

Three Pillars improving Vision Foundation Model Distillation for Lidar

Gilles Puy¹ Spyros Gidaris¹ Alexandre Boulch¹ Oriane Siméoni¹
 Corentin Sautier^{1,3} Patrick Pérez^{2*} Andrei Bursuc¹ Renaud Marlet^{1,3}

¹valeo.ai, Paris, France ²Kyutai, Paris, France

³LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France

Abstract

Self-supervised image backbones can be used to address complex 2D tasks (e.g., semantic segmentation, object discovery) very efficiently and with little or no downstream supervision. Ideally, 3D backbones for lidar should be able to inherit these properties after distillation of these powerful 2D features. The most recent methods for image-to-lidar distillation on autonomous driving data show promising results, obtained thanks to distillation methods that keep improving. Yet, we still notice a large performance gap when measuring by linear probing the quality of distilled vs fully supervised features. In this work, instead of focusing only on the distillation method, we study the effect of three pillars for distillation: the 3D backbone, the pretrained 2D backbone, and the pretraining 2D+3D dataset. In particular, thanks to our scalable distillation method named ScaLR, we show that scaling the 2D and 3D backbones and pretraining on diverse datasets leads to a substantial improvement of the feature quality. This allows us to significantly reduce the gap between the quality of distilled and fully-supervised 3D features, and to improve the robustness of the pretrained backbones to domain gaps and perturbations. The code is available at <https://github.com/valeoai/ScaLR>.

1. Introduction

Lidars capture the 3D geometry of a scene with high accuracy while being little sensitive to adverse light conditions, which is useful for advanced driver assistance systems. However, annotating lidar point clouds to train deep neural networks is notoriously long and expensive [5].

More frugal learning can be achieved by pretraining backbone networks with self-supervision on a pretext task [8, 9, 22, 24, 25] and a non-annotated dataset. The pretrained network can then be finetuned on various down-

stream tasks and other datasets, with much less supervision for the same level of performance, or with a higher performance than when trained from scratch. Such pre-training has been particularly successful for image backbones [9, 12, 22]. In particular, the gap between supervised and self-supervised representations, as evaluated by linear probing, has been closed on ImageNet [8, 9, 12, 21, 22, 24].

On autonomous driving (AD) data, we distinguish mainly two categories of self-supervised methods for 3D backbones: (1) methods leveraging only lidar data and defining a pretext task at the level of a single [6, 44, 81] or multiple scans [45, 59, 67, 70], and (2) methods exploiting images acquired in synchronization with the point clouds and distilling image representations to a 3D backbone [38, 39, 41, 58]. Our focus is on the second category of methods, thanks to which 3D backbones can inherit the properties of self-supervised image backbones.

Image-to-lidar distillation has improved a lot recently, in particular by designing better distillation losses. Yet, we still observe a large gap between distilled and supervised 3D representations when directly measuring their quality by linear probing: they achieve respectively 45.0% and 74.7% mIoU in linear probing on the validation of nuScenes in [39] for a MinkUNet [14] with cylindrical voxels [84].

To improve the quality of distilled features, we explore here the effect of three other pillars, instead of focusing only on the distillation method. These pillars are: (i) the 3D backbone, (ii) the pretrained 2D backbones and (iii) the pretraining 2D+3D dataset. We show that scaling the 2D and 3D backbones, and pretraining on diverse datasets leads to a considerable improvements of the feature quality. The role of these pillars is actually more important than the distillation method itself, which we simplify for easier scaling.

After proposing and studying a scalable distillation method, which we call ScaLR for Scalable Lidar Representation, we make the following contributions.

First, we are able to significantly reduce the gap between distilled and supervised lidar representations. We reach an mIoU of 67.8% in linear probing on the validation set of

*Work done at valeo.ai.

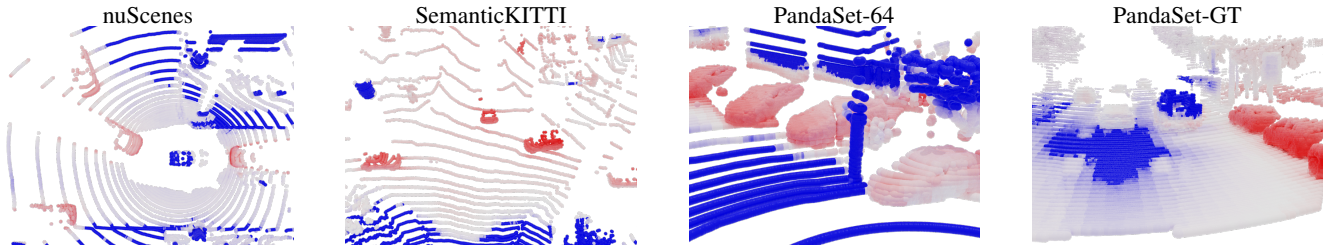


Figure 1. **Correlation properties of distilled 3D features.** Correlation maps with a point located on a car on four different scenes extracted from nuScenes [7], SemanticKITTI [5], PandaSet-64 and PandaSet-GT [69], respectively. The features used to compute these maps are extracted from a *single* pretrained backbone on all four datasets with ScaLR. Color goes from blue to red for low and high values.

nuScenes with a WaffleIron-48-768 backbone [51], i.e., an increase of 22.8 points compared to the score obtained with the best distillation method [39]. We recall that the highest¹ reported mIoU on the validation of nuScenes obtained under supervision is 78.4% [33], to the best of our knowledge. We are thus only about 10 points away from this upper bound.

Second, we show it is possible to pretrain a *single* backbone on a mixture of datasets, performing similarly or better than separate backbones specialized on each dataset individually. The capacity of this backbone in providing good features across multiple datasets is illustrated in Fig. 1. For each scene in this figure, we pick a point located on a car and present the feature correlation map with respect to this point. We notice that the most correlated points also belong to cars on all datasets, illustrating the capacity of our single pretrained backbone to correctly distinguish objects on multiple datasets.

Third, we thoroughly study the properties of our distilled features. We show that they are robust to both domain gaps and perturbations, actually leading to new state-of-the-art results on the benchmark of Robo3D [31]. We also show that pretraining on diverse datasets improves the robustness.

Finally, we show that a possible way to get even better features is to distill the knowledge from multiple vision foundation models at the same time, which can be easily done with our scalable distillation strategy.

2. Related Work

Pretraining 2D backbones. Supervised pretraining of 2D networks [19, 23, 32] with large-size data [16] produces powerful image representations that can transfer well to various downstream tasks in the image domain. Alternatively, self-supervised pretraining has been proven able to rival or surpass supervised pretraining while using only raw unlabeled image data. Two prominent approaches conduct self-supervised pretraining via either discriminative tasks that train a 2D network to extract features invariant to augmentations [9, 11, 22, 24], or via masked image modeling

tasks [4, 25] that hide part of an image and then train a network to reconstruct the missing fragment. An emerging pre-training paradigm is via language-image contrastive learning (CLIP) [53], in which the 2D network is trained with a text network to match image data with their paired captions.

Pretraining 3D backbones. Several self-supervised techniques for training 3D backbones have appeared recently. While the first techniques were limited to dense scans of single objects [13, 50, 57, 81], contrastive self-supervision has enabled significant improvements in self-supervision on large indoor and outdoor datasets [36, 44, 45, 70, 74, 81]. Instead of working on a single scan, some works also take advantage of the temporal dimension to construct their contrastive pretext tasks [28, 45, 59, 67]. Finally, other strategies reconstruct missing information hidden either artificially [26, 43, 47, 77, 78], or intrinsically because of the sparse point sampling in lidar acquisitions [6].

Using 2D pretrained features to train 3D backbones. Exploiting pretrained image backbones to train 3D networks has drawn a lot of attention recently. For example, in [17, 80], 3D autoencoders are pretrained on object-level point clouds where features extracted from a 2D backbone are used to build the supervision signal. Several methods [38, 39, 41, 58] distill the knowledge of 2D self-supervised backbones on large indoor or outdoor point clouds thanks to a contrastive loss that aligns 2D and 3D representations for pairs of corresponding point and pixel, while making the representation as dissimilar as possible for non matching point-pixel pairs. The same principle can also be applied for object-level point cloud [1]. Another type of strategy consists in pseudo-labeling 3D point clouds with class labels provided by 2D backbones trained under supervision [76]. Recently, several methods, e.g., [10, 49, 79], proposed to distill the knowledge of supervised vision-language models for open-vocabulary recognition or semantic segmentation. In addition to distillation, these methods also need to preserve alignment with the text representations. While our focus is not on open-vocabulary semantic segmentation, it is probable that some recipes discovered in our work can improve the distillation of vision-language models as well.

¹without test time augmentation.

Robustness. Making sure that trained models are robust to various type of perturbations is important for safety-critical applications, such as advanced driver assistance systems. Recently, several works [2, 18, 35, 72, 75] have proposed ways to evaluate or improve robustness of 3D perception models. In our work, we evaluate the robustness of our pretrained models on the Robo3D benchmark [31], which considers eight corruption types of different intensities meant to mimic different cases and degrees of distribution shifts from the original training distribution. As mentioned earlier, we show that distilling 2D features on data collected from multiple different lidars improves the robustness of our semantic segmentation models.

Domain generalization. 3D backbones trained on lidar data are sensitive to domain shifts such as the one induced by the different lidar sensors. Several techniques have thus been developed to improve the generalization capabilities of 3D models, e.g., [30, 55, 56, 68]. Alternatively, some techniques adapt these models to a new target domain using non-annotated target lidar data only, e.g., [42, 54, 73], or with the help of image data, e.g., [29, 37, 48]. While our method is not a domain generalization or domain adaptation technique per se, we consider the protocol used in this related literature to evaluate the capacity of our pretrained backbones to generalize to different lidar sensors.

3. Scalable Distillation Strategy

3.1. Motivation and Principle

Our goal is to study the benefit of using high-capacity 2D and 3D backbones, and of mixing data acquired by multiple lidars. To facilitate this study, it is important to use a distillation strategy that is scalable, with no or few hyperparameters to tune on each dataset, while remaining competitive.

While the state-of-the-art methods [38, 39, 41, 58] use a contrastive loss defined at the level of semantically coherent segments, in this work we relate pixel and point features directly using a simpler loss based on cosine similarity, whose usage actually appears naturally when distilling CLIP features [49]. This loss has the advantage to have no hyperparameters and, as we will see later, is very competitive without the cost of pre-computing segments.

Finally, AD datasets often contain images captured by multiple cameras synchronized with the lidar. The best practice for distillation is, for each lidar scan, to load all available synchronized images during batch loading. Nevertheless, the number of cameras varies from one dataset to the other, which creates some implementation difficulties when pretraining on a mix of different datasets, as in this work. Instead, we propose to have batches where each sample is created as follows: one camera is selected at random and we only load the corresponding image and points viewed in this camera. Therefore, data loading becomes

strictly identical whatever the number of cameras available in the pretraining dataset, which makes multi-datasets pre-training easier to implement. Furthermore, this strategy saves memory, making it easier to train large backbones.

3.2. Formal Description

Backbones. The 3D backbone ϕ_{3D} takes as input a point cloud $(p_1, \dots, p_N) \in \mathbb{R}^{N \times (3+1)}$, where each point p_i holds three Cartesian coordinates and the laser return intensity. It outputs deep features of dimension F_{3D} for all input points: $(f_1, \dots, f_N) \in \mathbb{R}^{N \times F_{3D}}$. Similarly, the 2D backbone ϕ_{2D} takes as input an RGB image of M pixels $(u_1, \dots, u_M) \in \mathbb{R}^{M \times 3}$ and outputs deep features of dimension F_{2D} for all pixels: $(g_1, \dots, g_M) \in \mathbb{R}^{M \times F_{2D}}$. In practice, we use bilinear interpolation to increase the resolution of the 2D feature map to the resolution of the input image.

3D Projection head. A linear projection head is added at the end of the 3D networks. We denote this projection head by ψ_{3D} . At distillation time, it projects 3D features into the output space of the 2D backbone. The projection head ψ_{3D} is removed after distillation and replaced by another one dedicated for the task of interest. We denote $(\tilde{f}_1, \dots, \tilde{f}_N) \in \mathbb{R}^{N \times F_{2D}}$ the output of $\psi_{3D} \circ \phi_{3D}$.

Point-pixel mapping. We assume the lidar and the camera (selected at random among all available cameras during batch creation) are calibrated so that we can put in correspondence 3D points and pixels. We can thus compute a point-to-pixel mapping $\rho : \{1, \dots, N\} \rightarrow \{1, \dots, M\}$. We are then able to retrieve the pixel feature $g_{\rho(i)}$ associated to each point p_i . We recall that we load only the points visible in the selected camera in our method, hence all points have a matching pixel.

Similarity loss. In our experiments, we use the following loss, which relies on pointwise cosine similarity

$$\mathcal{L}_{\text{sim}} = \frac{1}{N} \sum_{i \in \{1, \dots, N\}} \left\| \tilde{f}_i - g_{\rho(i)} \right\|_2, \quad (1)$$

where \tilde{f}_i and g_j are ℓ_2 -normalized.

3.3. Analysis of our Distillation Strategy

In this section, we conduct an analysis of different pretraining options to validate our choice of distillation strategy.

We conduct our experiments on two backbones: the MinkUNet [14] with cylindrical voxels [85] used in [39, 41, 58] and WaffleIron-48-256 [51] (48 layers with 256-dim. features). We perform our study on nuScenes [7]. We split the original 700 training scenes in a mini-train set and a mini-val set of 600 and 100 scenes, respectively (see [58] for details). All backbones are pretrained and linearly probed on the mini-train set and the results are reported on the mini-val set. For all settings, we distill features obtained from the DINO-pretrained ViT-S/8 [9].

	2D head	Contrast. loss	Cosine loss	No. cam.	Mem. / Time ↓	mIoU% ↑
MinkUNet	✓	✓	-	6	1.0 / 1.0	39.3
	✗	✓	-	6	1.6 / 1.0	43.4
	✗	-	✓	6	1.6 / 1.1	43.6
	✗	-	✓	1	0.8 / 0.3	42.1
WI-48-256	✓	✓	-	1	-	50.0 (±0.8)
	✗	✓	-	1	-	54.1
	✗	-	✓	1	-	56.7 (±0.9)

Table 1. **Analysis of distillation losses.** Effect of the distillation loss and number of images loaded per scan on the memory usage, training speed, and linear probing performance (mIoU). We show the relative gain in memory and training time with respect to the first row (lower value = less consumption). We report the standard deviation between parentheses evaluated over 3 different pretrainings for some selected experiments.

The results are presented in Tab. 1. We start from the baseline [38] which uses a contrastive loss and a 2D projection head on top of the 2D backbone. We then remove this 2D projection head and finally replace the loss by ours (Eq. 1). Note that the use of a 2D projection head is not compatible with (1) as an optimal but degenerated solution would be obtained by letting the parameters of the 2D and 3D projection heads go to zero. Finally, we measure the effect of loading only one image instead of six per batch. Based on the results of Tab. 1, we notice that our distillation strategy performs well thanks to the following advantages.

Preservation of the 2D feature space. The results in Tab. 1 show a clear detriment of the 2D projection for this unidirectional image-to-lidar distillation. The absence of a 2D projection head preserves the structure of the 2D feature space: there is no loss of information on the 2D side anymore. Leveraging as much as possible the original 2D features, obtained after pretraining on very large image datasets, is thus key to produce good 3D features.

Absence of false negatives. Even in absence of 2D projection head, the loss (1) performs better than the contrastive loss. We believe it is explained by the presence of false negatives in the contrastive loss. Indeed, with the cosine similarity loss, a feature f_i of a 3D point p_i , e.g., sampled on a car, is made as similar as possible to the 2D feature $g_{\rho(i)}$ of the corresponding pixel, falling at the same car location. However, with the contrastive loss, an additional mechanism comes into play: the feature f_i will also be made as dissimilar as possible to all other 2D features $g_{\rho(j)}, j \neq i$, even when these 2D features are originally similar, e.g., when they fall on the same object. This drawback of the contrastive loss seems to harm its performance significantly.

Scalability to large point-pixel pairs. Another drawback of the contrastive loss is that it cannot be computed using all

possible point-pixel pairs because of memory constraints. One needs to subsample these pairs, e.g., randomly, but then increasing the risk of ignoring small objects in the loss. The effects of this sub-optimality have been reduced by constructing the loss at the level of semantically coherent segments in [39, 41, 58], but at the cost of extra pre-processing time and additional parameters to tune. It is a cost we wish to avoid to pretrain more easily on multiple datasets. In fact, the cosine similarity loss does not require selecting point-pixel pairs or pre-extracting image/lidar segments.

Reduced memory usage. nuScenes provides images acquired from six different cameras, offering a 360° view of the scene. To reduce the memory requirement, we propose to use only one image (chosen at random among all available cameras) and only the points viewed in this camera when loading one element of a batch. Our result shows a slight drop of performance when batching with single cameras, but a significant gain in speed and memory. As explained in Sec. 3.1, this change has several benefits. First, the savings in memory and compute time make it easier to train large backbones, which, as we will see, is essential to reach good performance. Second, the data loading process becomes strictly identical whatever the number of cameras available in the pretraining dataset, which makes multi-datasets pretraining easier to implement.

In summary, our distillation strategy does not suffer from the presence of false negatives, does not require selection of point-pixel pairs, does not need extra pre-processing time (e.g., extracting segments), does not introduce extra hyper-parameters to tune (e.g., contrastive loss temperature, segmentation algorithm), and saves compute time and memory.

4. Experiments

In this section, we show that scaling the 2D and 3D backbones and pretraining on diverse datasets leads to a substantial improvement of the feature quality. First, we describe the backbones and datasets that we use. Second, we show that our scalable distillation strategy is competitive compared to other state-of-the-art methods. Third, we scale the 2D and 3D backbones and show that it results in distilled features of much better quality. Fourth, we pretrain a single backbone on multiple datasets captured by four different lidars and demonstrate that it performs as well as, or better than, backbones specialized to each lidar. Fifth, we evaluate the downstream finetuning performance, robustness and generalization properties of our pretrained backbones. Finally, we show that our method allows an easy combination of multiple 2D teachers to further boost the performance.

4.1. Three Pillars

3D backbones. We experiment with two different architectures designed for 3D point clouds: MinkUNet [14] with cylindrical voxels [85] and WaffleIron (WI) [51]. For WI,

we fix the depth to 48, as it led to the best results in [51], and vary the feature size: $F_{3D} \in \{96, 256, 384, 768\}$. We denote the corresponding backbone by WI- F_{3D} . We noticed numerical instabilities during training at $F_{3D} = 768$, which we solved by replacing all batchnorms by layernorms (except in the embedding and classification layers) at this size.

2D backbones. We concentrate on the distillation of self-supervised ViT features obtained with DINO [9], DINOv2 [46] or MAE [25]. Nevertheless, we also conduct one experiment with the MaskCLIP ViT-B/16 [82] to show the benefit of using multiple 2D teachers.

Pretraining datasets and lidars. We experiment on data collected from four different lidars: Velodyne-32 of nuScenes [7], Velodyne-64 of SemanticKITTI [5], Pandar 64 and Pandar GT of PandaSet [69]. On nuScenes and SemanticKITTI, we use the official train/val split and list of classes. We detail the train/val split and list of classes we used on PandaSet in the supplementary material.

4.2. Implementation Details

For all experiments conducted with MinkUNet, we use the implementation provided by [58] and the same hyperparameters, unless otherwise stated. We describe below the main implementation details used to pretrain, linear probe and finetune WI backbones. More information, such as learning rate schedules, weight decay and number of epochs are provided in the supplementary material.

Data augmentation. We do not apply any image augmentation, beyond systematic resizing to 224×448 . During pretraining, finetuning and linear probing, we apply the following point cloud augmentations: random rotation around the z -axis, random flip of the x and y axes. During finetuning and linear probing, we also globally scale the coordinates by a random factor chosen uniformly in $[0.9, 1.1]$.

Linear probing. We remove ψ_{3D} and replace it by a batch normalization layer followed by a linear classification layer. We denote the combination of these last layers by κ . Note that the combination of batch normalization followed by linear classification acts as a linear layer at inference, but the normalization layer makes the results less sensitive to the downstream learning rate [34]. While maintaining ϕ_{3D} fixed, we train $\kappa \circ \phi_{3D}$ using ground-truth labels.

Finetuning. For finetuning, we remove the distillation layer ψ_{3D} and replace it by a batch normalization layer directly followed by a linear classification layer. As before, we denote the combination of these last layers by κ and we train $\kappa \circ \phi_{3D}$ using ground-truth labels. The backbone is finetuned using a layer-wise learning rate decay [3, 15], a technique commonly used when finetuning ViTs [4, 25, 83].

4.3. Comparison of Distillation Methods

In this section, we verify that our distillation strategy remains competitive with respect to the state-of-the-art in

3D Back.	Method	2D Back.	Lin. Prob.	1%
MinkUNet	PPKT [38]	ResNet-50	<i>36.4</i>	<i>37.5</i>
	SLidR [58]	ResNet-50	<i>38.8</i>	<i>39.0</i>
	ST-SLidR [41]	ResNet-50	<i>40.5</i>	<i>40.8</i>
	Seal [39]*	ResNet-50	<i>45.0</i>	<i>45.8</i>
MinkUNet	PPKT [38]	ViT-S/8	38.6	40.6
	SLidR [58]	ViT-S/8	39.3	39.0
	ScaLR (ours)	ViT-S/8	42.4	40.5
WI-96	ScaLR (ours)	ViT-S/8	46.8	38.8
WI-256	ScaLR (ours)	ViT-S/8	54.2	41.4

Table 2. **Pretraining method comparison and effect of 2D/3D backbones.** The mIoU% reported in *italic* are found in the literature while the others are our production. All methods are pretrained on the same 600 scenes of nuScenes. Linear probing uses the 700 training scenes of nuScenes. Finetuning uses only 1% of nuScenes training set. All scores are reported on the official validation split. *leverages a supervised backbone [86].

image-to-lidar distillation. All experiments are conducted using the experimental protocol used in [39, 41, 58], i.e., pretraining on the reduced training set of nuScenes (600 scenes) defined in [58] and using the official train/val split for downstream linear probing and finetuning.

We start by distilling the representations of DINO ViT-S/8 in a MinkUNet. These representations are distilled using three different methods: PPKT [38], SLidR [58] and using our cosine-based similarity loss (but still loading all six images as in PPKT and SLidR). The MinkUNet backbones are pretrained with a batch size of 12 (we scale the learning rate proposed in [58] accordingly) during 25 epochs. The quality of the distilled features is assessed by linear probing. We also finetune the backbones using 1% the complete training set of nuScenes. We report the corresponding scores in Tab. 2, where we also present the results obtained in the literature by distilling ResNet-50 representations.

First, we notice that changing the 2D backbones from ResNet-50 to ViT-S/8 helps both PPKT and SLidR, which both reach a higher mIoU in linear probing and finetuning after this change. The improvement seems however more significant for PPKT than for SLidR.

Second, we remark that **ScaLR leads to better 3D features than those obtained with PPKT and SLidR**: we reach a higher score in linear probing; we also obtain a better result than SLidR in finetuning with 1% of the training data, and a level of performance similar to PPKT. This confirms that using the simple cosine-based similarity loss of Eq. (1) is a competitive choice.

Third, we also pretrained WI-96 and WI-256 backbones on this dataset using our full scalable distillation protocol. We notice that **changing MinkUNet to WI further boosts linear probing performance**. We remark that WI-96 has

3D Back.	2D Back.	nuScenes		SemKITTI	
		LP	$\Delta \downarrow$	LP	$\Delta \downarrow$
No pretraining. Best fully supervised WI.					
WI- F_{3D}	–	78.7	($F_{3D} = 768$)	65.1	($F_{3D} = 256$)
MAE-pretrained					
WI-256	ViT-B/16	47.8	30.9	–	–
DINO-pretrained					
WI-256	ViT-S/8	54.4	24.3	–	–
WI-384	ViT-S/8	57.8	20.9	–	–
WI-768	ViT-S/8	61.1	17.6	–	–
DINOv2-pretrained					
WI-256	ViT-S/14	57.7	21.0	–	–
WI-256	ViT-B/14	60.2	18.5	–	–
WI-256	ViT-L/14	61.6	17.1	48.3	16.8
WI-384	ViT-L/14	63.9	14.8	50.6	14.5
WI-768	ViT-L/14	66.8	11.9	51.1	14.0

Table 3. **Effect of the 3D network width and of the quality of the 2D features.** We report the mIoU% obtained by linearly probing distilled features. The networks are trained on nuScenes or SemanticKITTI using ScaLR. The scores are reported on the corresponding official validation split. The column Δ shows the gap with respect to the best fully supervised mIoU% reached with WI on the corresponding dataset.

the same output feature dimension than the MinkUNet backbone but fewer trainable parameters (1M vs 35M). Hence, the improvement of performance in linear probing cannot be solely explained by the 3D feature dimension or the capacity of the backbone; other elements in the architecture must come into play but identifying the exact origin of this improvement is beyond the scope of this work.

Fourth, when finetuning on 1% of nuScenes training set, **our ScaLR WI-256 model surpasses prior methods.** The only exception is Seal [39] that leverages a supervised backbone (segment-everything-everywhere model [86]) to create segments. It is probable that we could reach higher performance by using such extra supervision. Yet, we leave this research direction for the future as our distillation strategy is competitive enough for the purpose of our study.

Finally, we already notice that distilling features in WI backbones of increasing width helps both the linear probing and finetuning performance. We continue to study this property in the next section with WI, as this is the backbone which led to the best performance with our technique.

4.4. 2D Backbone Choice & 3D Backbone Scale

In this section, we show that increasing the width of WI leads to substantial improvements of the feature quality. This effect combined with a good choice of 2D backbone significantly reduces the gap between the quality of distilled 3D features and fully-supervised 3D features.

We conduct experiments with three different WaffleIron

Pretrain. Dataset	Downstream & Test Dataset			
	nuScenes	SemKITTI	Pand. 64	Pand. GT
<i>No Pretraining - Best fully supervised WI</i>				
WI- F_{3D}	78.7	65.1	47.8	40.6
–	$F_{3D} = 768$	$F_{3D} = 256$	$F_{3D} = 768$	$F_{3D} = 256$
<i>Pretraining with DINO-ViT-S/8 and linear probing</i>				
WI- F_{3D}	54.4	28.8	26.9	25.2
WI-256	39.5	<u>46.6</u>	25.3	25.7
Pandar 64	39.6	25.6	30.0	24.7
Pandar GT	29.9	26.9	23.5	<u>28.5</u>
nuSc. & KITTI	54.4	50.1	29.3	28.9
All datasets	54.6	50.6	33.1	32.3
<i>Pretraining with DINOv2-ViT-L/14 and linear probing</i>				
WI-768	67.8	43.1	33.9	29.9
All datasets	67.8	55.8	37.9	34.5

Table 4. **Benefit of pretraining on diverse datasets – Linear probing.** Performance after linearly probing distilled features. The backbones are pretrained on nuScenes, SemanticKITTI, PandaSet-64, PandaSet-GT, nuScenes & SemanticKITTI, or all these datasets together, and then linearly probed on each individual dataset. The reported mIoU% is computed on the val split of each dataset. The underlined mIoU% highlights the score obtained by pretraining and linear probing on the same dataset.

backbones: WI-256, WI-384, WI-768. We pretrain these backbones on the complete training set of nuScenes or SemanticKITTI, and evaluate the quality of the distilled features on the validation set of the respective datasets. The evaluation consists of linear probings using 100% of the annotated training scans. The results obtained by distilling MAE, DINO, and DINOv2 features are presented in Tab. 3.

First, we observe on nuScenes that pretraining WI-256 backbones with ViTs pretrained with DINOv2 instead of DINO consistently improves the scores. We also notice that the linear probing performance improves when distilling DINOv2 ViTs of increasing capacities. This indicates that 3D backbones pretrained with **our method can directly benefit from an improved quality of the image features.** Interestingly, the MAE-pretrained ViT leads to the worst performance. We attribute this to the fact that MAE features are less linearly separable, as reported in [25].

Second, we notice that **the quality of our distilled features increases significantly with the WI width** for ViT-S/8 and ViT-L/14 on nuScenes and SemanticKITTI. We reach a mIoU of 66.8% by linear probing on nuScenes. This is only about 11.9 points away from the mIoU reached by WI-768 when trained under full supervision. Note that WI-768 is actually well-performing on this dataset as it reaches a mIoU of 78.7% while the best published score we know of on this dataset, without test time augmentations, is 78.4% (reported in [33]). It shows that **distilling 2D features**

Pretrain. Dataset	nuScenes			Sem.KITTI		Pan. 64	Pan. GT
	1%	10%	100%	1%	100%	100%	100%
<i>No Pretraining - Fully supervised WI-768</i>							
-	35.2	62.2	78.7	49.6	63.4	47.8	40.0
<i>Pretraining WI-768 with DINOv2-ViT-L/14 and finetuning</i>							
nuScenes	51.0	70.5	77.9	49.7	63.9	49.4	42.1
All	50.7	69.2	78.4	56.8	65.8	48.3	41.1

Table 5. **Finetuning.** The backbones are pretrained on nuScenes alone or all considered datasets (‘All’), and finetuned on the train split of each dataset for different amounts of data. The reported mIoU% is computed on the val split of each dataset.

without manual annotations can bridge the gap with the quality of fully supervised 3D features.

4.5. Pretraining on Multiple Datasets

In this section, we show that we can pretrain a single backbone on multiple datasets so that it performs as well or better than individual backbones specialized to each dataset.

We conduct our main experiment with WI-256 and the DINO-pretrained ViT-S/8. The datasets that we consider (nuScenes, SemanticKITTI, PandaSet with Pandar64 or PandarGT) have different sizes. To avoid drawing conclusions which could be explained by different amount of pretraining, we adjust the number of pretraining epochs on each dataset so that the backbones are pretrained for the same number of iterations (the batch size is fixed to 16).

We start by pretraining WI-256 on each dataset individually and evaluate the quality of the distilled features by linear probing on each of the datasets. The results are presented in Tab. 4. We notice that the pretrained backbones are quite sensitive to the domain gap induced by different lidars. For example, the backbone pretrained on nuScenes performs significantly worse on SemanticKITTI than the backbone pretrained on SemanticKITTI: 28.8% vs 46.6% mIoU. The observation is valid whatever pairs of pretraining/downstream datasets we take. However, by pretraining a backbone on both nuScenes and SemanticKITTI, we do not notice any drop in linear probing performance on nuScenes and achieve even better results on SemanticKITTI. The best overall performance is obtained when pretraining on all datasets together. This experiment shows that **a single backbone easily benefits from diverse lidar data without performance loss in linear probing.**

We conduct one more experiment using the best combination of 3D and 2D backbone discovered: WI-768 and DINOv2 ViT-L/14. We pretrain this backbone either on nuScenes alone or on the combination of all datasets. Due to memory constraints at this scale, we reduce the batch size to 8 for both pretrainings. We thus also used a longer training schedule. The results are presented in the last

Method	3D Back.	Pretrain dataset	mCE% ↓	mRR% ↑	Avg. mIoU%
-	Cylind3D [84]	-	<i>105.6</i>	<i>78.1</i>	<i>57.4</i>
-	2DPASS [71]	-	<i>98.6</i>	<i>75.2</i>	<i>58.6</i>
-	SPVCNN [64]	-	<i>97.5</i>	<i>75.1</i>	<i>57.5</i>
-	GFNet [52]	-	<i>92.6</i>	<i>83.3</i>	<i>64.0</i>
-	WI-768	-	90.9	80.6	63.5
PPKT	MinkUNet	nuScenes	<i>105.6</i>	<i>76.1</i>	<i>56.6</i>
SLiDR	MinkUNet	nuScenes	<i>106.1</i>	<i>76.0</i>	<i>56.8</i>
Seal	MinkUNet	nuScenes	<i>92.6</i>	<i>83.1</i>	<i>62.8</i>
ScaLR	WI-768	nuScenes	89.1	83.7	65.2
ScaLR	WI-768	Multiple	87.4	83.8	65.7

Table 6. **Robustness to corruptions.** The evaluation is done on nuScenes-C from the Robo3D benchmark [31]. We report the mCE%, mRR%, and the mIoU% averaged over all eight corruptions. The scores in italic are obtained from [31, 39]. PPKT, SLiDR and Seal use a MoCov2 ResNet-50. We use DINOv2 ViT-L/14.

section of Tab. 4. We notice again that **pretraining on multiple datasets leads to better linear probing performance across all downstream datasets.** We also remark that reducing the batch size, and training for longer, gains 1 mIoU point when training only on nuScenes (see WI-768 in Tabs. 3 and 4). The remaining gap with respect to the best fully-supervised WI baseline is less than 10.9 mIoU points on all datasets. Notably, on nuScenes, we reach 67.8 in mIoU while the best score so far in linear probing was 45.0 (Tab. 8), reached by [39] with the help of fully supervised 2D backbones to create semantically coherent segments.

4.6. Properties of Distilled Features

In this section, we use the WI-768 backbones pretrained with ScaLR using DINOv2 ViT-L/14. We consider two pretraining datasets: (1) nuScenes alone, and (2) the combination of nuScenes, SemanticKITTI, PandaSet-64/GT (denoted ‘Multiple’ in pretraining dataset column in tables).

Performance in finetuning. We present in Tab. 5 the results obtained by finetuning our pretrained WI-768 on different datasets for different amounts of data. First, we notice that **pretraining always improves the score compared to the non-pretrained fully supervised baseline,** except on nuScenes when using all the available annotations. We also notice that WI-768 pretrained on multiple datasets: (1) significantly outperforms the nuScenes-pretrained WI-768 when finetuning on SemanticKITTI; (2) performs nearly as well on nuScenes as the WI-768 model pretrained on nuScenes, i.e., the model specialized to this dataset. Interestingly, WI-768 pretrained on nuScenes seems to have a small advantage for finetuning on PandaSet. We recall nevertheless that the model pretrained on multiple datasets still improves the performance compared to no pretraining. More importantly, as will be seen in the next

			3D Back	2D Back		LP
				MaskCLIP	DINOv2	
			WI-256	✓	✗	59.4
Pretrain Dataset	S.KITTI	S.POSS	WI-256	✗	✓	61.6
			WI-256	✓	✓	62.5
			WI-384	✓	✗	60.9
–	28.6	54.0	WI-384	✗	✓	63.9
nuScenes	29.6	57.1	WI-384	✓	✓	65.1
Multiple	36.6	59.1	WI-384	✓	✓	65.1

Table 7. **Domain generalization properties (left) and benefit of pretraining with multiple teachers (right).** *Left:* mIoU% on SemanticKITTI (abbreviated S.KITTI) and SemanticPOSS (abbreviated S.POSS) of WI-768 pretrained either on nuScenes alone or on all considered pretraining datasets, and then finetuned on nuScenes. The performance of WI-768 trained under full supervision on nuScenes without pretraining is also reported. *Right:* Linear probing mIoU% obtained on nuScenes after pretraining WI with ScaLR on nuScenes. The considered 2D backbones are DINOv2 ViT-L/14 and MaskCLIP ViT-B/16.

experiment, **multi-dataset pretraining produces more robust backbones against perturbations and domain gaps.**

Robustness to corruptions. We evaluate the robustness of the pretrained WI-768 to corruptions on Robo3D benchmark [31]. The results are in Tab. 6. We notice that WI-768 trained under full supervision on nuScenes, without any pretraining, already performs well compared to the other 3D backbones trained in the same condition. WI-768 achieves the best results in terms of mean corruption error (mCE) and is second in terms of mean resilience rate (mRR). We remark that pretraining always improves the robustness of WI-768. The best performance is achieved thanks to our multiple datasets pretraining with a mCE of 87.4% and a mRR of 83.8%, i.e., yielding new state-of-the-art results.

Domain generalization properties. We evaluate the capacity of our pretrained WI-768 to generalize to different lidars. The backbones are finetuned on nuScenes after pretraining. We measure the performance of the resulting models directly on the val set of SemanticKITTI (train set seen during pretraining) or SemanticPOSS (not seen during pretraining) using respectively 10 and 6 aggregated classes, as done, e.g., in [42, 73]. We notice in Tab. 7 that pretraining always helps generalization and that the best performance are obtained when pretraining on multiple datasets.

4.7. Pretraining with Multiple Teachers

We show in Tab. 7 that using multiple 2D teachers is a way to further improve the performance. Distillation was done using the two modifications in ScaLR: (1) On the 2D side, we concatenate the pixel features obtained from both backbones; (2) On the 3D side, we use a 2-layer MLP for ψ_{3D} with an output size matching the dimension of the concate-

Method	3D	Pretrain	nuScenes				S.KITTI	
			LP	1%	10%	100%	1%	
–	MinkUNet	–	<i>8.1</i>	<i>30.3</i>	<i>56.2</i>	<i>74.7</i>	<i>39.5</i>	
–	WI-768	–	–	35.2	62.2	78.7	49.6	
PPKT	MinkUNet	nuScenes	<i>35.9</i>	<i>37.8</i>	<i>60.3</i>	<i>74.5</i>	<i>44.0</i>	
SLidR	MinkUNet	nuScenes	<i>38.8</i>	<i>38.3</i>	<i>59.8</i>	<i>74.8</i>	<i>44.6</i>	
ST-SLidR	MinkUNet	nuScenes	<i>40.5</i>	<i>40.8</i>	<i>60.8</i>	<i>75.1</i>	<i>44.7</i>	
Seal	MinkUNet	nuScenes	<i>45.0</i>	<i>45.8</i>	<i>63.0</i>	<i>75.6</i>	<i>46.6</i>	
ScaLR	WI-768	Multiple	67.8	50.7	69.2	78.4	56.8	

Table 8. **Progress made with ScaLR.** We report the mIoU% obtained by linear probing (LP) or finetuning on different amounts of data. The scores in italic are reported from [39]. WI-768 is pretrained using DINOv2 ViT-L/14 on nuScenes, SemanticKITTI (abbreviated S.KITTI), PandaSet-64/GT. The MinkUNet backbones are pretrained using MoCov2 ResNet-50 on nuScenes.

nated pixel features (see details in supp. mat.).

5. Conclusion

We have seen that scaling 2D and 3D backbones and pretraining on multiple datasets lead to better and more robust distilled features. We summarize in Tab. 8 the scores obtained by [38, 39, 41, 58] and with ScaLR to highlight the significant progress we have made. The most notable improvement appears when linear probing and finetuning on small amounts of data. Note that we *do not* claim that the concurrent distillation methods cannot surpass our proposed scalable distillation method; the take-away message is that scaling the 2D and 3D backbones and pretraining on diverse datasets actually lead to significantly better pretrained backbones even with a simple distillation method. These are three pillars that should not be overlooked to get good pretrained backbones for lidars. We hope that these findings, combined with more powerful distillation methods, will lead to even better 3D backbones in the future.

Acknowledgement This project benefited from an access to the CINES HPC resources under allocation GDA2213 and CIN4182, and to Karolina at IT4Innovations (EuroHPC Joint Undertaking). We thank Mr Cloirec and Mr Penalva at CINES. This research received the support of the French Agence Nationale de la Recherche (ANR), under grant ANR-21-CE23-0032 (project MultiTrans), and of EXA4MIND project, funded by a European Union’s Horizon Europe Research and Innovation Programme, under Grant Agreement N° 101092944. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

References

- [1] Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. CrossPoint: Self-supervised cross-modal contrastive learning for 3D point cloud understanding. In *CVPR*, 2022. [2](#)
- [2] Fatima Albreiki, Sultan Abu Ghazal, Jean Lahoud, Rao Anwer, Hisham Cholakkal, and Fahad Khan. On the robustness of 3d object detectors. In *ICM Asia*, 2022. [3](#)
- [3] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, et al. A cookbook of self-supervised learning. *arXiv:2304.12210*, 2023. [5](#)
- [4] Hangbo Bao, Li Dong, and Furu Wei. BEiT: BERT pre-training of image transformers. In *ICLR*, 2022. [2](#), [5](#)
- [5] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jürgen Gall. SemanticKITTI: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, 2019. [1](#), [2](#), [5](#)
- [6] Alexandre Boulch, Corentin Sautier, Björn Michele, Gilles Puy, and Renaud Marlet. ALSO: Automotive lidar self-supervision by occupancy estimation. In *CVPR*, 2023. [1](#), [2](#), [14](#)
- [7] 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 multi-modal dataset for autonomous driving. In *CVPR*, 2020. [2](#), [3](#), [5](#)
- [8] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020. [1](#)
- [9] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021. [1](#), [2](#), [3](#), [5](#)
- [10] Runnan Chen, Youquan Liu, Lingdong Kong, Xinge Zhu, Yuexin Ma, Yikang Li, Yuenan Hou, Yu Qiao, and Wenping Wang. CLIP2Scene: Towards label-efficient 3D scene understanding by CLIP. *CVPR*, 2023. [2](#)
- [11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. [2](#)
- [12] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021. [1](#)
- [13] Ye Chen, Jinxian Liu, Bingbing Ni, Hang Wang, Jiancheng Yang, Ning Liu, Teng Li, and Qi Tian. Shape self-correction for unsupervised point cloud understanding. In *ICCV*, 2021. [2](#)
- [14] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D spatio-temporal ConvNets: Minkowski convolutional neural networks. In *CVPR*, 2019. [1](#), [3](#), [4](#)
- [15] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *ICLR*, 2020. [5](#)
- [16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. [2](#)
- [17] Runpei Dong, Zekun Qi, Linfeng Zhang, Junbo Zhang, Jianjian Sun, Zheng Ge, Li Yi, and Kaisheng Ma. Autoencoders as cross-modal teachers: Can pretrained 2D image transformers help 3D representation learning? *ICLR*, 2022. [2](#)
- [18] Yinpeng Dong, Caixin Kang, Jinlai Zhang, Zijian Zhu, Yikai Wang, Xiao Yang, Hang Su, Xingxing Wei, and Jun Zhu. Benchmarking robustness of 3d object detection to common corruptions. In *CVPR*, 2023. [3](#)
- [19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Jakob Uszkoreit, Mostafa Dehghani, Neil Houlsby, Matthias Minderer, Georg Heigold, and Sylvain Gelly and. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. [2](#)
- [20] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *CVPR*, pages 3354–3361, 2012. [14](#)
- [21] Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Pérez. Obow: Online bag-of-visual-words generation for self-supervised learning. In *CVPR*, 2021. [1](#)
- [22] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap Your Own Latent: A new approach to self-supervised learning. In *NeurIPS*, 2020. [1](#), [2](#)
- [23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. [2](#)
- [24] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. [1](#), [2](#)
- [25] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. [1](#), [2](#), [5](#), [6](#)
- [26] Georg Hess, Johan Jaxing, Elias Svensson, David Hagerman, Christoffer Petersson, and Lennart Svensson. Masked autoencoder for self-supervised pre-training on lidar point clouds. In *WACV*, 2023. [2](#)
- [27] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016. [13](#)
- [28] Siyuan Huang, Yichen Xie, Song-Chun Zhu, and Yixin Zhu. Spatio-temporal self-supervised representation learning for 3D point clouds. In *ICCV*, 2021. [2](#)
- [29] Maximilian Jaritz, Tuan-Hung Vu, Raoul de Charette, Emilie Wirbel, and Patrick Pérez. xmuda: Cross-modal unsupervised domain adaptation for 3d semantic segmentation. In *CVPR*, 2020. [3](#)
- [30] Hyeonseong Kim, Yoonsu Kang, Changgyoon Oh, and Kuk-Jin Yoon. Single domain generalization for lidar semantic segmentation. In *CVPR*, 2023. [3](#)

- [31] 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. In *ICCV*, 2023. [2](#), [3](#), [7](#), [8](#), [12](#), [13](#)
- [32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *NeurIPS*, 2012. [2](#)
- [33] Xin Lai, Yukang Chen, Fanbin Lu, Jianhui Liu, and Jiaya Jia. Spherical transformer for lidar-based 3D recognition. In *CVPR*, 2023. [2](#), [6](#)
- [34] Jae-Hun Lee, Doyoung Yoon, ByeongMoon Ji, Kyungyul Kim, and Sangheum Hwang. Rethinking evaluation protocols of visual representations learned via self-supervised learning. *arXiv:2304.03456*, 2023. [5](#)
- [35] Shuangzhi Li, Zhijie Wang, Felix Juefei-Xu, Qing Guo, Xingyu Li, and Lei Ma. Common corruption robustness of point cloud detectors: Benchmark and enhancement. *T-Multimedia*, 2023. [3](#)
- [36] Kangcheng Liu, Aoran Xiao, Xiaoqin Zhang, Shijian Lu, and Ling Shao. FAC: 3D representation learning via foreground aware feature contrast. In *CVPR*, 2023. [2](#)
- [37] Wei Liu, Zhiming Luo, Yuanzheng Cai, Ying Yu, Yang Ke, José Marcato Junior, Wesley Nunes Gonçalves, and Jonathan Li. Adversarial unsupervised domain adaptation for 3d semantic segmentation with multi-modal learning. *ISPRS*, 2021. [3](#)
- [38] Yueh-Cheng Liu, Yu-Kai Huang, HungYueh Chiang, Hung-Ting Su, Zhe Yu Liu, Chin-Tang Chen, Ching-Yu Tseng, and Winston H. Hsu. Learning from 2D: Pixel-to-point knowledge transfer for 3D pretraining. *arxiv:2104.04687*, 2021. [1](#), [2](#), [3](#), [4](#), [5](#), [8](#), [13](#)
- [39] Youquan Liu, Lingdong Kong, Jun Cen, Runnan Chen, Wenwei Zhang, Liang Pan, Kai Chen, and Ziwei Liu. Segment Any Point Cloud Sequences by Distilling Vision Foundation Models. In *NeurIPS*, 2023. [1](#), [2](#), [3](#), [4](#), [5](#), [6](#), [7](#), [8](#), [13](#)
- [40] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. [12](#)
- [41] 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. *CVPR*, 2023. [1](#), [2](#), [3](#), [4](#), [5](#), [8](#)
- [42] Bjoern Michele, Alexandre Boulch, Gilles Puy, Tuan-Hung Vu, Renaud Marlet, and Nicolas Courty. SALUDA: Surface-based automotive lidar unsupervised domain adaptation. In *3DV*, 2024. [3](#), [8](#)
- [43] Chen Min, Dawei Zhao, Liang Xiao, Yiming Nie, and Bin Dai. Voxel-MAE: Masked autoencoders for pre-training large-scale point clouds. *arXiv:2206.09900*, 2022. [2](#)
- [44] Lucas Nunes, Rodrigo Marcuzzi, Xieyuanli Chen, Jens Behley, and Cyrill Stachniss. SegContrast: 3D point cloud feature representation learning through self-supervised segment discrimination. *RA-L*, 2022. [1](#), [2](#)
- [45] Lucas Nunes, Louis Wiesmann, Rodrigo Marcuzzi, Xieyuanli Chen, Jens Behley, and Cyrill Stachniss. Temporal consistent 3D lidar representation learning for semantic perception in autonomous driving. In *CVPR*, 2023. [1](#), [2](#)
- [46] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *TMLR*, 2024. [5](#)
- [47] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *ECCV*, 2022. [2](#)
- [48] Duo Peng, Yinjie Lei, Wen Li, Pingping Zhang, and Yulan Guo. Sparse-to-dense feature matching: Intra and inter domain cross-modal learning in domain adaptation for 3d semantic segmentation. In *ICCV*, 2021. [3](#)
- [49] Songyou Peng, Kyle Genova, Chiyu "Max" Jiang, Andrea Tagliasacchi, Marc Pollefeys, and Thomas Funkhouser. OpenScene: 3D scene understanding with open vocabularies. In *CVPR*, 2023. [2](#), [3](#)
- [50] Omid Poursaeed, Tianxing Jiang, Han Qiao, Nayun Xu, and Vladimir G Kim. Self-supervised learning of point clouds via orientation estimation. In *3DV*, 2020. [2](#)
- [51] Gilles Puy, Alexandre Boulch, and Renaud Marlet. Using a waffle iron for automotive point cloud semantic segmentation. In *ICCV*, 2023. [2](#), [3](#), [4](#), [5](#), [12](#), [14](#)
- [52] Haibo Qiu, Baosheng Yu, and Dacheng Tao. GFNet: Geometric Flow Network for 3D Point Cloud Semantic Segmentation. *TMLR*, 2022. [7](#), [13](#)
- [53] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. [2](#)
- [54] Cristiano Saltori, Fabio Galasso, Giuseppe Fiameni, Nicu Sebe, Elisa Ricci, and Fabio Poiesi. CoSMix: Compositional semantic mix for domain adaptation in 3d lidar segmentation. In *ECCV*, 2022. [3](#)
- [55] Cristiano Saltori, Aljosa Osep, Elisa Ricci, and Laura Leal-Taixé. Walking your lidog: A journey through multiple domains for lidar semantic segmentation. In *ICCV*, 2023. [3](#)
- [56] Jules Sanchez, Jean-Emmanuel Deschaud, and François Goulette. Domain generalization of 3d semantic segmentation in autonomous driving. In *ICCV*, 2023. [3](#)
- [57] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *NeurIPS*, 2019. [2](#)
- [58] Corentin Sautier, Gilles Puy, Spyros Gidaris, Alexandre Boulch, Andrei Bursuc, and Renaud Marlet. Image-to-lidar self-supervised distillation for autonomous driving data. In *CVPR*, 2022. [1](#), [2](#), [3](#), [4](#), [5](#), [8](#), [12](#), [13](#), [14](#)
- [59] Corentin Sautier, Gilles Puy, Alexandre Boulch, Renaud Marlet, and Vincent Lepetit. BEVContrast: Self-supervision in bev space for automotive lidar point clouds. In *3DV*, 2023. [1](#), [2](#)

- [60] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, pages 770–779, 2019. 14
- [61] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: Point-voxel feature set abstraction for 3D object detection. In *CVPR*, 2020. 14
- [62] Oriane Siméoni, Chloé Sekkat, Gilles Puy, Antonin Vobecky, Éloi Zablocki, and Patrick Pérez. Unsupervised object localization: Observing the background to discover objects. In *CVPR*, 2023. 12
- [63] Oriane Siméoni, Gilles Puy, Huy V. Vo, Simon Roburin, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, Renaud Marlet, and Jean Ponce. Localizing objects with self-supervised transformers and no labels. In *BMVC*, 2021. 12
- [64] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *ECCV*, 2020. 7, 13
- [65] OpenPCDet Development Team. OpenPCDet: An open-source toolbox for 3D object detection from point clouds. <https://github.com/open-mmlab/OpenPCDet>, 2020. 14
- [66] Yangtao Wang, Xi Shen, Shell Xu Hu, Yuan Yuan, James L. Crowley, and Dominique Vaufreydaz. Self-supervised transformers for unsupervised object discovery using normalized cut. In *CVPR*, 2022. 12
- [67] Yanhao Wu, Tong Zhang, Wei Ke, Sabine Süsstrunk, and Mathieu Salzmann. Spatiotemporal self-supervised learning for point clouds in the wild. In *CVPR*, 2023. 1, 2
- [68] Aoran Xiao, Jiaying Huang, Weihao Xuan, Ruijie Ren, Kangcheng Liu, Dayan Guan, Abdulmotaheb El Saddik, Shijian Lu, and Eric P. Xing. 3d semantic segmentation in the wild: Learning generalized models for adverse-condition point clouds. In *CVPR*, 2023. 3
- [69] Pengchuan Xiao, Zhenlei Shao, Steven Hao, Zishuo Zhang, Xiaolin Chai, Judy Jiao, Zesong Li, Jian Wu, Kai Sun, Kun Jiang, Yunlong Wang, and Diange Yang. PandaSet: Advanced sensor suite dataset for autonomous driving. In *ITSC*, 2021. 2, 5, 12
- [70] Saining Xie, Jiatao Gu, Demi Guo, Charles R. Qi, Leonidas J. Guibas, and Or Litany. PointContrast: Unsupervised pre-training for 3D point cloud understanding. In *ECCV*, 2020. 1, 2
- [71] Xu Yan, Jiantao Gao, Chaoda Zheng, Chao Zheng, Ruimao Zhang, Shuguang Cui, and Zhen Li. 2DPASS: 2D priors assisted semantic segmentation on lidar point clouds. In *ECCV*, 2022. 7, 13
- [72] Xu Yan, Chaoda Zheng, Zhen Li, Shuguang Cui, and Dengxin Dai. Benchmarking the robustness of lidar semantic segmentation models. *IJCV*, 2024. 3
- [73] Li Yi, Boqing Gong, and Thomas Funkhouser. Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds. In *CVPR*, 2021. 3, 8
- [74] Junbo Yin, Dingfu Zhou, Liangjun Zhang, Jin Fang, Chengzhong Xu, Jianbing Shen, and Wenguan Wang. Proposal-Contrast: Unsupervised pre-training for lidar-based 3D object detection. In *ECCV*, 2022. 2
- [75] Kaicheng Yu, Tang Tao, Hongwei Xie, Zhiwei Lin, Tingting Liang, Bing Wang, Peng Chen, Dayang Hao, Yongtao Wang, and Xiaodan Liang. Benchmarking the robustness of lidar-camera fusion for 3d object detection. In *CVPR*, 2023. 3
- [76] Ping-Chung Yu, Cheng Sun, and Min Sun. Data efficient 3D learner via knowledge transferred from 2D model. In *ECCV*, 2022. 2
- [77] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-BERT: Pre-training 3D point cloud transformers with masked point modeling. In *CVPR*, 2022. 2
- [78] Renrui Zhang, Ziyu Guo, Peng Gao, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, and Hongsheng Li. Point-M2AE: multi-scale masked autoencoders for hierarchical point cloud pre-training. In *NeurIPS*, 2022. 2
- [79] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. PointCLIP: Point cloud understanding by CLIP. In *CVPR*, 2022. 2
- [80] Renrui Zhang, Lihui Wang, Yu Qiao, Peng Gao, and Hongsheng Li. Learning 3D representations from 2D pre-trained models via image-to-point masked autoencoders. In *CVPR*, 2023. 2
- [81] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. Self-supervised pretraining of 3D features on any point-cloud. In *ICCV*, 2021. 1, 2
- [82] Chong Zhou, Chen Change Loy, and Bo Dai. Extract free dense labels from CLIP. In *ECCV*, 2022. 5
- [83] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. iBOT: Image BERT pre-training with online tokenizer. In *ICLR*, 2022. 5
- [84] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *CVPR*, 2021. 1, 7, 13
- [85] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and asymmetrical 3D convolution networks for lidar segmentation. In *CVPR*, 2021. 3, 4
- [86] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Wang, Lijuan Wang, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once. In *NeurIPS*, 2023. 5, 6