ItTakesTwo: Leveraging Peer Representations for Semi-supervised LiDAR Semantic Segmentation (Supplementary Material)

Table 1: Partial sampling results in SemanticKITTI [1] benchmark based on batch-wise evaluation following [4]. TTA indicates the test time augmentation and the results "w/o" TTA are reported based on the checkpoints from the official GitHub [1]. The best results are marked in red.

TTA	Method	SemanticKITTI (partial)			
		5%	10%	20%	40%
✗	GPC [4]	42.10	48.30	57.90	59.32
	IT2	45.97 (3.87↑)	50.87 (2.57↑)	60.33 (2.43↑)	63.31 (3.99↑)
✓	GPC [4]	42.45	48.77	58.78	59.96
	IT2	46.44 (3.99↑)	51.97 (3.20↑)	61.43 (2.69↑)	64.83 (4.87↑)

1 Different Evaluation Process

The partial sampling approach GPC [4] employs a **distinct evaluation protocol**[2] comparing to the common LiDAR point semantic segmentation methods [7,8,18][3]. We emphasise that different evaluation protocols can lead to unfair competition. In Tab. 1, we present our results based on the evaluation process following the approach in GPC [4]. *We highlight that our reported results with all the labelled ratios are derived from consistent checkpoints in the main paper Tab. 2.* The 'TTA' indicates the test-time augmentation, where the paper results of GPC [4] are all based on the 'TTA'.
Our method achieves the best performance across various labelled ratios. For instance, in the case of 5% and 40% labelled data with TTA post-processing, we outperform GPC [4] by 3.99% and 4.87% in mIoU, respectively. These consistent improvements underscore the robustness of peer representation in label-efficient LiDAR segmentation and also show the effectiveness of the IT2 approach.

2 Representation-specific Data Augmentation

As shown in the Fig. 1, we perform the augmentations for distinct representations to cases (a), (b) and (c) in a batch of inputs and the mixed results **(d)** are in the last column. We experimentally observed that utilising such different augmentations in different representations can achieve better performance.
Voxel representation augmentation. We implement LaserMix [6] for augmentation of our voxel inputs as illustarted in the top row of Fig. 1. We set

[1] https://github.com/llijiang/GuidedContrast/tree/main?tab=readme-ov-file#semantickitti-2
[2] https://github.com/llijiang/GuidedContrast/blob/add37a8ecf68a59698d6b6aa73735d94ae9c002d/util/utils.py#L23
[3] https://github.com/xinge008/Cylinder3D/blob/30a0abb2ca4c657a821a5e9a343934b0789b2365/utils/metric_util.py#L19

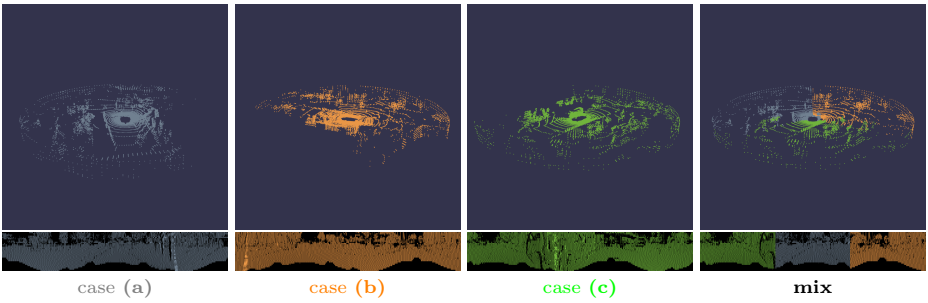


Fig. 1: Illustration of our representation-specific data augmentation. In the first row, we apply single-inclination LaserMix [6] for the voxel grids and in the second row, we apply multi-boxes CutMix [16] for the range images. The mixed results for each representations are displayed in the last column, where the different colors demonstrate the mix protocol of case (a), (b) and (c).

$\alpha = \frac{360}{\text{batch size}}$ and $\phi = 1$ to achieve optimal performance, where α represents azimuth, and ϕ denotes the inclination direction. For further details, please refer to Table 4 in the LaserMix [6] paper.

Range representation augmentation. As demonstrated in the bottom row of Fig. 1, we incorporate CutMix [16] into our range images, utilising multi-boxes cropped within the same mini-batch. To prevent overlap between these boxes, we set the width of each box to $\frac{\text{image width}}{\text{batch size}}$ and they are mixed in box-by-box manner.

Algorithm 1.1: Pseudo code of the data augmentation process

```
# x_voxel, x_range: voxel, range representations from same point scan.
# f_voxel, f_range: voxel, range networks.
y_voxel = range2voxel(model_range(x_range)) # transfer pseudo labels: range → voxel
y_range = voxel2range(model_voxel(x_voxel)) # transfer pseudo labels: voxel → range
x_voxel, y_voxel = voxel_augment(x_voxel, y_voxel) # voxel augmentation [6]
x_range, y_range = range_augment(x_range, y_range) # range augmentation [16]
# {x_range, y_range} and {x_voxel, y_voxel} are the augmented samples and labels.
```

Algorithm 1.1 shows the whole process of the representation-specific data augmentation in our approach in python coding style. Following [10, 15], these augmentations are carried out after generating the pseudo-labels (i.e., Eq. (2) from the main paper) for each representation. These augmentations are applied for both inputs and the labels, while their mixed results are utilised for the training of the unlabelled data.

3 Additional Implementation Details

In this section, we provide more implementation details of the model configuration and our proposed contrastive learning approach.

3.1 Model Configuration

Cylinder3D [18] setup. Following Lasermix [6], we employ the Cylinder3D [18] network for the voxel representation. For the *Uniform Sampling* strategy in SemanticKITTI [1] and nuScenes [2] benchmarks, we use an input resolution of $[240, 180, 20]$ and reduce the feature map to 16 for fair comparison. We adopt the original configuration from the paper [18] with a resolution of $[480, 360, 32]$ and a feature map size of 32 for all other experiments.

FidNet [17] setup. We use FidNet [17] for the range images, following [6], where we employ ResNet-34 [3] variants for all experiments. Similarly, in the *Uniform Sampling* strategy of SemanticKITTI [1] and nuScenes [2] benchmarks, we set the resolution to 32×1920 and 64×2048 for a fair comparison with [6]. For all other experiments, we maintain a fixed image width of 64×960 for the enhanced efficiency.

3.2 Contrastive Learning Configuration

Following [5, 14], we employ a 3-layer projector to generate embedding results for intermediate features. The projector has the architecture of '`convolution layer, batch-norm layer, convolution layer`'. For the voxel representation, we use the 3D convolution layer from the SparseConv library[4], while 2D convolution is utilised for the range images following the common setup [8, 9, 14]. The depth of the embedding features is fixed at 64 for all experiments.

Sampling Strategies. The current hardware cannot handle the contrastive learning in the dense tasks, and a sampling strategy is necessary for all contrastive learning based methods [9, 14]. We adopt the easy-hard mining for the dense embedding samples based on [14]. The easy embedding samples are randomly selected based on the dense results where the predictions equal with (pseudo or real) labels, while the hard embedding samples are chosen based on the results that exhibit disagreement between the prediction and the (pseudo or real) label. We follow [14] to split the ratio of these easy and hard samples with the ratio being $1:1$, and the total number is set to 200 for both voxel and range representations in all experiments.

3.3 Training Configuration

In the nuScenes [2] dataset, we use a batch size of 4 for both labelled and unlabelled data. The learning rate is set to $6e^{-3}$, and the maximum number of epochs is set to 90. In the SemanticKITTI [1] dataset, we opt for a smaller batch size of 2 for both labelled and unlabelled data, with a learning rate of $8e^{-3}$ and a maximum of 105 epochs. For both datasets, we employ 4 GPUs for distributed data parallel training, incorporating a *polynomial* learning rate decay

[4] https://github.com/traveller59/spconv

Table 2: The fusion results in the nuScenes [2] dataset based on the prediction from range and voxel representations. The best results are in red and the second best results are in *italic*.

Repr.	nuScenes [2]			
	1%	10%	20%	50%
range	56.76	71.33	73.27	74.04
voxel	*56.96*	*72.12*	*73.55*	*74.14*
ensemble	58.14 (1.18↑)	73.77 (1.65↑)	75.29 (1.74↑)	76.54 (2.40↑)

of $(1 - \frac{\text{iter}}{\text{max\_iter}})^{0.9}$. We use the AdamW [11] optimizer for all experiments with a decay of 0.001, and beta values in the range [0.9, 0.999].

3.4 Gaussian Mixture Model Configuration

We employ category-wise Gaussian Mixture Model (GMM) to learn from the incoming embedding features for both range and voxel representations in Eq. (5) of the main paper, where each category in the dataset have 5 Gaussian Curves. The GMMs' parameters are updated in each iteration based on the exponential moving average (EMA) to keep track of their historical distribution via $\Gamma^{(t+1)} = \alpha \times \Gamma^{(t)} + (1-\alpha) \times \Gamma$, where $\alpha = 0.996$ for all experiments, where Γ is from Eq. (6) main paper. In the sampling strategy of the virtual prototype, we firstly identify the related GMMs based on the categories of each of the incoming feature samples. Then we choose the Gaussian within the GMMs based on the probability distribution of $\mathbf{q}_m^y$ from Eq. (7) in the main paper for each of the samples. After that, we assign the random variable $\xi \in [0, 1]$ and generate the prototypes $\mathbf{z}^p$ for each feature samples, where $\mathbf{z}^p = \mu_m^y + \xi \times \mathbf{\Sigma}_m^y$ with μ_m^y, $\mathbf{\Sigma}_m^y$ from Eq. (7).

4 Fusion Results

In Tab. 2, we present the fusion results obtained from the models' outputs of both voxel and range representations in the nuScenes [2] dataset. Notably, we observe consistent improvements in the fusion results compared to the individual representation results. For example, it demonstrates 1.18% and 2.40% improvements in 1% and 50% labelled partition protocol, respectively. Given that the fusion prediction yields better generalisation, how to ensemble the pseudo label of the unlabelled data during the training can be an interesting topic in the future research.

5 Detailed Results

We have provided the class-wise Intersection-over-Union (IoU) validation results in the Tab. 3, Tab. 4 and Tab. 5 for the nuScenes [2], SemanticKITTI [1] and ScribbleKITTI [13] datasets, respectively. The mIou results follow the table results in the main paper and they are highlighted in red.

Table 3: The class-wise IoU results in nuScenes [2] dataset among different partition protocol. The mIoU results are highlighted in red.

Repr.	ratio	mean	barr	bicy	bus	car	const	moto	ped	cone	trail	truck	driv	othe	walk	terr	manm	veg
Range	1%	**56.5**	63.6	1.9	63.0	84.8	9.0	57.7	62.8	51.1	17.2	44.1	94.5	53.2	64.1	69.9	83.4	83.5
	10%	**71.3**	75.0	26.4	81.1	90.1	39.9	77.2	76.0	61.5	56.5	73.3	96.2	66.7	72.5	74.4	87.8	86.6
	20%	**73.4**	75.9	36.9	84.4	90.7	43.9	79.9	77.8	64.8	58.1	75.6	96.3	68.6	73.2	74.3	88.0	87.1
	50%	**74.0**	76.1	38.2	85.0	89.4	45.5	76.6	77.9	64.9	66.7	76.3	96.3	69.5	73.4	74.2	88.2	87.0
Voxel	1%	**57.5**	58.2	3.8	67.6	83.3	16.9	63.0	62.9	47.2	18.5	47.1	94.0	53.5	64.8	70.3	83.8	84.8
	10%	**72.1**	72.4	26.7	89.7	91.0	46.8	75.7	73.2	58.2	57.1	80.0	95.8	68.4	72.4	73.2	87.5	85.9
	20%	**73.6**	74.0	34.0	90.7	91.3	47.9	76.8	76.4	60.9	57.8	79.8	95.8	67.8	73.6	73.3	88.7	86.5
	50%	**74.1**	73.3	33.5	91.7	91.1	46.9	77.8	75.2	59.8	65.3	80.9	95.8	72.4	73.3	74.1	88.1	86.4

Table 4: The class-wise IoU results in SemanticKITTI [1] dataset among different partition protocol. The mIoU results are highlighted in red.

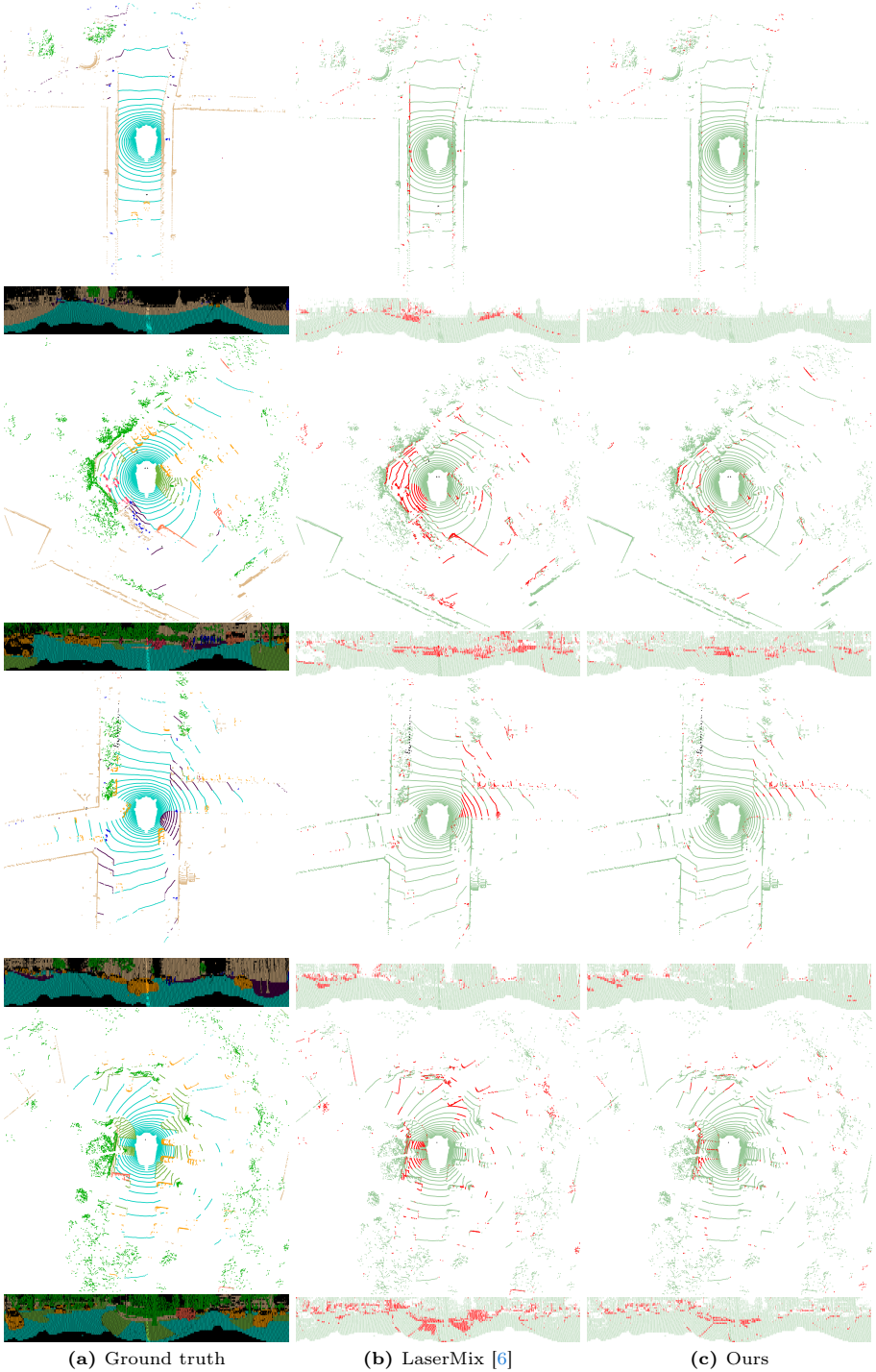
Repr.	ratio	mean	car	bicy	moto	truck	bus	ped	b.cyc	m.cyc	road	park	walk	o.gro	build	fence	veg	trunk	terr	pole	sign
Range	5%	**57.0**	93.1	44.8	51.5	67.3	42.5	44.8	54.0	0.0	93.9	42.4	80.4	0.0	87.7	54.0	87.3	62.5	77.4	58.6	40.1
	10%	**62.4**	95.2	46.3	56.5	69.7	47.9	73.2	80.7	0.0	95.3	46.9	83.1	1.9	87.9	57.4	88.6	67.5	77.6	65.1	44.8
	20%	**62.7**	95.4	46.3	59.1	90.6	51.1	71.1	81.8	0.0	95.5	39.1	82.7	1.6	86.7	51.1	87.4	66.5	76.1	66.5	43.1
	40%	**63.9**	94.8	54.8	62.1	70.9	45.0	73.6	82.3	0.0	95.6	52.1	83.9	10.7	88.9	59.4	87.1	67.2	74.8	65.8	45.3
Voxel	5%	**60.3**	92.9	49.5	43.9	85.7	40.6	65.0	83.0	0.0	91.5	35.8	77.5	0.3	89.4	54.2	86.5	66.5	73.0	64.6	45.7
	10%	**63.3**	94.9	50.9	70.9	78.8	48.2	74.5	84.2	0.0	94.1	42.9	79.8	2.9	87.4	51.3	87.7	66.3	74.3	63.0	50.8
	20%	**64.0**	96.7	54.1	73.6	74.4	59.5	75.4	86.8	0.3	93.5	41.8	79.5	1.2	88.3	50.6	86.1	67.6	69.5	64.8	52.1
	40%	**64.8**	95.7	50.3	75.4	79.4	52.5	75.4	90.8	1.6	94.7	46.9	81.6	1.0	87.3	52.4	87.5	69.2	75.1	64.6	49.6

Table 5: The class-wise IoU results in ScribbleKITTI [13] dataset among different partition protocol. The mIoU results are highlighted in red.

Repr.	ratio	mean	car	bicy	moto	truck	bus	ped	b.cyc	m.cyc	road	park	walk	o.gro	build	fence	veg	trunk	terr	pole	sign
Range	1%	**46.6**	79.4	31.8	28.3	35.9	11.4	39.2	60.0	0.0	73.2	16.7	65.8	0.2	86.5	46.7	82.1	62.2	66.	60.8	40.2
	10%	**57.1**	92.4	47.5	45.4	67.4	30.3	55.2	62.8	0.0	92.9	40.9	79.4	1.6	87.7	52.6	84.7	66.3	72.5	60.2	45.9
	20%	**57.3**	87.9	33.0	43.9	66.8	43.3	59.6	63.0	0.0	91.7	40.6	79.6	6.1	88.2	53.3	82.6	67.5	74.3	61.3	46.7
	50%	**58.6**	86.7	35.1	51.7	77.4	49.3	66.1	74.5	0.1	87.5	31.1	76.3	8.7	88.3	45.3	85.0	66.8	71.1	64.0	47.7
Voxel	1%	**47.9**	85.4	29.2	37.7	20.4	24.2	45.9	53.1	0.0	77.0	20.3	67.7	0.5	83.2	49.4	77.6	64.5	64.9	61.7	48.1
	10%	**56.7**	93.1	43.1	47.2	63.8	33.2	60.6	70.0	0.0	89.5	34.1	74.6	1.4	87.4	51.9	84.7	61.1	72.5	60.9	48.8
	20%	**57.5**	92.0	49.5	45.1	68.4	30.6	56.1	60.8	0.0	93.2	41.2	80.1	2.9	88.5	52.1	85.4	66.9	73.4	60.8	46.2
	50%	**58.3**	89.2	48.7	46.7	73.2	42.0	62.2	74.0	0.0	83.3	40.3	79.1	3.2	85.3	54.2	80.6	67.1	64.4	64.2	50.1

6 Error Maps Visualisation

In Fig. 2, we present additional visualizations of our IT2 framework on the nuScenes [2] dataset, under 10% labelled partition protocol, comparing it with LaserMix [6]. Each case has two rows, with the top row showing the *bird's eye view* results and the bottom row presenting the range images. Correct predictions are highlighted in green, and mistakes are indicated in red. Our approach consistently yields visually superior results across all cases.



(a) Ground truth **(b)** LaserMix [6] **(c)** Ours

Fig. 2: Additional error maps visualised from LiDAR *bird's eye view* (top) and *range view* (bottom) in the nuScenes [12] dataset under 10% labelled partition protocol.

7 Additional Ablation studies

Importance of the number of GMM components. We study the number of GMM components in Tab. 6, using the nuScenes dataset [2] under the 10% labelled data protocol. When $M = 5$, we notice an improvement over the single multi-variate component ($M = 1$), with an increase of 0.9 mIoU in range and 0.7 mIoU in voxel representations. However, increasing the number of components to $M = 7$ results in a slight decrease in performance.

Table 6: Ablation study of Components (M) in Gaussian Mixure Model (GMM) on the nuScenes dataset under the 10% labelled data protocol. The best results are highlighted in red.

nuScenes (with 10% labelled data)				
Component (M)	M=1	M=3	M=5	M=7
range Repr.	70.4	70.8	71.3	71.2
voxel Repr.	71.4	71.7	72.1	71.9

LaserMix [6] with contrastive learning. As shown in Tab. 7, we applied our cross-distribution (GMM-based) contrastive learning (ctrs.) to LaserMix [6] using the voxel representation, as their range model's code is not public. Although LaserMix+ctrs. improves over the original LaserMix by 0.9% and 1.2% under the 1% and 10% labelled protocols on the nuScenes dataset [2], respectively, our IT2 provides further improvements over LaserMix+ctrs of 1.3% and 1.1%, respectively.

Table 7: Comparison between our IT2 and LaserMix [6] with our contrastive learning (i.e., ctrs.) on the nuScenes dataset based on voxel representation. Our results are highlighted in red.

Method	nuScenes	
	1%	10%
LaserMix [6]	55.3	69.9
LaserMix [6] +ctrs.	56.2	71.0
IT2	57.5 (1.3↑)	72.1 (1.1↑)

Sensitiveness of the parameter Temperature (Temp.) in Contrastive learning. As demonstrated in Tab. 8, the Temp. parameter significantly influences both the ContrasSeg [14] method and our approach. Minor adjustments

from $t = 0.05$ to $t = 0.10$ and from $t = 0.10$ to $t = 0.15$ result in nearly 1% performance differences for both methods. Such large differences have also been noted in the 'Supervised Contrastive Learning' paper [5] (Page 8, Fig.4). A potential reason for this is the relatively high weight of 1 to the InfoNCE loss for all experiments.

Table 8: Ablation study of different **temperature (Temp.)** values comparing our methods with ContrasSeg [14] on the nuScenes dataset, under the 10% labelled data protocol. The best results for both methods are highlighted in red.

Repr.	range			voxel		
Temp. (t)	t=0.05	t=0.10	t=0.15	t=0.05	t=0.10	t=0.15
ContrasSeg [14]	69.4	70.3	69.3	70.1	71.2	70.5
Ours	70.3	71.3	70.4	71.0	72.1	71.5

References

1. Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J.: Semantickitti: A dataset for semantic scene understanding of lidar sequences. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9297–9307 (2019) 1, 3, 4, 5
2. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11621–11631 (2020) 3, 4, 5, 7
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 3
4. Jiang, L., Shi, S., Tian, Z., Lai, X., Liu, S., Fu, C.W., Jia, J.: Guided point contrastive learning for semi-supervised point cloud semantic segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6423–6432 (2021) 1
5. Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C., Krishnan, D.: Supervised contrastive learning. Advances in Neural Information Processing Systems **33**, 18661–18673 (2020) 3, 8
6. Kong, L., Ren, J., Pan, L., Liu, Z.: Lasermix for semi-supervised lidar semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21705–21715 (2023) 1, 2, 3, 5, 6, 7
7. Lai, X., Tian, Z., Jiang, L., Liu, S., Zhao, H., Wang, L., Jia, J.: Semi-supervised semantic segmentation with directional context-aware consistency. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1205–1214 (2021) 1
8. Li, L., Shum, H.P., Breckon, T.P.: Less is more: Reducing task and model complexity for 3d point cloud semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9361–9371 (2023) 1, 3
9. Liu, Y., Ding, C., Tian, Y., Pang, G., Belagiannis, V., Reid, I., Carneiro, G.: Residual pattern learning for pixel-wise out-of-distribution detection in semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1151–1161 (2023) 3
10. Liu, Y., Tian, Y., Chen, Y., Liu, F., Belagiannis, V., Carneiro, G.: Perturbed and strict mean teachers for semi-supervised semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4258–4267 (2022) 2
11. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017) 4
12. Nunes, L., Marcuzzi, R., Chen, X., Behley, J., Stachniss, C.: Segcontrast: 3d point cloud feature representation learning through self-supervised segment discrimination. IEEE Robotics and Automation Letters **7**(2), 2116–2123 (2022) 6
13. Unal, O., Dai, D., Van Gool, L.: Scribble-supervised lidar semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2697–2707 (2022) 4, 5
14. Wang, W., Zhou, T., Yu, F., Dai, J., Konukoglu, E., Van Gool, L.: Exploring cross-image pixel contrast for semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7303–7313 (2021) 3, 7, 8

15. Yang, L., Zhuo, W., Qi, L., Shi, Y., Gao, Y.: St++: Make self-training work better for semi-supervised semantic segmentation. arXiv preprint arXiv:2106.05095 (2021) 2
16. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6023–6032 (2019) 2
17. Zhao, Y., Bai, L., Huang, X.: Fidnet: Lidar point cloud semantic segmentation with fully interpolation decoding. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4453–4458. IEEE (2021) 3
18. 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. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9939–9948 (2021) 1, 3