Supplementary Material: Segment Any Point Cloud Sequences by Distilling Vision Foundation Models

Anonymous Author(s)
Affiliation
Address
email

In this file, we supplement the following materials to support the findings and observations in the main body of this paper:

- Section A elaborates on additional implementation details to facilitate reproduction.
- Section B provides the complete quantitative results of our experiments.
- Section C includes more qualitative results to allow better visual comparisons.
- Section D acknowledges the public resources used during the course of this work.

A Additional Implementation Detail

A.1 Datasets

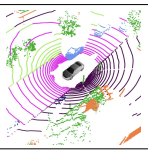
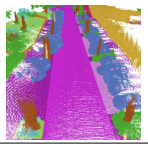
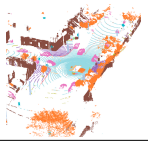
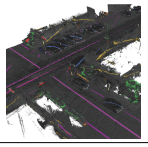
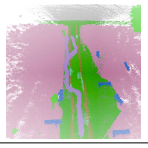
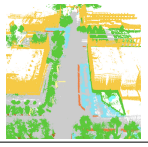
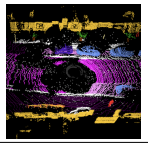
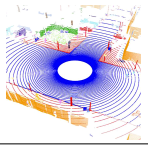
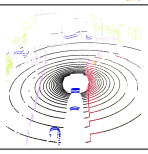
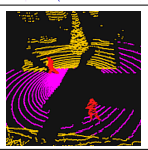
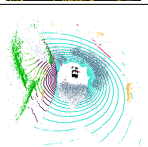
In this work, we conduct extensive experiments from a wide range of point cloud datasets to verify the effectiveness of the proposed ***Seal*** framework. A summary of the detailed configurations and emphases of these datasets is shown in Table A.

- [1]**nuScenes** [7]: The nuScenes[1] dataset offers a substantial number of samples collected by the LiDAR, RADAR, camera, and IMU sensors from Boston and Singapore, allowing machine learning models to learn useful multi-modal features effectively. For the point cloud segmentation task, it consists of 1000 scenes of a total number of 1.1B annotated points, collected by a Velodyne HDL32E LiDAR sensor. It also includes image data from six cameras, which are synchronized with the LiDAR sensor. In this work, we use the LiDAR point clouds and image data from nuScenes for model pretraining. We also conduct detailed fine-tuning experiments to validate the effectiveness of representation learning. More details of this dataset can be found at `https://www.nuscenes.org/nuscenes`.
- [2]**SemanticKITTI** [2]: The SemanticKITTI dataset is a comprehensive dataset designed for semantic scene understanding of LiDAR sequences. This dataset aims to advance the development of algorithms and models for autonomous driving and scene understanding using LiDAR data. It provides 22 densely labeled point cloud sequences that cover urban street scenes, which are captured by a Velodyne HDL-64E LiDAR sensor. In this work, we use the LiDAR point clouds from SemanticKITTI as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `http://semantic-kitti.org/`.
- [3]**Waymo Open** [19]: The Waymo Open dataset is a large-scale collection of real-world autonomous driving data. The 3D semantic segmentation subset contains 1150 scenes,

[1]Here we refer to the *lidarseg* subset of the nuScenes dataset. Know more details about this dataset at the official webpage: `https://www.nuscenes.org/nuscenes`.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

Table A: The sensor configuration and data statistics for the ***eleven*** datasets used in our experiments.

Dataset	Illustration	Sensor Setup	Statistics	Type
nuScenes [7]		1× LiDAR (32-beam) 6× RGB Camera 5× RADAR 1× IMU & GPS	16 semantic classes 29130 training samples 6019 validation samples 6008 testing samples	Real-world Low-resolution point cloud Dense annotation Multi-modality
SemanticKITTI [2]		1× LiDAR (64-beam) 1× Stereo Camera 1× IMU & GPS	19 semantic classes 19130 training samples 4071 validation samples 20351 testing samples	Real-world High-resolution point cloud Dense annotation Multi-modality
Waymo Open [19]		1× LiDAR (64-beam) 5× RGB Camera 1× IMU & GPS	23 semantic classes 23691 training samples 5976 validation samples 2982 testing samples	Real-world High-resolution point cloud Dense annotation Multi-modality
ScribbleKITTI [20]		1× LiDAR (64-beam) 1× Stereo Camera 1× IMU & GPS	19 semantic classes 19130 training samples 4071 validation samples 20351 testing samples	Real-world High-resolution point cloud Sparse annotation Weakly-supervised learning
RELLIS-3D [9]		1× LiDAR (64-beam) 1× LiDAR (32-beam) 1× Stereo Camera 1× IMU & GPS	20 semantic classes 7800 training samples 2413 validation samples 3343 testing samples	Real-world High-resolution point cloud Dense annotation Multi-modality
SemanticPOSS [16]		1× LiDAR (40-beam) 1× RGB Camera 1× IMU & GPS	14 semantic classes 2488 training samples 500 validation samples	Real-world High-resolution point cloud Dense annotation Dynamic instance
SemanticSTF [22]		1× LiDAR (64-beam) 1× LiDAR (32-beam) 1× Stereo Camera 1× RADAR	21 semantic classes 1326 training samples 250 validation samples 500 testing samples	Real-world High-resolution point cloud Dense annotation Adverse weather
SynLiDAR [21]		1× LiDAR (64-beam) 1× simulation suite	32 semantic classes 198396 total samples	Synthetic High-resolution point cloud Dense annotation Transfer learning
Synth4D [17]		1× LiDAR (64-beam) 1× LiDAR (32-beam) 1× simulation suite	22 semantic classes 10000 training samples 10000 validation samples	Synthetic Low-resolution point cloud Dense annotation Transfer learning
DAPS-3D [11]		3× LiDAR (64-beam) 1× simulation suite	4 semantic classes 19061 training samples 3995 validation samples	Semi-synthetic High-resolution point cloud Dense annotation Transfer learning
nuScenes-C [12]		1× LiDAR (32-beam) 6× RGB Camera 5× RADAR 1× IMU & GPS	16 semantic classes 144456 validation samples	Synthetic Low-resolution point cloud Dense annotation Robustness

split into 798 training, 202 validation, and 150 testing scenes. This subset contains 23691 training scans, 5976 validation scans, and 2982 testing scans, respectively, with semantic segmentation labels from 23 classes. The data are captured by five LiDAR sensors: one mid-range LiDAR sensor truncated to a maximum of 75 meters, and four short-range LiDAR sensors truncated to a maximum of 20 meters. In this work, we use the LiDAR point clouds from Waymo Open as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `https://waymo.com/open`.

- [4]**ScribbleKITTI** [20]: The ScribbleKITTI dataset is a recent variant of the SemanticKITTI dataset, with weak supervisions annotated by line scribbles. It shares the exact same amount of training samples with SemanticKITTI, *i.e.*, 19130 scans collected by a Velodyne HDL-64E LiDAR sensor, where the total number of valid semantic labels is 8.06% compared to the fully-supervised version. Annotating the LiDAR point cloud in such a way corresponds to roughly a 90% time-saving. In this work, we use the LiDAR point clouds from ScribbleKITTI as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `https://github.com/ouenal/scribblekitti`.
- [5]**RELLIS-3D** [9]: The RELLIS-3D dataset is a multimodal dataset collected in an off-road environment from the Rellis Campus of Texas A&M University. It consists of 13556 LiDAR scans from 5 traversal sequences. The point-wise annotations are initialized by using the camera-LiDAR calibration to project the more than 6000 image annotations onto the point clouds. In this work, we use the LiDAR point clouds from RELLIS-3D as a downstream task to validate the generalizability of pertaining methods.
- [6]**SemanticPOSS** [16]: The SemanticPOSS dataset is a relatively small-scale point cloud dataset with an emphasis on dynamic instances. It includes 2988 scans collected by a Hesai Pandora LiDAR sensor, which is a 40-channel LiDAR with 0.33 degree vertical resolution, a forward-facing color camera, 4 wide-angle mono cameras covering 360 degrees around the ego-car. The data in this dataset was collected from the campus of Peking University. In this work, we use the LiDAR point clouds from SemanticPOSS as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `http://www.unmannedlab.org/research/RELLIS-3D`.
- [7]**SemanticSTF** [22]: The SemanticSTF dataset is a small-scale collection of 2076 scans, where the data are borrowed from the STF dataset [4]. The scans are collected by a Velodyne HDL64 S3D LiDAR sensor and covered various adverse weather conditions, including 694 snowy, 637 dense-foggy, 631 light-foggy, and 114 rainy scans. The whole dataset is split into three sets: 1326 scans for training, 250 scans for validation, and 500 scans for testing. All three splits have similar proportions of scans from different weather conditions. In this work, we use the LiDAR point clouds from SemanticSTF as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `https://github.com/xiaoaoran/SemanticSTF`.
- [8]**SynLiDAR** [21]: The SynLiDAR dataset contains synthetic point clouds captured from constructed virtual scenes using the Unreal Engine 4 simulator. In total, this dataset contains 13 LiDAR point cloud sequences with 198396 scans. As stated, the virtual scenes in SynLiDAR are constituted by physically accurate object models that are produced by expert modelers with the 3D-Max software. In this work, we use the LiDAR point clouds from SynLiDAR as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `https://github.com/xiaoaoran/SynLiDAR`.
- [9]**Synth4D** [17]: The Synth4D dataset includes two subsets with point clouds captured by simulated Velodyne LiDAR sensors using the CARLA simulator. We use the Synth4D-nuScenes subset in our experiments. It is composed of around 20000 labeled point clouds captured by a virtual vehicle navigating in town, highway, rural area, and city. The label mappings are mapped to that of the nuScenes dataset. In this work, we use the LiDAR point clouds from Synth4D-nuScenes as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `https://github.com/saltoricristiano/gipso-sfouda`.
- [10]**DAPS-3D** [11]: The DAPS-3D dataset consists of two subsets: DAPS-1 and DAPS-2; while the former is a semi-synthetic one with a larger scale, the latter is recorded during a real field trip of the cleaning robot to the territory of the VDNH Park in Moscow. Both subsets are with scans collected by a 64-line Ouster OS0 LiDAR sensor. We use the DAPS-1

subset in our experiments, which contains 11 LiDAR sequences with more than 23000 labeled point clouds. In this work, we use the LiDAR point clouds from DAPS-1 as a downstream task to validate the generalizability of pertaining methods. More details of this dataset can be found at `https://github.com/subake/DAPS3D`.

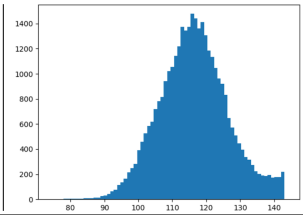
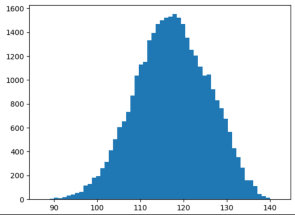
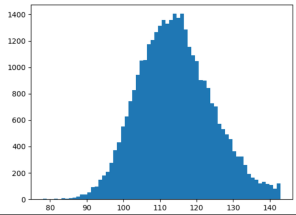
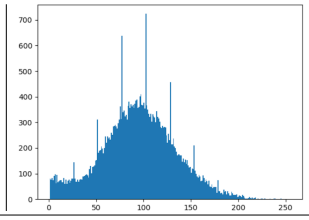
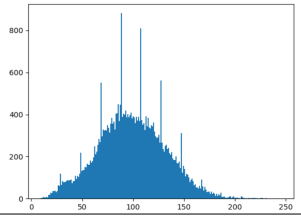
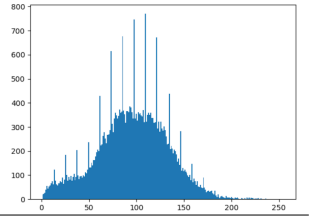
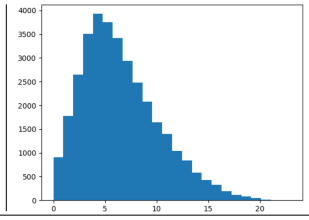
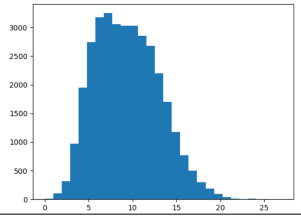
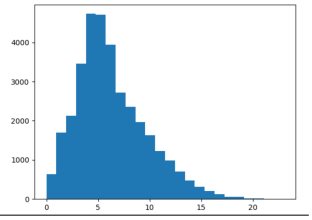
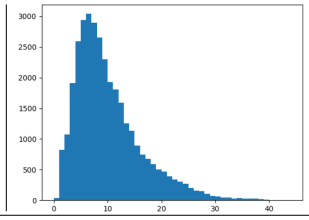
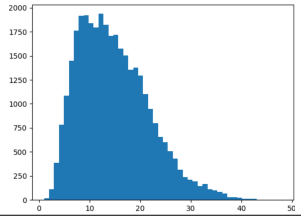
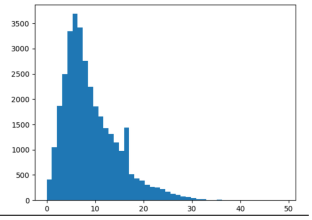
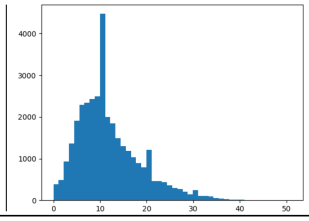
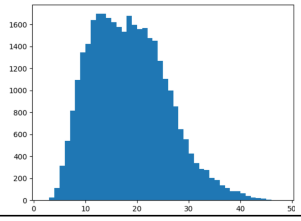
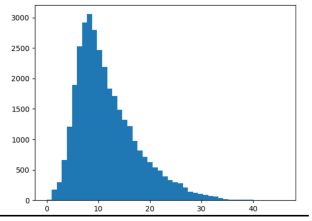
- [11]**nuScenes-C** [12]: The nuScenes-C dataset is one of the corruption sets in the Robo3D benchmark, which is a comprehensive benchmark heading toward probing the robustness of 3D detectors and segmentors under out-of-distribution scenarios against natural corruptions that occur in real-world environments. A total number of eight corruption types, stemming from severe weather conditions, external disturbances, and internal sensor failure, are considered, including 'fog', 'wet ground', 'snow', 'motion blur', 'beam missing', 'crosstalk', 'incomplete echo', and 'cross-sensor' scenarios. These corruptions are simulated with rules constrained by physical principles or engineering experiences. In this work, we use the LiDAR point clouds from nuScenes-C as a downstream task to validate the robustness of pertaining methods under out-of-distribution scenarios. More details of this dataset can be found at `https://github.com/ldkong1205/Robo3D`.

A.2 Vision Foundation Models

In this work, we conduct comprehensive experiments on analyzing the effects brought by different vision foundation models (VFMs), compared to the traditional SLIC [1] method. Some statistical analyses of these different visual partition methods are shown in Table B and Table C.

- **SLIC** [1] (traditional method): The SLIC model, which stands for 'simple linear iterative clustering', is a popular choice for visual partitions of RGB images. It adapts a k-means clustering approach to generate superpixels, in an efficient manner, and offers good unsupervised partition abilities for many downstream tasks. The pursuit of adherence to boundaries and computational and memory efficiency allows SLIC to perform well on different image collections. In this work, we follow SLidR [18] and use SLIC to generate superpixels, with a fixed quota of 150 superpixels per image, and compare our framework with previous ones using SLIC superpixels. More details of this model can be found at `https://github.com/valeoai/SLidR`.
- **SAM** [10]: The Segment Anything Model (SAM) is a recent breakthrough towards zero-shot transferable visual understanding across a wide range of tasks. This model is trained on SA-1B, a large-scale dataset with over 1 billion masks on 11M licensed and privacy-respecting images. As a result, SAM is able to segment images, with either point, box, or mask prompts, across different domains and data distributions. In this work, we use a fixed SAM model with the ViT-H backbone (termed as ViT-H SAM model) to generate superpixels. We use this pretrained model directly without any further fine-tuning. More details of this model can be found at `https://github.com/facebookresearch/segment-anything`.
- **X-Decoder** [24]: The X-Decoder model is a generalized decoding framework that can predict pixel-level segmentation and language tokens seamlessly. This model is pretrained on three types of data, including panoptic segmentation, image-text pairs, and referring segmentation. For the panoptic segmentation task, the model is trained on COCO2017, which includes around 104k images for model training. In this work, we use a fixed X-Decoder model termed as BestSeg-Tiny to generate superpixels. We use this pretrained model directly without any further fine-tuning. More details of this model can be found at `https://github.com/microsoft/X-Decoder`.
- **OpenSeeD** [23]: The OpenSeeD model is designed for open-vocabulary segmentation and detection, which jointly learns from different segmentation and detection datasets. This model consists of an image encoder, a text encoder, and a decoder with foreground, background, and conditioned mask decoding capability. The model is trained on COCO2017 and Objects365, under the tasks of panoptic segmentation and object detection, respectively. In this work, we use a fixed OpenSeeD model termed as openseed-swint-lang to generate superpixels. We use this pretrained model directly without any further fine-tuning. More details of this model can be found at `https://github.com/IDEA-Research/OpenSeeD`.
- **SEEM** [25]: The SEEM model contributes a new universal interactive interface for image segmentation, where 'SEEM' stands for 'segment everything everywhere with multi-modal

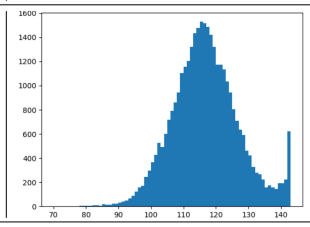
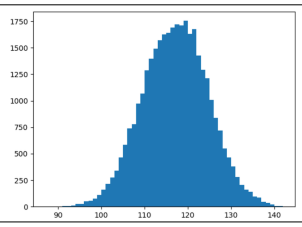
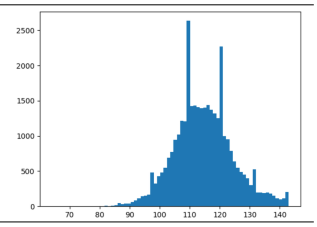
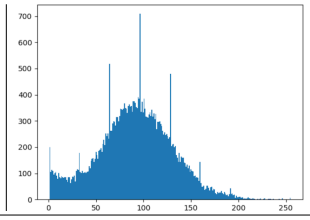
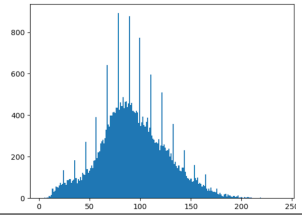
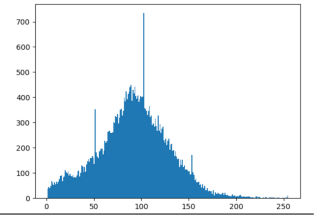
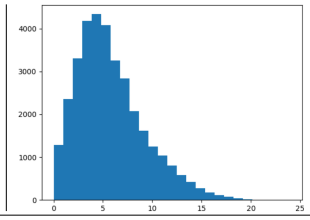
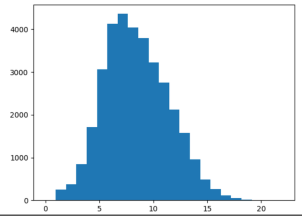
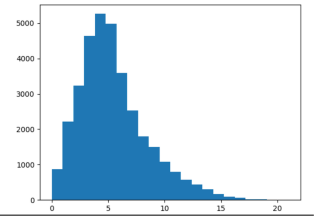
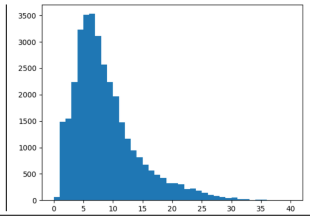
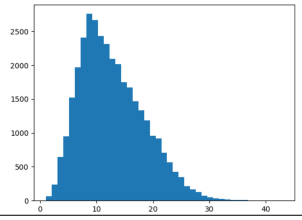
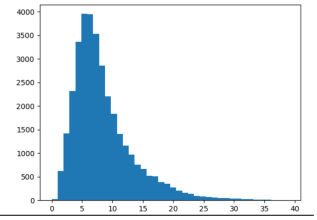
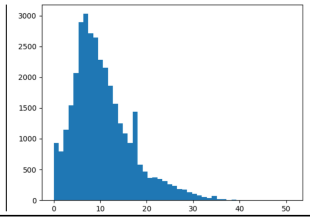
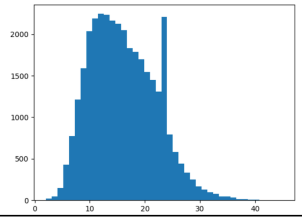
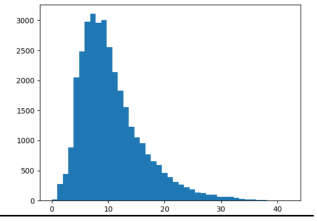
Table B: The **statistics of superpixels** (for front-view cameras) generated by SLIC [1] and different vision foundation modes [10, 24, 23, 25]. The **horizontal axis** denotes the number of superpixels per image. The **vertical axis** denotes the frequency of occurrence.

Method	Front Left	Front	Front Right
SLIC [1]			
SAM [10]			
X-Decoder [24]			
OpenSeeD [23]			
SEEM [25]			

prompts all at once'. The newly designed prompting scheme can encode various user intents into prompts in a joint visual-semantic space, which possesses properties of versatility, compositionality, interactivity, and semantic awareness. As such, this model is able to generalize well to different image datasets, under a zero-shot transfer manner. Similar to X-Decoder, SEEM is trained on COCO2017, with a total number of 107k images used during model training. In this work, we use a fixed SEEM model termed as SEEM-oq101 to generate superpixels. We use this pretrained model directly without any further fine-tuning. More details of this model can be found at `https://github.com/UX-Decoder/Segment-Everything-Everywhere-All-At-Once`.

The quality of superpixels directly affects the performance of self-supervised representation learning. Different from the previous paradigm [18, 15], our ***Seal*** framework resorts to the recent VFMs for generating superpixels. Compared to the traditional SLIC method, these VFMs are able to generate semantically-aware superpixels that represent coherent semantic meanings of objects and backgrounds around the ego-vehicle.

Table C: The **statistics of superpixels** (for back-view cameras) generated by SLIC [1] and different vision foundation modes [10, 24, 23, 25]. The **horizontal axis** denotes the number of superpixels per image. The **vertical axis** denotes the frequency of occurrence.

Method	Back Left	Back	Back Right
SLIC [1]			
SAM [10]			
X-Decoder [24]			
OpenSeeD [23]			
SEEM [25]			

As been verified in our experiments, these semantic superpixels have the ability to ease the "over-segment" problem in current self-supervised learning frameworks [18, 15] and further improve the performance for both linear probing and downstream fine-tuning.

The histograms shown in Table B and Table C verify that the number of superpixels per image of VFMs is much smaller than that of SLIC. This brings two notable advantages: *i)* Since semantically similar objects and backgrounds are grouped together in semantic superpixels, the "self-conflict" problem in existing approaches is largely mitigated, which directly boosts the quality of representation learning. *ii)* Since the embedding length D of the superpixel embedding features $\mathbf{Q} \in \mathbb{R}^{M \times D}$ and superpoint embedding features $\mathbf{K} \in \mathbb{R}^{M \times D}$ directly relates to computation overhead, a reduction (*e.g.*, around 150 superpixels per image in SLIC [1] and around 25 superpixels per image in X-Decoder [24], OpenSeeD [23], and SEEM [25]) on D would allow us to train the segmentation models in a more efficient manner.

Some typical examples of our generated superpixels and their corresponding superpoints are shown in Fig. A, Fig. B, Fig. C, and Fig. D. As will be shown in Section B, our use of semantic superpixels

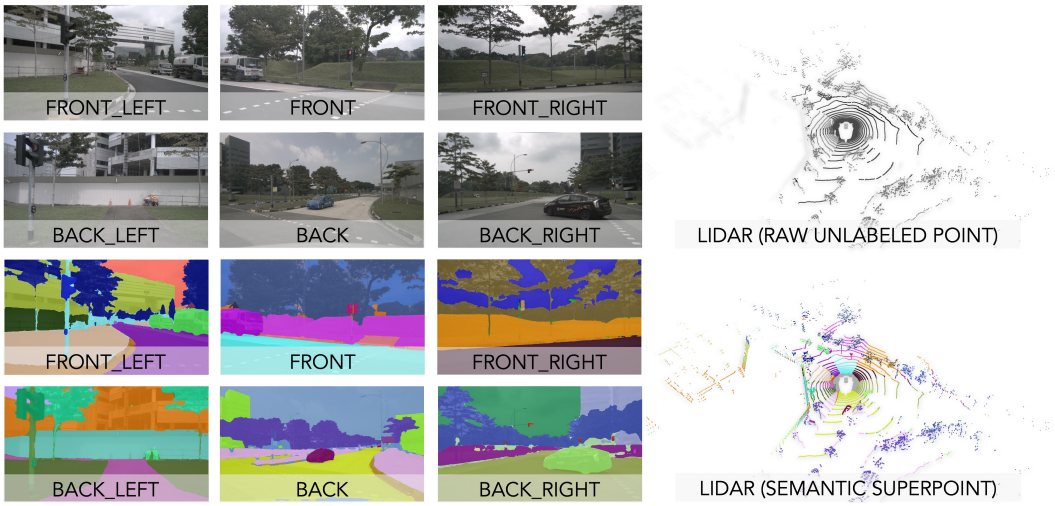


Figure A: Illustration of the **semantic superpixel-to-superpoint transformation** in the proposed *Seal* framework. **[Row 1 & 2]** The raw data captured by the multi-view camera and LiDAR sensors. **[Row 3 & 4]** The semantic superpixel on the camera images and superpoint formed by projecting superpixel into the point cloud via camera-LiDAR correspondence. The superpixels are generated using SEEM [25]. Each color represents one distinct segment. Best viewed in color.

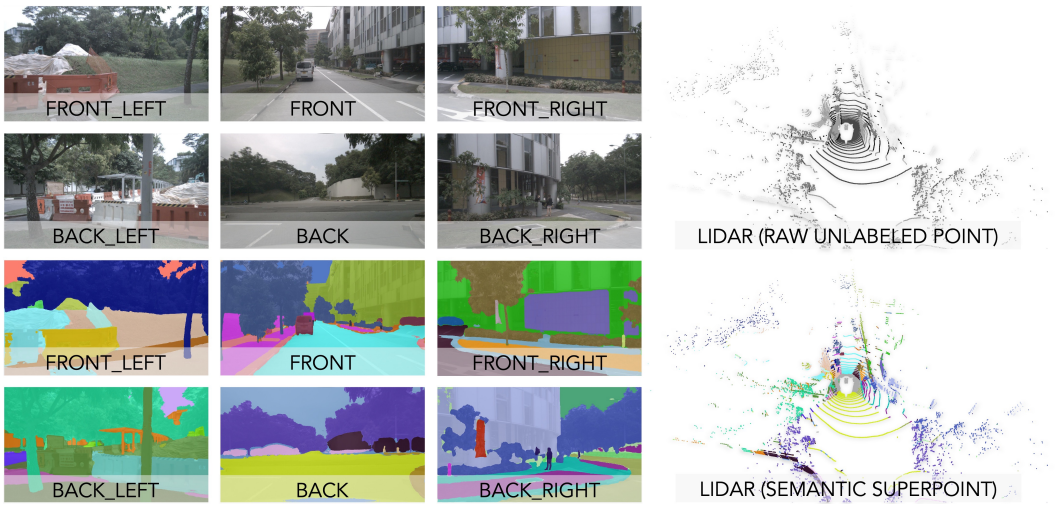


Figure B: Illustration of the **semantic superpixel-to-superpoint transformation** in the proposed *Seal* framework. **[Row 1 & 2]** The raw data captured by the multi-view camera and LiDAR sensors. **[Row 3 & 4]** The semantic superpixel on the camera images and superpoint formed by projecting superpixel into the point cloud via camera-LiDAR correspondence. The superpixels are generated using SEEM [25]. Each color represents one distinct segment. Best viewed in color.

generated by VFMs brings not only performance gains but also a much faster convergence rate during the model pretraining stage.

A.3 Implementation Details

A.3.1 Data Split

For **model pertaining**, we follow the SLidR protocol [18] in data splitting. Specifically, the nuScenes [7] dataset consists of 700 training scenes in total, 100 of which are kept aside, which constitute the SLidR mini-val split. All models are pretrained using all the scans from the 600 remaining training scenes. The 100 scans in the mini-val split are used to find the best-possible hyperparameters. The

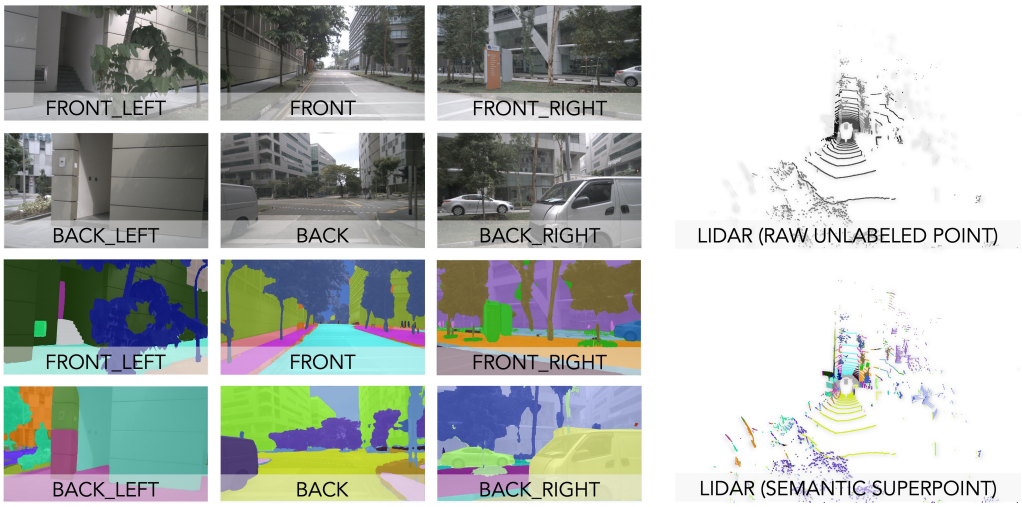


Figure C: Illustration of the **semantic superpixel-to-superpoint transformation** in the proposed *Seal* framework. **[Row 1 & 2]** The raw data captured by the multi-view camera and LiDAR sensors. **[Row 3 & 4]** The semantic superpixel on the camera images and superpoint formed by projecting superpixel into the point cloud via camera-LiDAR correspondence. The superpixels are generated using SEEM [25]. Each color represents one distinct segment. Best viewed in color.

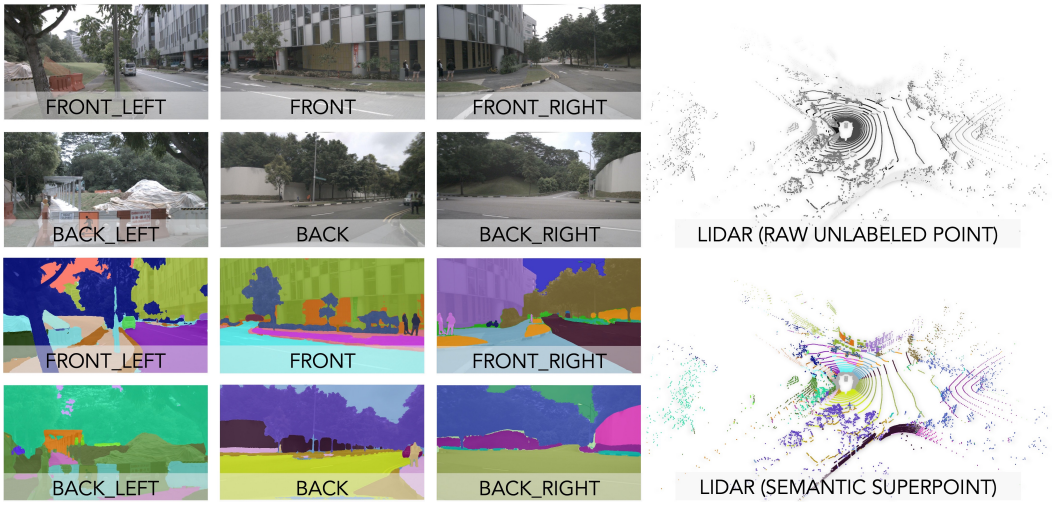


Figure D: Illustration of the **semantic superpixel-to-superpoint transformation** in the proposed *Seal* framework. **[Row 1 & 2]** The raw data captured by the multi-view camera and LiDAR sensors. **[Row 3 & 4]** The semantic superpixel on the camera images and superpoint formed by projecting superpixel into the point cloud via camera-LiDAR correspondence. The superpixels are generated using SEEM [25]. Each color represents one distinct segment. Best viewed in color.

trained models are then validated on the official nuScenes validation set, without any kind of test-time augmentation or model ensemble. This is to ensure a fair comparison with previous works and also in line with the practical requirements.

For **linear probing**, the pretrained 3D network F_{θ_p} is frozen with a trainable point-wise linear classification head which is trained for 50 epochs on a A100 GPU with a learning rate of 0.05, and batch size is 16 on the nuScenes train set for all methods.

For **downstream fine-tuning** tasks, we stick with the common practice in SLidR [18] whenever possible. The detailed data split strategies are summarized as follows.

- For fine-tuning on **nuScenes** [7], we follow the SLidR protocol to split the train set of nuScenes to generate 1%, 5%, 10%, 25%, and 100% annotated scans for the training subset.
- For fine-tuning on **DAPS-3D** [11], we take sequences 38-18_7_72_90 as the training set and 38-18_7_72_90, 42-48_10_78_90, and 44-18_11_15_32 as the validation set.
- For fine-tuning on **SynLiDAR** [21], we use the sub-set which is a uniformly downsampled collection from the whole dataset.
- For fine-tuning on **SemanticPOSS** [16], we use sequences 00 and 01 as *half* of the annotated training scans and use sequences 00 to 05, except 02 for validation to create *full* of the annotated training samples.
- For fine-tuning on **SemanticKITTI** [2], **Waymo Open** [19], **ScribbleKITTI** [20], **RELLIS-3D** [9], **SemanticSTF** [22], and **Synth4D** [17], we follow the SLidR protocol to create 1%, 10%, *half*, or *full* split of the annotated training scans, *e.g.*, one scan is taken every 100 frame from the training set to get 1% of the labeled training samples. Notably, the point cloud segmentation performance in terms of IoU is reported on the official validation sets for all the above-mentioned datasets.

A.3.2 Experimental Setup

In our experiments, we fine-tune the entire 3D network on the semantic segmentation task using a linear combination of the cross-entropy loss and the Lovász-Softmax loss [3] as training objectives on a single A100 GPU. For the few-shot semantic segmentation tasks, the 3D networks are fine-tuned for 100 epochs with a batch size of 10 for the SemanticKITTI [2], Waymo Open [19], ScribbleKITTI [20], RELLIS-3D [9], SemanticSTF [22], SemanticPOSS [16], DAPS-3D [11], SynLiDAR [21], and Synth4D [17] datasets.

For the nuScenes [7] dataset, we fine-tune the 3D network for 100 epochs with a batch size of 16 while training on the 1% annotated scans. The 3D network train on the other portions of nuScenes is fine-tuned for 50 epochs with a batch size of 16. We adopt different learning rates on the 3D backbone F_{θ_p} and the classification head, except for the case that F_{θ_p} is randomly initialized. The learning rate of F_{θ_p} is set as 0.05 and the learning rate of the classification head is set as 2.0, respectively, for all the above-mentioned datasets except nuScenes. On the nuScenes dataset, the learning rate of F_{θ_p} is set as 0.02.

We train our framework using the SGD optimizer with a momentum of 0.9, a weight decay of 0.0001, and a dampening ratio of 0.1. The cosine annealing learning rate strategy is adopted which decreases the learning rate from its initial value to zero at the end of the training.

A.3.3 Data Augmentation

For the **model pretraining**, we apply two sets of data augmentations on the point cloud and the multi-view image, respectively, and update the point-pixel correspondences after each augmentation by following SLidR [18].

- Regarding the point cloud, we adopt a random rotation around the z-axis and flip the x-axis or y-axis with a 50% probability. Besides, we randomly drop the cuboid where the length of each side covers no more than 10% range of point coordinates on the corresponding axis, and the cuboid center is located on a randomly chosen point in the point cloud. We also ensure that the dropped cuboid retains at least 1024 pairs of points and pixels; otherwise, we select another new cuboid instead. For the temporal frame, we apply the same data augmentation to it.
- Regarding the multi-view image, we apply a horizontal flip with a 50% probability and a cropped resize which reshapes the image to 416×224. Before resizing, the random crop fills at least 30% of the available image space with a random aspect ratio between $14:9$ and $17:9$. If this random cropping does not preserve at least 1024 or 75% of the pixel-point pairs, a different crop is chosen.

For the **downstream fine-tuning** tasks, we apply a random rotation around the z-axis and flip the x-axis or y-axis with a 50% probability for all the points in the point cloud. If the results are mentioned with LaserMix [13] augmentation, we augment the point clouds via the LaserMix before the above

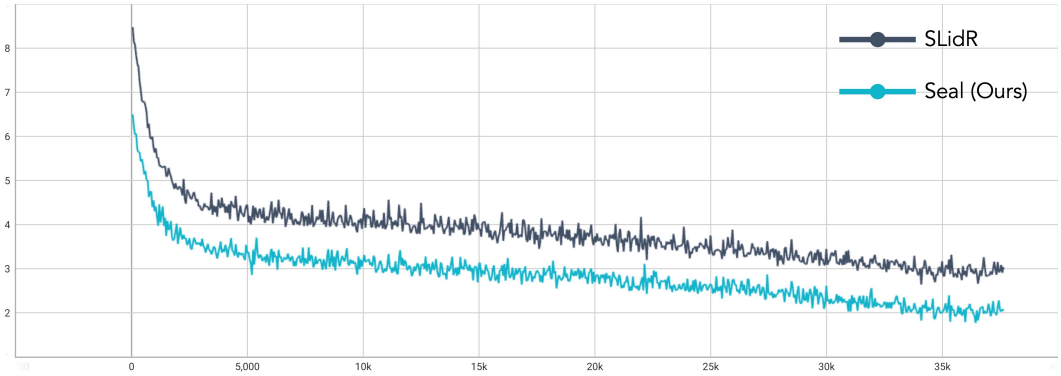


Figure E: The convergence rate comparison between SLidR [18] and the proposed ***Seal*** framework.

data augmentations. We report both results in the main paper so that we can fairly compare the point cloud segmentation performance with previous works.

A.3.4 Model Configuration

For the **model pretraining**, the 3D backbone F_{θ_p} is a Minkowski U-Net [6] with $3 \times 3 \times 3$ kernels; while the 2D image encoder G_{θ_i} is a ResNet-50 [8] initialized with 2D self-supervised pretrained model of MoCoV2 [5]. These configurations are kept the same as SLidR [18]. The channel dimension of G_{θ_i} head and F_{θ_p} head is set to 64. For the **linear probing** task, we adopt a linear classification head, as mentioned in previous sections. For the **downstream fine-tuning** tasks, the same 3D backbone F_{θ_p}, *i.e.*, the Minkowski U-Net [6] with $3 \times 3 \times 3$ kernels, is used.

B Additional Quantitative Result

B.1 Self-Supervised Learning

We report the complete results (*i.e.*, the class-wise IoU scores) for the **linear probing** and **downstream fine-tuning** tasks shown in the main paper. We report the official results from previous works whenever possible. We also report our reproduced results for random initialization, PPKT [14], and SLidR [18]. Specifically, the complete results on the nuScenes [7], SemanticKITTI [2], Waymo Open [19], and Synth4D [17] datasets are shown in Table D, Table E, Table F, and Table G, respectively. We observe that ***Seal*** constantly outperforms prior methods for most semantic classes on all datasets.

We also compare the convergence rate of ***Seal*** with SLidR [18] in Fig. E. As can be seen, our framework is able to converge faster with the use of semantic superpixels, where these higher qualitative contrastive samples gradually form a much more coherent optimization landscape.

B.2 Robustness Probing

We report the complete results of the robustness evaluation experiments in the main paper, including the per-corruption CE scores and the per-corruption RR scores. Specifically, the complete results on the nuScenes-C [12] dataset are shown in Table H and Table I. We observe that ***Seal*** is superior to previous methods in terms of both CE and RR metrics.

C Additional Qualitative Result

C.1 Cosine Similarity

We provide more examples for the cosine similarity study in Fig. F, Fig. G, and Fig. H.

C.2 Downstream Tasks

We provide more qualitative results for downstream fine-tuning tasks in Fig. I, Fig. J, and Fig. K.

Table D: The **per-class IoU scores** of different pretraining methods pretrained on *nuScenes* [7] and linear probed or fine-tuned on different proportions (1%, 5%, 10%, 25%, and Full) of the *nuScenes* [7] data. Symbol ¶ denotes our reproduced results and the remaining are reported scores. All IoU scores are given in percentage (%). The **best** mIoU score is highlighted in **bold**.

Method	mIoU	barrier	bicycle	bus	car	const. veh.	motorcycle	pedestrian	traffic cone	trailer	truck	driv. surf.	other flat	sidewalk	terrain	manmade	vegetation
Linear Probing																	
Random ¶	8.4	0.5	0.0	0.0	3.9	0.0	0.0	0.0	6.4	0.0	3.9	60.4	0.0	0.1	16.2	30.6	12.2
PointContrast	21.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DepthContrast	22.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PPKT	35.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SLidR ¶	39.2	44.2	0.0	30.8	60.2	15.1	22.4	47.2	27.7	16.3	34.3	80.6	21.8	35.2	48.1	71.0	71.9
ST-SLidR	40.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Seal (Ours)	**45.0**	54.7	5.9	3.6	61.7	18.9	28.8	48.1	31.0	22.1	39.5	83.8	35.4	46.7	56.9	74.7	74.7
Fine-tuning (1%)																	
Random	30.3	0.0	0.0	8.1	65.0	0.1	6.6	21.0	9.0	9.3	25.8	89.5	14.8	41.7	48.7	72.4	73.3
PointContrast	32.5	0.0	1.0	5.6	67.4	0.0	3.3	31.6	5.6	12.1	30.8	91.7	21.9	48.4	50.8	75.0	74.6
DepthContrast	31.7	0.0	0.6	6.5	64.7	0.2	5.1	29.0	9.5	12.1	29.9	90.3	17.8	44.4	49.5	73.5	74.0
PPKT	37.8	0.0	2.2	20.7	75.4	1.2	13.2	45.6	8.5	17.5	38.4	92.5	19.2	52.3	56.8	80.1	80.9
SLidR	38.8	0.0	1.8	15.4	73.1	1.9	19.9	47.2	17.1	14.5	34.5	92.0	27.1	53.6	61.0	79.8	82.3
ST-SLidR	40.8	0.0	2.7	16.0	74.5	3.2	25.4	50.9	20.0	17.7	40.2	92.0	30.7	54.2	61.1	80.5	82.9
Seal (Ours)	**45.8**	0.0	9.4	32.6	77.5	10.4	28.0	53.0	25.0	30.9	49.7	94.0	33.7	60.1	59.6	83.9	83.4
Fine-tuning (5%)																	
Random	47.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Random ¶	44.5	50.1	2.9	57.3	70.3	1.1	6.1	39.1	18.3	17.3	44.8	92.3	38.6	54.9	61.1	80.3	77.9
PPKT ¶	52.7	56.1	8.2	65.3	79.0	9.1	15.5	54.3	34.5	26.7	58.6	93.2	44.1	63.1	64.8	85.1	83.6
SLidR	52.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ST-SLidR	54.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Seal (Ours)	**55.6**	61.0	7.4	70.4	82.4	11.9	30.4	59.2	34.0	33.6	61.1	94.7	46.2	63.4	63.9	85.7	84.9
Fine-tuning (10%)																	
Random	56.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Random ¶	53.2	56.8	5.2	66.3	74.5	5.7	36.1	49.5	38.2	29.2	54.4	94.0	47.7	61.4	67.6	82.8	81.1
PPKT ¶	60.3	64.0	12.0	67.8	77.6	16.0	56.8	63.3	49.8	28.3	56.3	94.1	62.7	66.4	68.7	85.9	85.4
SLidR	59.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ST-SLidR	60.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Seal (Ours)	**63.0**	64.9	15.1	78.9	83.5	22.5	61.0	63.0	51.5	37.4	65.2	95.2	59.4	67.7	69.6	86.2	86.4
Fine-tuning (25%)																	
Random	65.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Random ¶	63.0	64.9	15.1	78.9	83.5	22.5	61.0	63.0	51.5	37.4	65.2	95.2	59.4	67.7	69.6	86.2	86.4
PPKT ¶	67.1	63.7	12.1	87.4	85.2	42.0	61.2	69.6	54.7	50.1	74.9	95.8	64.6	69.9	70.4	87.4	85.1
SLidR	66.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ST-SLidR	67.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Seal (Ours)	**68.4**	67.4	15.5	90.5	85.1	40.0	62.3	68.1	58.3	54.5	76.0	95.8	65.6	69.8	71.0	87.9	86.7
Fine-tuning (Full)																	
Random	74.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	
Random ¶	74.9	77.4	34.4	90.4	86.5	51.1	78.4	77.0	64.9	67.8	77.7	96.6	72.2	74.3	73.2	89.6	86.7
PPKT ¶	74.5	75.4	37.4	91.7	86.1	45.3	77.3	75.6	65.1	65.0	79.9	96.6	72.1	74.6	73.4	89.2	86.6
SLidR	74.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ST-SLidR	75.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Seal (Ours)	**75.6**	76.7	26.0	93.4	86.1	55.8	82.7	77.3	66.0	67.9	83.9	96.5	73.6	74.3	73.2	89.3	87.1

Table E: The **per-class IoU scores** of different pretraining methods pretrained on *nuScenes* [7] and fine-tuned on 1% of the *SemanticKITTI* [2] data. Symbol ¶ denotes our reproduced results and the remaining are reported scores. All IoU scores are given in percentage (%). The **best** mIoU score is highlighted in **bold**.

Method	mIoU	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	trunk	trunk	trunk
Random	39.5	91.2	0.0	9.4	8.0	10.7	21.2	0.0	0.0	89.4	21.4	73.0	1.1	85.3	41.1	84.9	50.1	71.4	55.4	37.6
PPKT	43.9	91.3	1.9	11.2	23.1	12.1	27.4	37.3	0.0	91.3	27.0	74.6	0.3	86.5	38.2	85.3	58.2	71.6	57.7	40.1
SLidR	44.6	92.2	3.0	17.0	22.4	14.3	36.0	22.1	0.0	91.3	30.0	74.7	0.2	87.7	41.2	85.0	58.5	70.4	58.3	42.4
ST-SLidR	44.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Seal	**46.6**	92.3	14.9	18.7	16.1	23.7	43.0	34.4	0.0	91.3	27.2	75.3	0.7	85.7	38.8	85.1	61.9	71.3	57.7	47.7

Table F: The **per-class IoU scores** of different pretraining methods pretrained on *nuScenes* [7] and fine-tuned on 1% of the *Waymo Open* [19] data. Symbol ¶ denotes our reproduced results and the remaining are reported scores. All IoU scores are given in percentage (%). The **best** mIoU score is highlighted in **bold**.

Method	mIoU	car	truck	bus	other vehicle	motorcyclist	bicyclist	pedestrian	sign	traffic light	pole	construction	bicycle	motorcycle	building	vegetation	tree trunk	curb	road	lane marker	other ground	walkable	sidewalk
Random	39.4	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PPKT ¶	47.6	92.3	48.8	34.8	6.9	0.0	22.0	73.0	58.3	17.6	55.8	20.4	25.5	7.8	91.5	87.1	53.8	55.1	89.1	38.6	34.7	71.4	62.6
SLidR	47.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ST-SLidR	44.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Seal	**49.3**	92.5	52.6	33.5	3.7	0.0	31.9	73.1	61.0	23.5	57.0	31.4	20.1	12.4	91.4	87.1	53.2	57.2	89.5	38.8	34.9	72.1	68.7

Table G: The **per-class IoU scores** of different pretraining methods pretrained on *nuScenes* [7] and fine-tuned on 1% of the *Synth4D* [17] data. Symbol ¶ denotes our reproduced results and the remaining are reported scores. All IoU scores are given in percentage (%). The **best** mIoU score is highlighted in **bold**.

Method	mIoU	building	fences	other	pedestrain	pole	roadlines	road	sidewalk	vegetation	vehicle	wall	traffic sign	sky	ground	bridge	rail track	guardrail	traffic light	static	dynamic	water	terrain
Random	20.2	33.2	13.8	0.0	16.3	13.9	0.0	88.6	51.0	48.9	95.7	27.6	0.0	0.0	0.0	0.0	0.1	19.4	0.0	0.2	0.0	0.0	36.2
PPKT ¶	61.1	84.4	67.1	62.9	77.4	75.0	2.2	92.0	76.3	92.5	99.2	73.4	67.3	0.0	62.2	39.6	83.5	66.8	0.0	63.6	55.3	37.4	66.0
SLidR ¶	63.1	83.7	66.3	64.9	77.4	76.4	6.5	92.8	78.7	92.5	99.0	72.5	64.2	0.0	74.3	48.9	85.3	67.1	0.0	67.1	60.0	47.1	63.6
Seal	**64.5**	84.8	70.5	64.8	80.3	76.3	9.3	92.9	79.8	92.7	98.9	73.0	60.7	0.0	75.2	55.3	84.6	67.0	0.0	68.2	60.7	53.2	70.9

Table H: The **Corruption Error (CE) scores** of different pretraining methods pretrained on *nuScenes* [7] and probed under the eight out-of-distribution corruptions in the *nuScenes-C* dataset from the Robo3D benchmark [12]. All CE scores are given in percentage (%). The **best** CE score for each corruption type is highlighted in **bold**.

	Initial	Backbone	mCE ↓	Fog	Wet	Snow	Move	Beam	Cross	Echo	Sensor
LP	PPKT	MinkUNet_{18}	183.44	149.59	247.53	120.50	266.03	213.54	109.62	199.03	161.65
	SLidR	MinkUNet_{18}	179.38	140.47	237.29	**112.93**	276.44	210.65	**107.86**	189.27	160.16
	Seal (Ours)	MinkUNet_{18}	**166.18**	**135.18**	**219.36**	117.47	**234.01**	**189.70**	108.54	**172.16**	**153.03**
Full	Random	PolarNet	115.09	90.10	115.33	58.98	208.19	121.07	80.67	128.17	118.23
	Random	FIDNet	122.42	75.93	122.58	68.78	192.03	164.84	57.95	141.66	155.56
	Random	CENet	112.79	71.16	115.48	64.31	156.67	159.03	**53.27**	129.08	153.35
	Random	WaffleIron	106.73	94.76	99.92	84.51	152.35	110.65	91.09	106.41	114.15
	Random	Cylinder3D	105.56	83.22	111.08	69.74	165.28	113.95	74.42	110.67	116.15
	Random	SPVCNN_{18}	106.65	88.42	105.56	98.78	156.48	110.11	86.04	104.26	103.55
	Random	SPVCNN_{34}	97.45	95.21	99.50	97.32	**95.34**	**98.73**	97.92	**96.88**	**98.74**
	Random	MinkUNet_{18}	112.20	79.90	112.50	74.64	181.47	120.76	93.22	111.58	123.53
	PPKT	MinkUNet_{18}	105.64	77.63	104.22	68.60	160.95	114.81	86.71	108.96	123.20
	SLidR	MinkUNet_{18}	106.08	74.61	106.13	73.75	165.09	118.02	79.08	107.38	124.57
	Seal (Ours)	MinkUNet_{18}	**92.63**	**58.97**	**98.47**	**56.63**	127.25	108.20	57.97	110.95	122.63

Table I: The **Resilience Rate (RR) scores** of different pretraining methods pretrained on *nuScenes* [7] and probed under the eight out-of-distribution corruptions in the *nuScenes-C* dataset from the Robo3D benchmark [12]. All RR scores are given in percentage (%). The **best** RR score for each corruption type is highlighted in **bold**.

	Initial	Backbone	mRR ↑	Fog	Wet	Snow	Move	Beam	Cross	Echo	Sensor
LP	PPKT	MinkUNet_{18}	**78.15**	85.38	**98.66**	78.33	81.36	**91.42**	**54.37**	78.02	**57.69**
	SLidR	MinkUNet_{18}	77.18	**89.90**	98.17	**84.12**	68.14	86.93	53.63	81.29	55.26
	Seal (Ours)	MinkUNet_{18}	75.38	83.05	95.15	66.59	**83.94**	89.70	45.18	**83.94**	55.48
Full	Random	PolarNet	76.34	81.59	97.95	**90.82**	62.49	86.75	57.12	75.16	58.86
	Random	FIDNet	73.33	90.78	95.29	82.61	68.51	67.44	**80.48**	68.31	33.20
	Random	CENet	76.04	91.44	95.35	84.12	79.57	68.19	83.09	72.75	33.82
	Random	WaffleIron	72.78	73.71	97.19	65.19	78.16	85.70	43.54	80.86	57.85
	Random	Cylinder3D	78.08	83.52	96.57	79.41	76.18	87.23	61.68	81.55	58.51
	Random	SPVCNN_{18}	74.70	79.31	97.39	55.22	78.44	87.85	49.50	83.72	66.14
	Random	SPVCNN_{34}	75.10	72.95	96.70	54.79	**97.47**	**90.04**	36.71	**84.84**	**67.35**
	Random	MinkUNet_{18}	72.57	84.33	94.63	74.31	69.26	83.06	42.27	79.88	52.79
	PPKT	MinkUNet_{18}	76.06	85.90	97.71	79.28	76.72	85.72	48.77	81.31	53.10
	SLidR	MinkUNet_{18}	75.99	87.46	96.68	74.89	74.97	84.06	56.08	81.78	52.01
	Seal (Ours)	MinkUNet_{18}	**83.08**	**96.11**	**98.29**	87.59	87.49	87.25	75.98	79.19	52.71

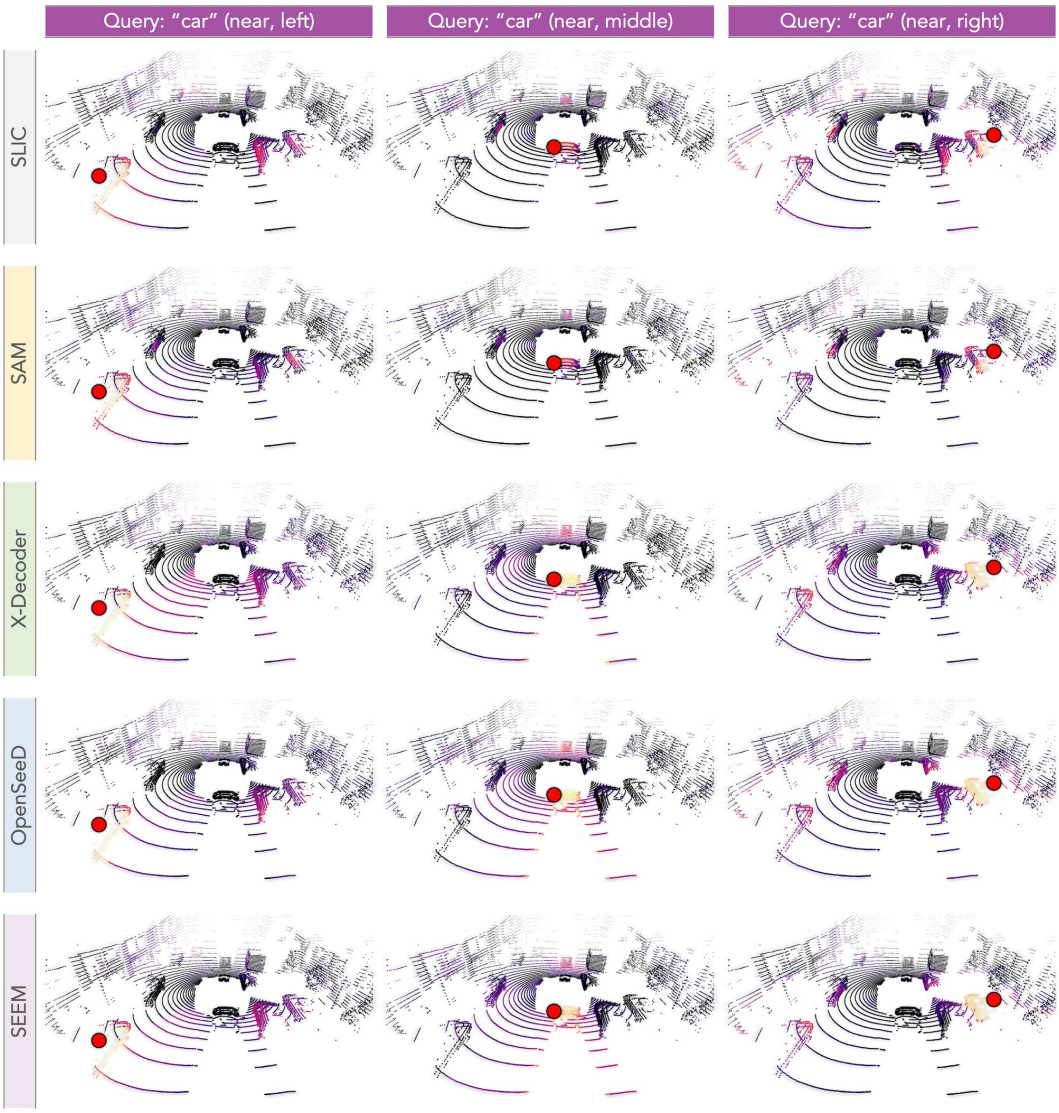


Figure F: The **cosine similarity** between the query point (denoted as the **red dot**) and the feature learned with SLIC [1] and different VFMs [10, 24, 23, 25]. The color goes from **violet** to **yellow** denoting **low** and **high** similarity scores, respectively. Best viewed in color.

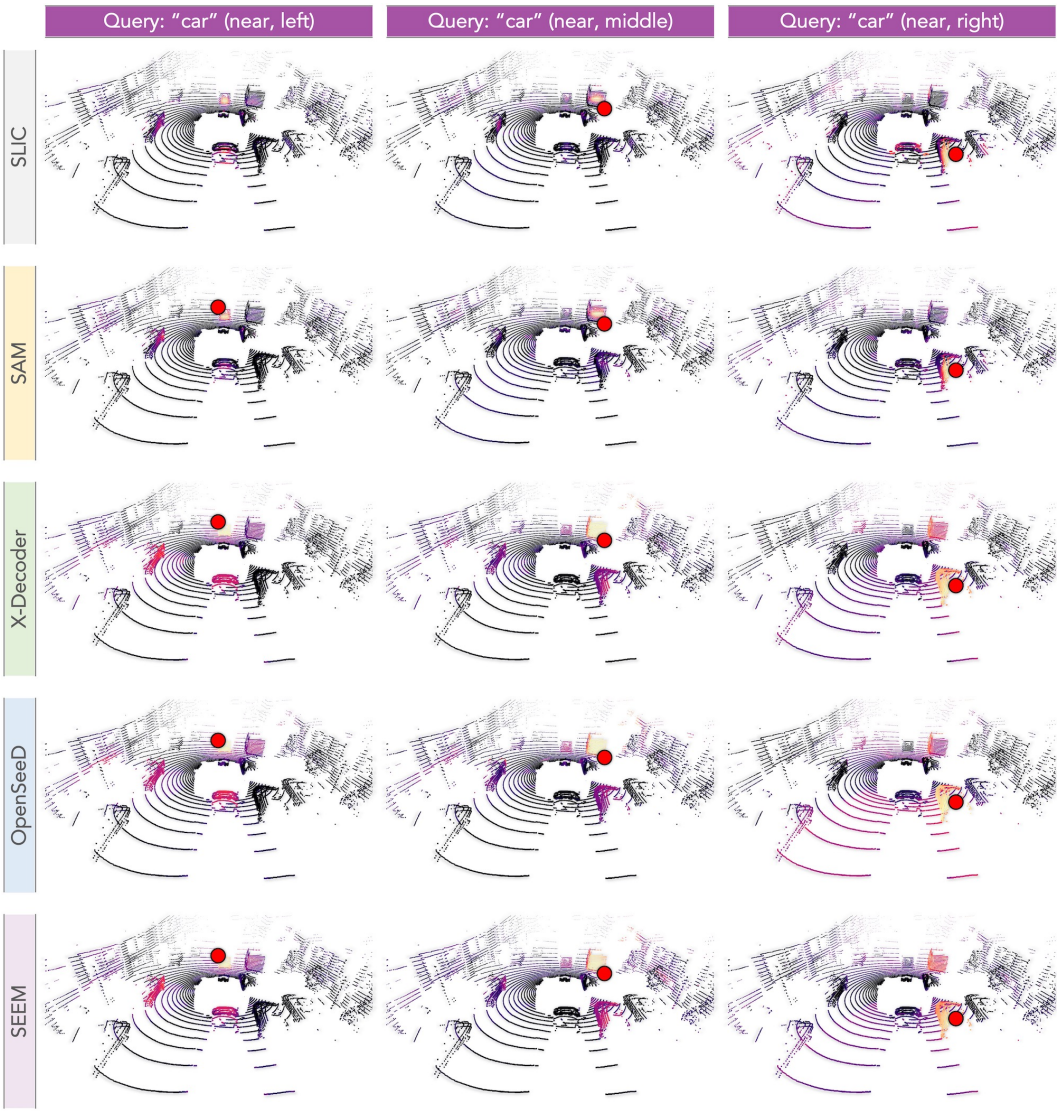


Figure G: The **cosine similarity** between the query point (denoted as the **red dot**) and the feature learned with SLIC [1] and different VFMs [10, 24, 23, 25]. The color goes from **violet** to **yellow** denoting **low** and **high** similarity scores, respectively. Best viewed in color.

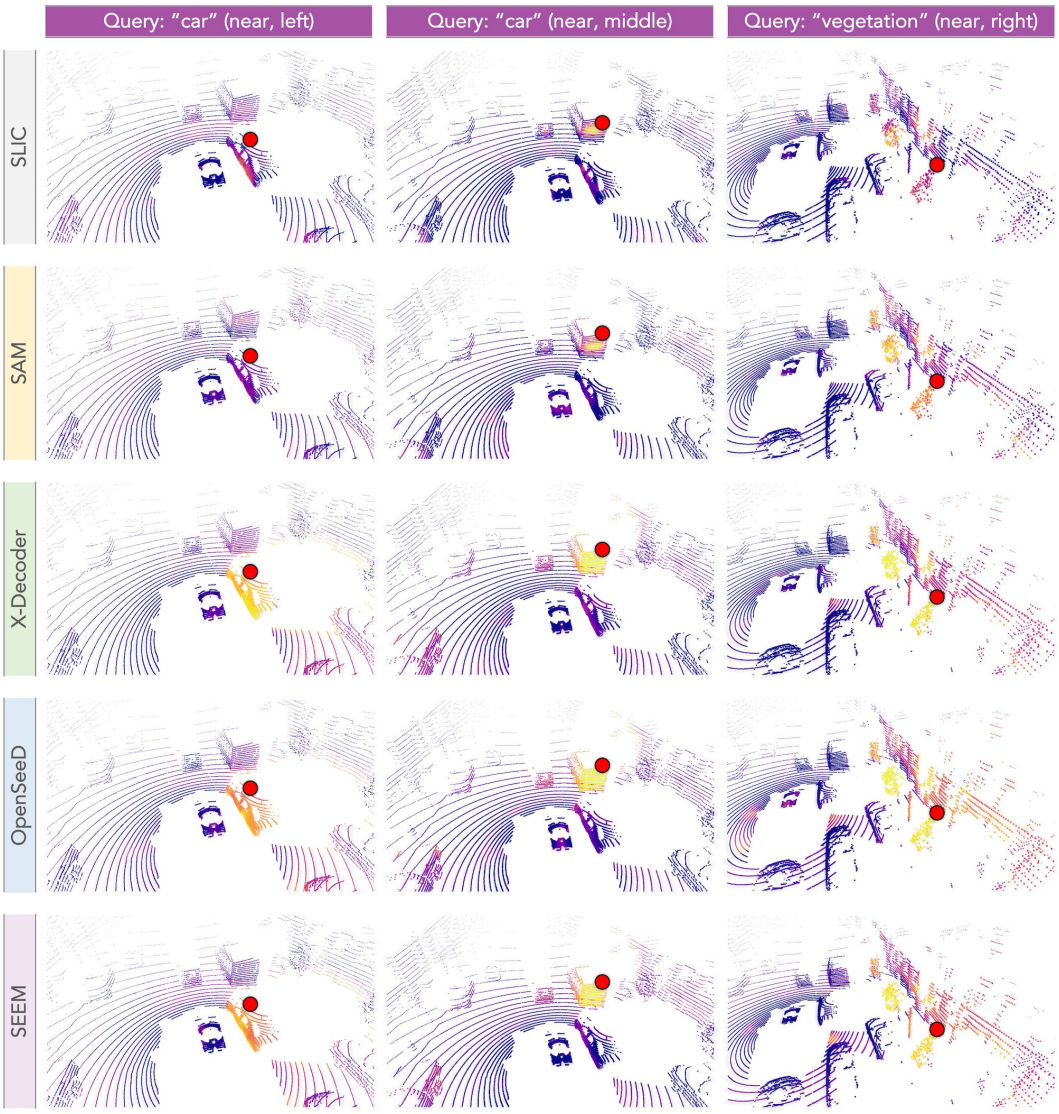


Figure H: The **cosine similarity** between the query point (denoted as the **red dot**) and the feature learned with SLIC [1] and different VFMs [10, 24, 23, 25]. The color goes from **violet** to **yellow** denoting **low** and **high** similarity scores, respectively. Best viewed in color.

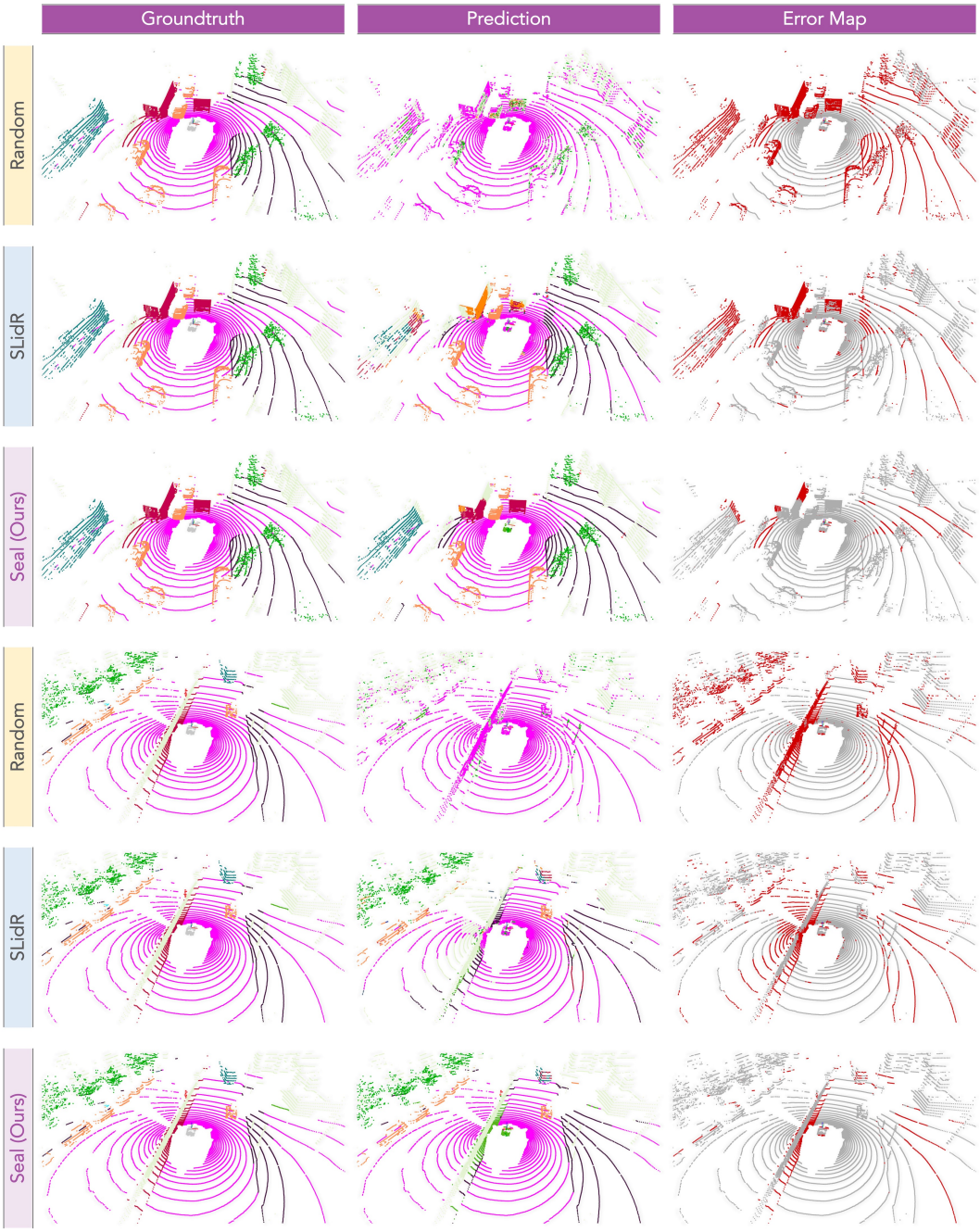


Figure I: The **qualitative results** of different point cloud pretraining approaches pretrained on the raw data of *nuScenes* [7] and fine-tuned with 1% labeled data. To highlight the differences, the **correct** / **incorrect** predictions are painted in **gray** / **red**, respectively. Best viewed in color.

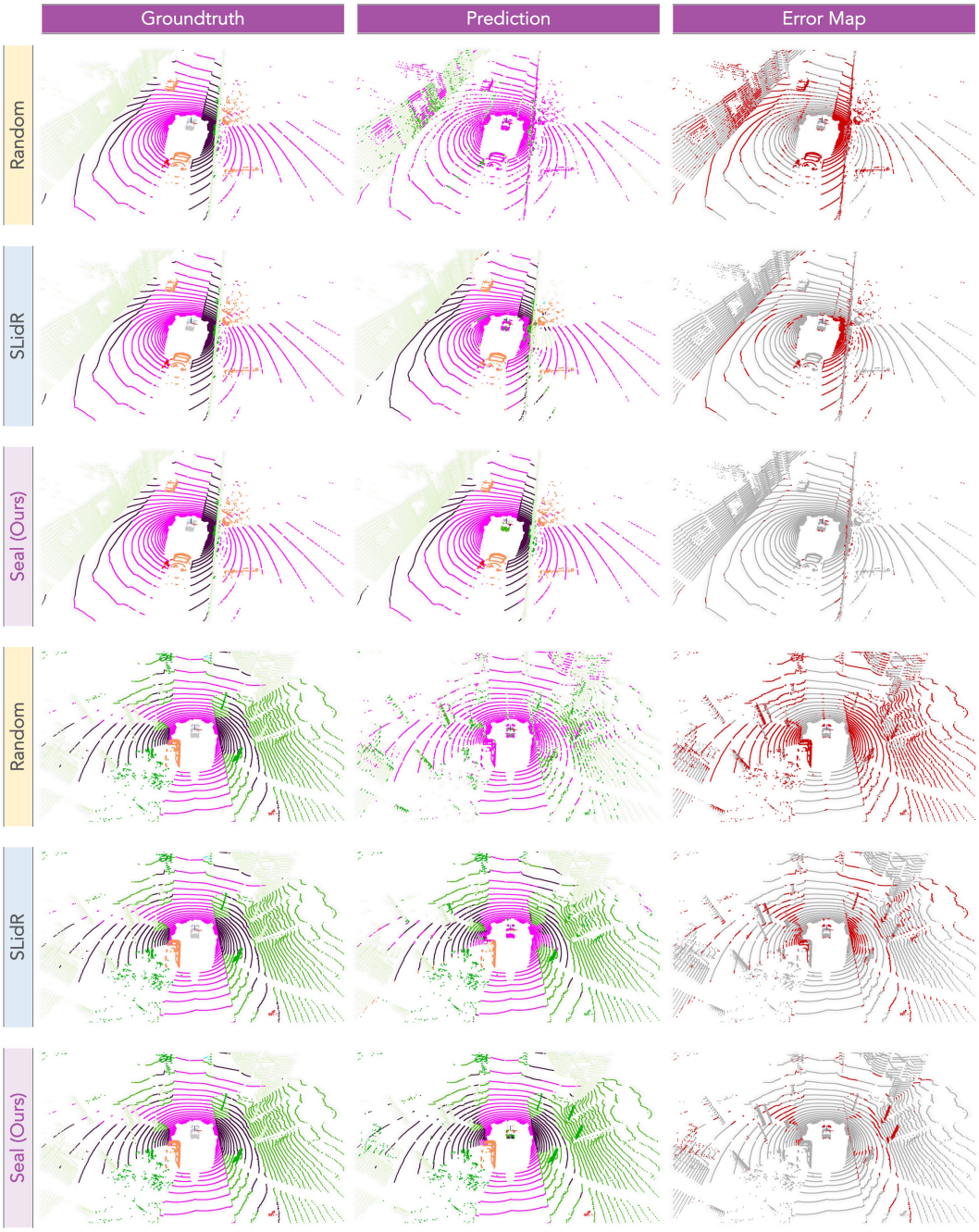


Figure J: The **qualitative results** of different point cloud pretraining approaches pretrained on the raw data of *nuScenes* [7] and fine-tuned with 1% labeled data. To highlight the differences, the **correct** / **incorrect** predictions are painted in **gray** / **red**, respectively. Best viewed in color.

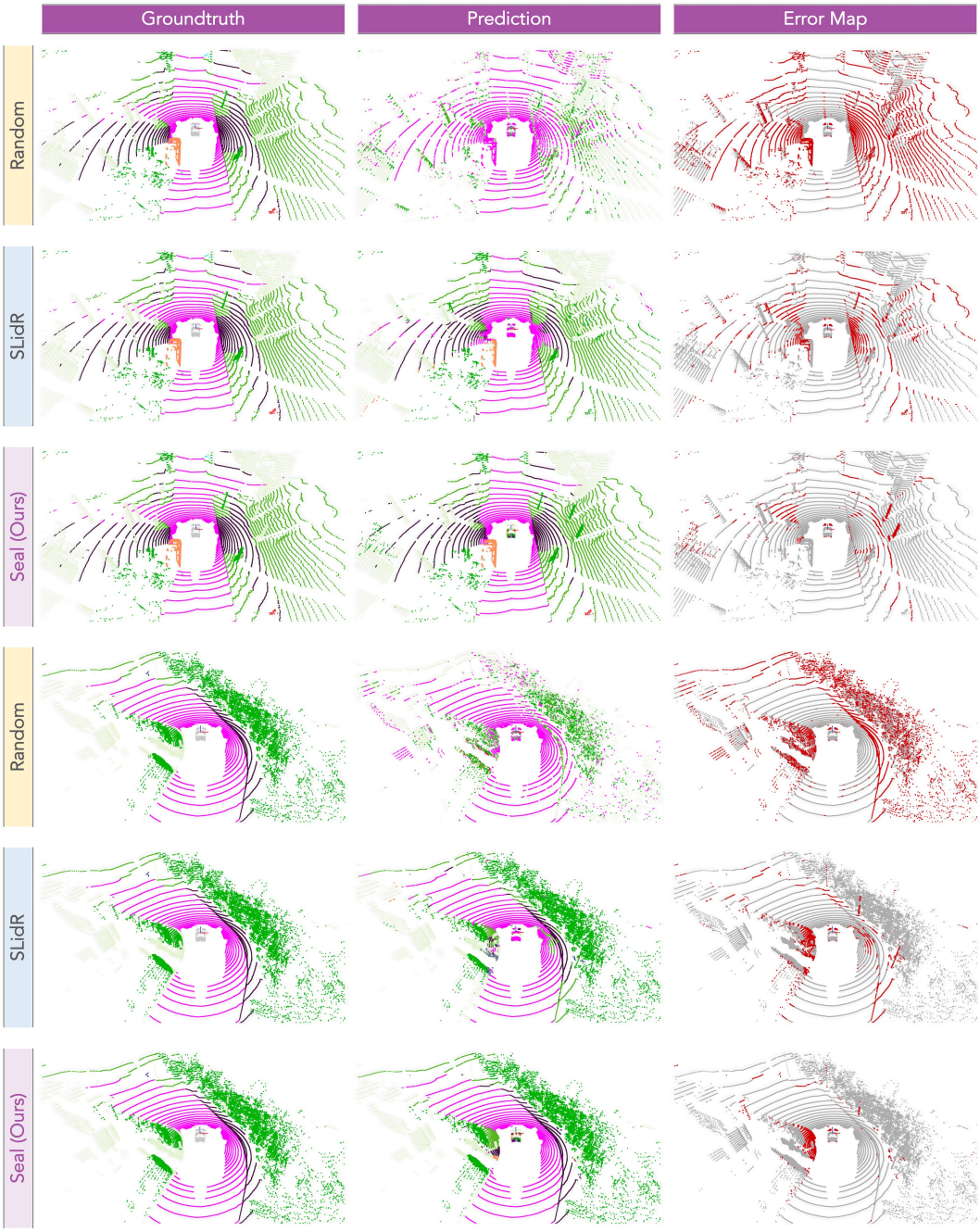


Figure K: The **qualitative results** of different point cloud pretraining approaches pretrained on the raw data of *nuScenes* [7] and fine-tuned with 1% labeled data. To highlight the differences, the **correct** / **incorrect** predictions are painted in **gray** / **red**, respectively. Best viewed in color.

D Public Resources Used

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

- nuScenes[2] ..... CC BY-NC-SA 4.0
- nuScenes-devkit[3] ..... Apache License 2.0
- SemanticKITTI[4] ..... CC BY-NC-SA 4.0
- SemanticKITTI-API[5] ..... MIT License
- Waymo Open Dataset[6] ..... Waymo Dataset License
- ScribbleKITTI[7] ..... Unknown
- RELLIS-3D[8] ..... CC BY-NC-SA 3.0
- SemanticPOSS[9] ..... Unknown
- SemanticSTF[10] ..... CC BY-NC-SA 4.0
- SynLiDAR[11] ..... MIT License
- Synth4D[12] ..... GNU General Public License 3.0
- DAPS-3D[13] ..... MIT License
- nuScenes-C[14] ..... CC BY-NC-SA 4.0
- MinkowskiEngine[15] ..... MIT License
- SLidR[16] ..... Apache License 2.0
- spvnas[17] ..... MIT License
- Cylinder3D[18] ..... Apache License 2.0
- LaserMix[19] ..... CC BY-NC-SA 4.0
- mean-teacher[20] ..... Attribution-NonCommercial 4.0 International
- PyTorch-Lightning[21] ..... Apache License 2.0
- mmdetection3d[22] ..... Apache License 2.0

[2] https://www.nuscenes.org/nuscenes.
[3] https://github.com/nutonomy/nuscenes-devkit.
[4] http://semantic-kitti.org.
[5] https://github.com/PRBonn/semantic-kitti-api.
[6] https://waymo.com/open.
[7] https://github.com/ouenal/scribblekitti.
[8] http://www.unmannedlab.org/research/RELLIS-3D.
[9] http://www.poss.pku.edu.cn/semanticposs.html.
[10] https://github.com/xiaoaoran/SemanticSTF.
[11] https://github.com/xiaoaoran/SynLiDAR.
[12] https://github.com/saltoricristiano/gipso-sfouda.
[13] https://github.com/subake/DAPS3D.
[14] https://github.com/ldkong1205/Robo3D.
[15] https://github.com/NVIDIA/MinkowskiEngine.
[16] https://github.com/valeoai/SLidR.
[17] https://github.com/mit-han-lab/spvnas.
[18] https://github.com/xinge008/Cylinder3D.
[19] https://github.com/ldkong1205/LaserMix.
[20] https://github.com/CuriousAI/mean-teacher.
[21] https://github.com/Lightning-AI/lightning.
[22] https://github.com/open-mmlab/mmdetection3d.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[2] 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 *IEEE/CVF International Conference on Computer Vision*, pages 9297–9307, 2019.

[3] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: a tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4413–4421, 2018.

[4] Mario Bijelic, Tobias Gruber, Fahim Mannan, Florian Kraus, Werner Ritter, Klaus Dietmayer, and Felix Heide. Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11682–11692, 2020.

[5] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

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

[7] Whye Kit Fong, Rohit Mohan, Juana Valeria Hurtado, Lubing Zhou, Holger Caesar, Oscar Beijbom, and Abhinav Valada. Panoptic nuscenes: A large-scale benchmark for lidar panoptic segmentation and tracking. *IEEE Robotics and Automation Letters*, pages 3795–3802, 2022.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[9] Peng Jiang, Philip Osteen, Maggie Wigness, and Srikanth Saripallig. Rellis-3d dataset: Data, benchmarks and analysis. In *IEEE International Conference on Robotics and Automation*, 2021.

[10] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.

[11] Alexey Klokov, Di Un Pak, Aleksandr Khorin, Dmitry Yudin, Leon Kochiev, Vladimir Luchinskiy, and Vitaly Bezuglyj. Daps3d: Domain adaptive projective segmentation of 3d lidar point clouds. *Preprint*, 2023.

[12] Lingdong Kong, Youquan Liu, Xin Li, Runnan Chen, Wenwei Zhang, Jiawei Ren, Liang Pan, Kai Chen, and Ziwei Liu. Robo3d: Towards robust and reliable 3d perception against corruptions. *arXiv preprint arXiv:2303.17597*, 2023.

[13] Lingdong Kong, Jiawei Ren, Liang Pan, and Ziwei Liu. Lasermix for semi-supervised lidar semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21705–21715, 2023.

[14] Yueh-Cheng Liu, Yu-Kai Huang, Hung-Yueh Chiang, Hung-Ting Su, Zhe-Yu Liu, Chin-Tang Chen, Ching-Yu Tseng, and Winston H. Hsu. Learning from 2d: Contrastive pixel-to-point knowledge transfer for 3d pretraining. *arXiv preprint arXiv:2104.0468*, 2021.

[15] Anas Mahmoud, Jordan SK Hu, Tianshu Kuai, Ali Harakeh, Liam Paull, and Steven L. Waslander. Self-supervised image-to-point distillation via semantically tolerant contrastive loss. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[16] 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 *IEEE Intelligent Vehicles Symposium*, pages 687–693, 2020.

[17] Cristiano Saltori, Evgeny Krivosheev, Stéphane Lathuiliére, Nicu Sebe, Fabio Galasso, Giuseppe Fiameni, Elisa Ricci, and Fabio Poiesi. Gipso: Geometrically informed propagation for online adaptation in 3d lidar segmentation. In *European Conference on Computer Vision*, pages 567–585, 2022.

[18] Corentin Sautier, Gilles Puy, Spyros Gidaris, Alexandre Boulch, Andrei Bursuc, and Renaud Marlet. Image-to-lidar self-supervised distillation for autonomous driving data. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9891–9901, 2022.

[19] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454, 2020.

[20] Ozan Unal, Dengxin Dai, and Luc Van Gool. Scribble-supervised lidar semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2697–2707, 2022.

[21] 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 Conference on Artificial Intelligence*, pages 2795–2803, 2022.

[22] Aoran Xiao, Jiaxing Huang, Weihao Xuan, Ruijie Ren, Kangcheng Liu, Dayan Guan, Abdulmotaleb El Saddik, Shijian Lu, and Eric Xing. 3d semantic segmentation in the wild: Learning generalized models for adverse-condition point clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[23] Hao Zhang, Feng Li, Xueyan Zou, Shilong Liu, Chunyuan Li, Jianfeng Gao, Jianwei Yang, and Lei Zhang. A simple framework for open-vocabulary segmentation and detection. *arXiv preprint arXiv:2303.08131*, 2023.

[24] Xueyan Zou, Zi-Yi Dou, Jianwei Yang, Zhe Gan, Linjie Li, Chunyuan Li, Xiyang Dai, Harkirat Behl, Jianfeng Wang, Lu Yuan, Nanyun Peng, Lijuan Wang, Yong Jae Lee, and Jianfeng Gao. Generalized decoding for pixel, image, and language. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[25] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once. *arXiv preprint arXiv:2304.06718*, 2023.