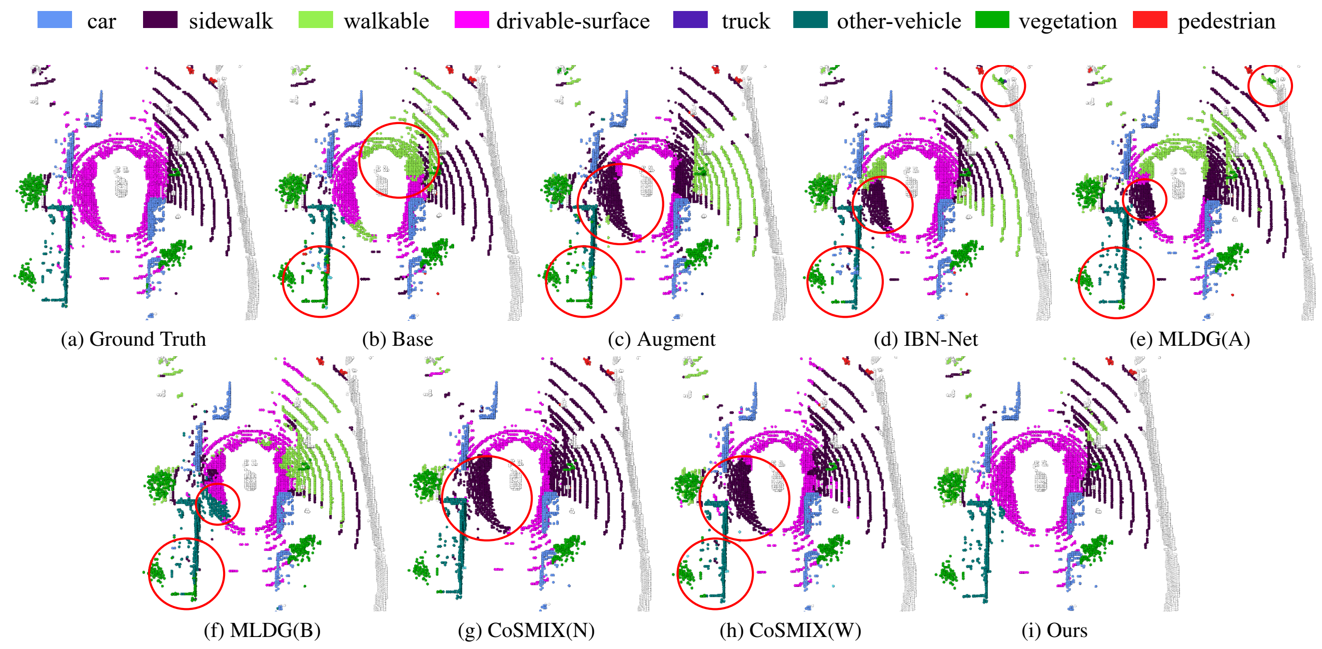

Figure 1. Qualitative comparisons between the proposed method and the baseline methods when trained on **SemanticKITTI** and tested on **nuScenes-lidarseg**. The circles shown in the figure indicate the misclassified parts.

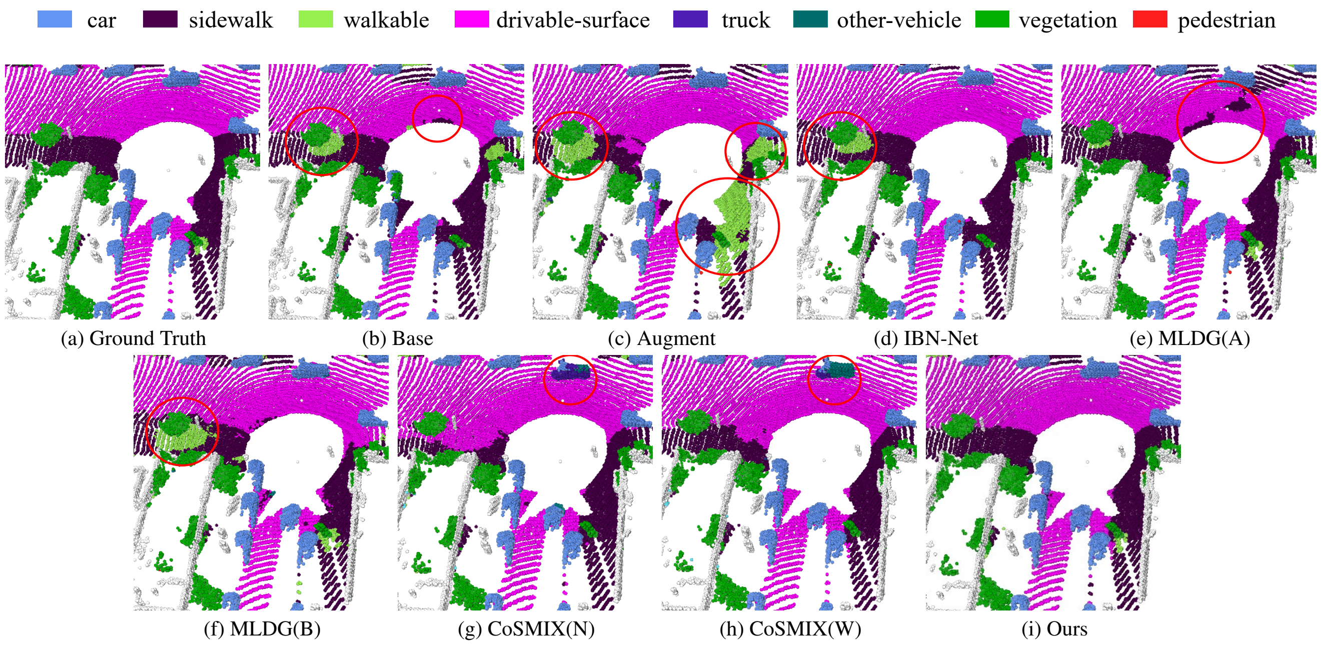

Figure 2. Qualitative comparisons between the proposed method and the baseline methods when trained on **SemanticKITTI** and tested on **Waymo**. The circles shown in the figure indicate the misclassified parts.

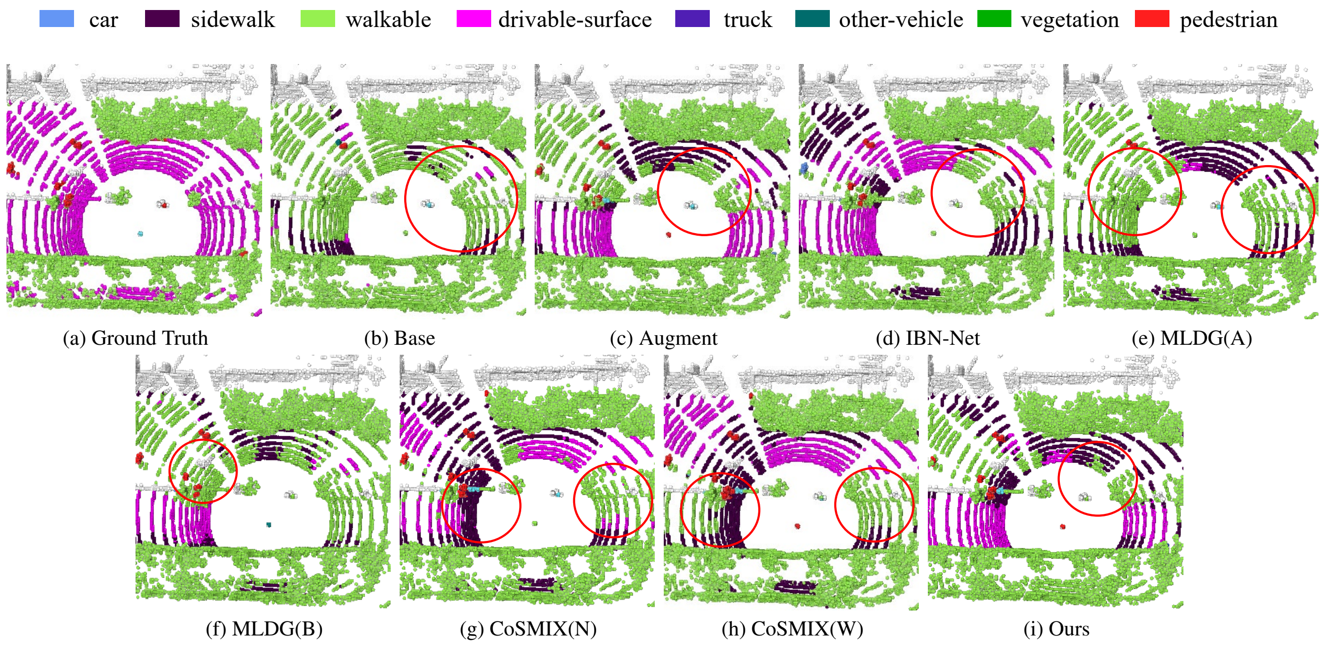

Figure 3. Qualitative comparisons between the proposed method and the baseline methods when trained on **SemanticKITTI** and tested on **SemanticPOSS**. The circles shown in the figure indicate the misclassified parts.

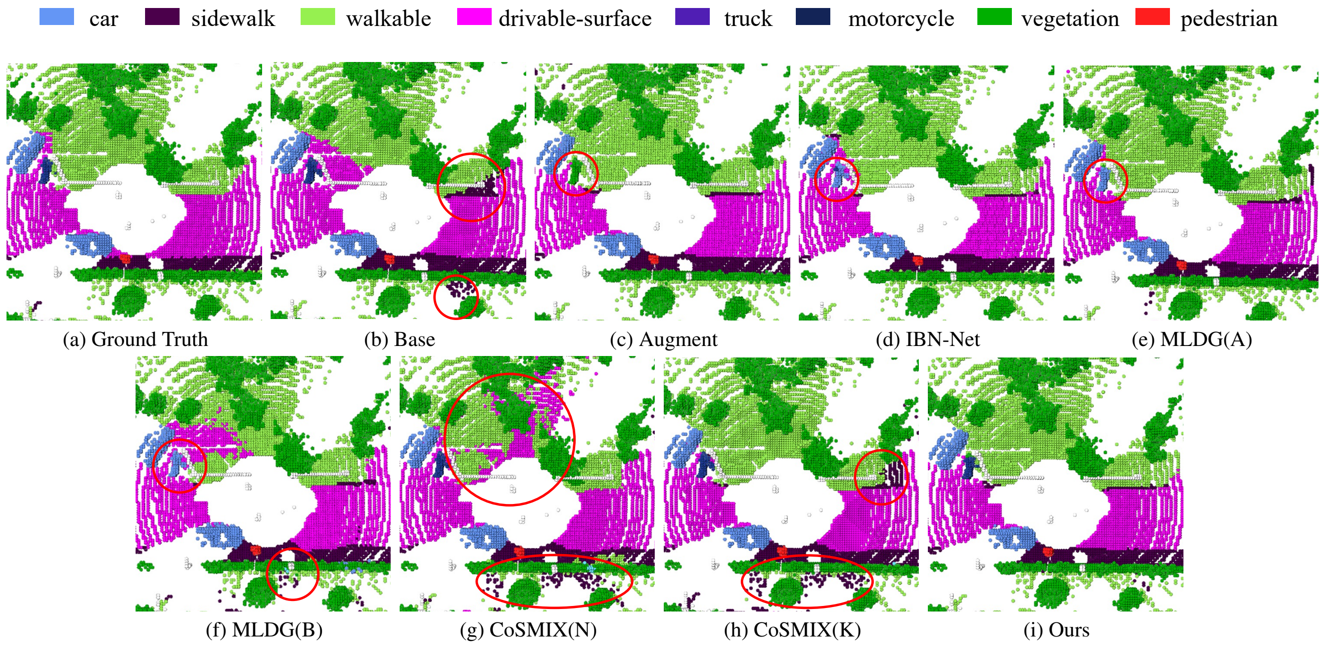

Figure 4. Qualitative comparisons between the proposed method and the baseline methods when trained on **Waymo** and tested on **SemanticKITTI**. The circles shown in the figure indicate the misclassified parts.

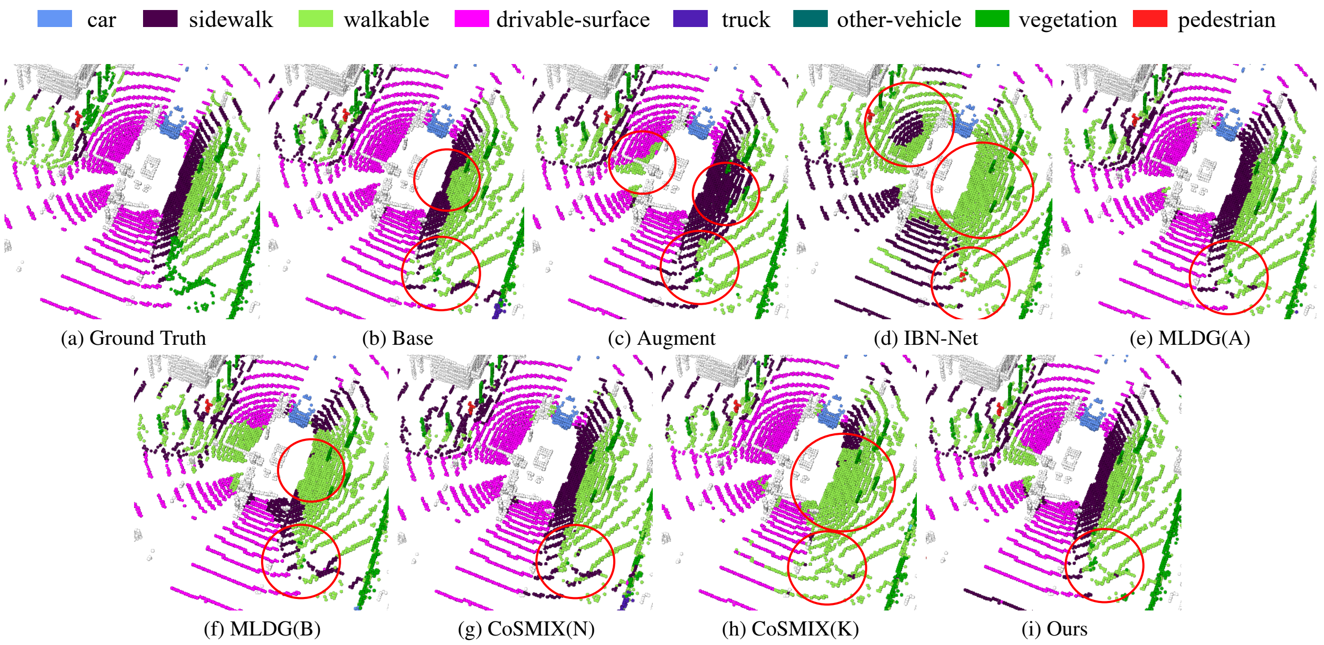

Figure 5. Qualitative comparisons between the proposed method and the baseline methods when trained on **Waymo** and tested on **nuScenes-lidarseg**. The circles shown in the figure indicate the misclassified parts.

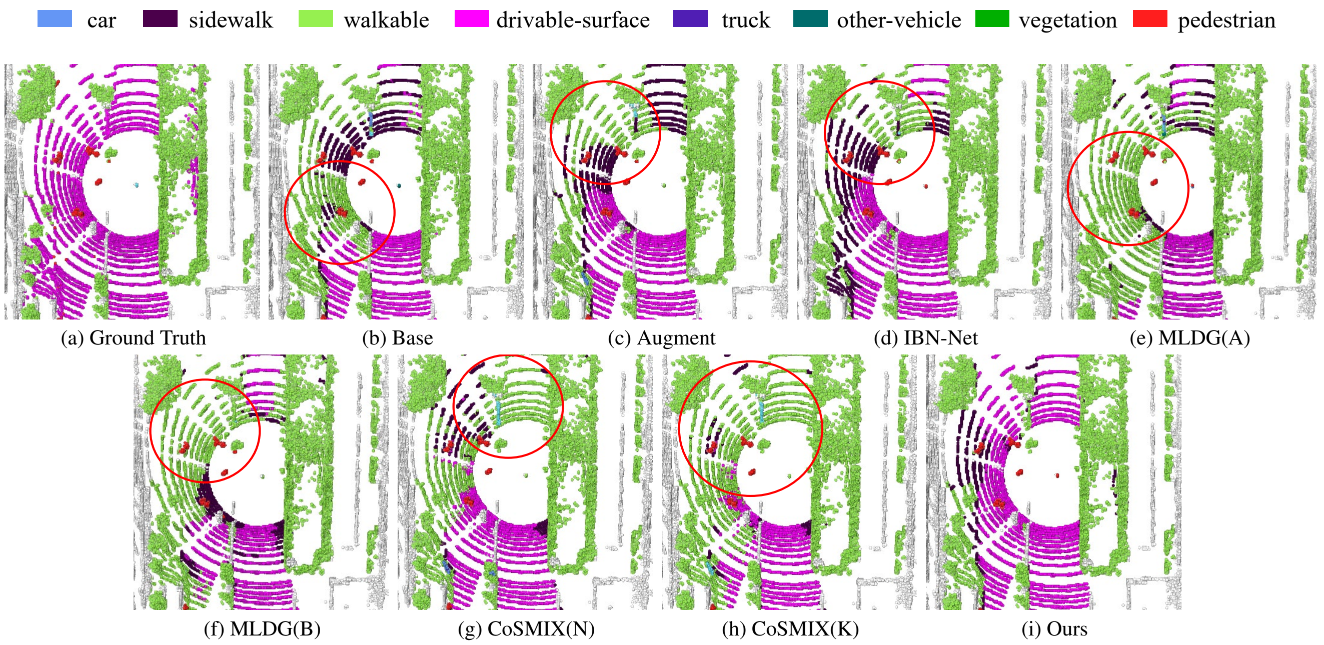

Figure 6. Qualitative comparisons between the proposed method and the baseline methods when trained on **Waymo** and tested on **SemanticPOSS**. The circles shown in the figure indicate the misclassified parts.