TTT-KD: Test-Time Training for 3D Semantic Segmentation through Knowledge Distillation from Foundation Models

Lisa Weijler
TU Wien

M. Jehanzeb Mirza
CSAIL, MIT

Leon Sick
Ulm University

Can Ekkazan
YTU

Pedro Hermosilla
TU Wien

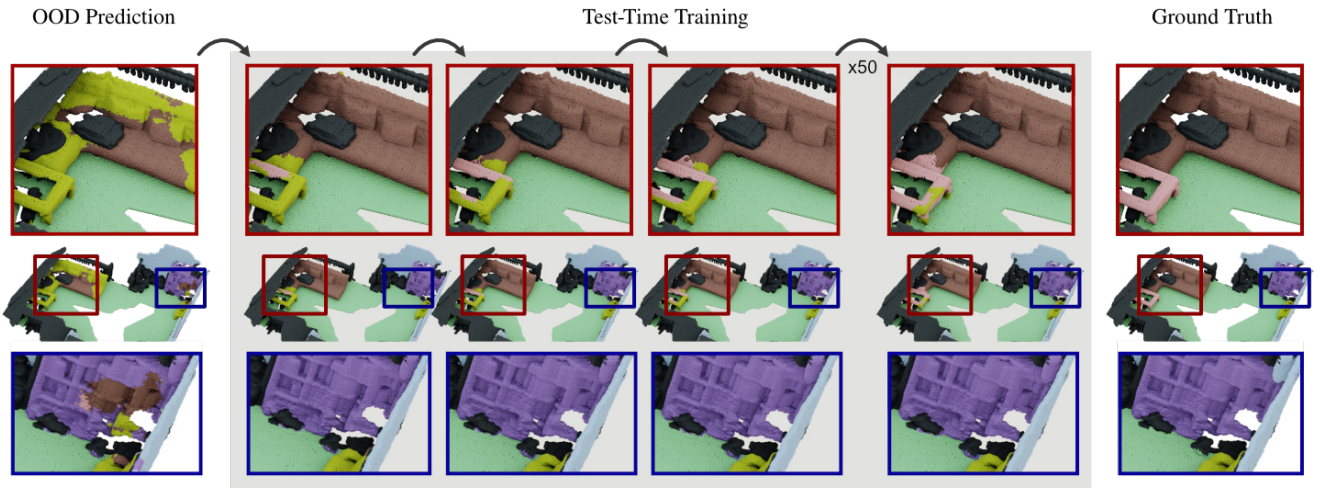

Figure 1. **TTT-KD.** We propose the first test-time training method for 3D semantic segmentation which adapts to distribution shifts at test time. As shown in the illustration above, our method is able to adapt to Out-of-Distribution (OOD) scenes (SCANNET) where the model was not trained on (S3DIS). Iteratively, via knowledge distillation from 2D foundation models, our algorithm adjusts the weights of the network, progressively improving prediction with each step **(top)**. Moreover, our approach is able to significantly improve the predictions with even a single step while maintaining the quality of those without degradation over multiple steps **(bottom)**.

Abstract

Test-Time Training (TTT) proposes to adapt a pre-trained network to changing data distributions on-the-fly. In this work, we propose the first TTT method for 3D semantic segmentation, ***TTT-KD****, which models Knowledge Distillation (KD) from foundation models (*e.g. *DINOv2) as a self-supervised objective for adaptation to distribution shifts at test-time. Given access to paired image-pointcloud (2D-3D) data, we first optimize a 3D segmentation backbone for the main task of semantic segmentation using the pointclouds and the task of 2D → 3D KD by using an off-the-shelf 2D pre-trained foundation model. At test-time, our TTT-KD updates the 3D segmentation backbone for each test sample by using the self-supervised task of knowledge distillation before performing the final prediction. Extensive evaluations on multiple indoor and outdoor 3D segmentation benchmarks show the utility of TTT-KD, as it improves performance for both in-distribution (ID) and out-of-distribution (OOD) test datasets. We achieve a gain of up to 13 % mIoU (7 % on average) when the train and test distributions are similar and up to 45 % (20 % on average) when adapting to OOD test samples. The code is available in the following repository.*

1. Introduction

3D semantic segmentation represents a fundamental benchmark for neural networks that process pointclouds [7, 16, 55, 58]. In this task, the model's primary objective is to predict the semantic label of each point in the scene. Successful execution of this task requires a profound comprehension of scene objects and their precise spatial localization. Despite the recent success obtained by different models for the task

of 3D semantic segmentation, the generalization of these models on different data sets still remains an open problem. This generalization gap can be provoked due to a variety of reasons: sensors used during acquisition, reconstruction algorithms used to obtain the pointcloud, inherent noise on the point coordinates, colors, and normals, or even the different scene compositions.

One way to bridge the domain gap is to label the pointcloud sequences from different datasets and train the network in a supervised manner on this data [59]. However, labeling can incur huge monetary costs and manual effort. To avoid these challenges, several works suggested adapting the network in an unsupervised manner to the OOD data. A popular paradigm is Unsupervised Domain Adaptation (UDA), where the network is trained jointly on the labeled source domain and unlabeled target domain, with the goal of learning an invariant feature representation for both domains. Many works [4, 42–45, 47, 62], have proposed UDA approaches for 3D semantic segmentation for outdoor pointclouds, but countering domain shifts for indoor scenes is relatively less studied. Moreover, in real-world scenarios, there could rarely be situations where the target domain is known in advance, rendering these methods unsuitable.

To forego the need for access to the target domain data, Test-Time Adaptation (TTA) algorithms [24, 50, 56] instead propose to adapt the network weights at test-time, more generally with some posthoc regularization. For effective adaptation, these works usually require constrained optimization of the network parameters, for example, only updating the affine parameters of the normalization layers. However, this could be insufficient to adapt to severe domain shifts. Moreover, these methods also rely on larger batch sizes for adaptation, making their applicability to large 3D scenes challenging.

Sharing the philosophy with TTA of adapting the network weights at test-time but differing in its application and methodology – TTT proposes to first train a network jointly for the main (downstream) task and a self-supervised auxiliary objective. At test-time, given a (single) pointcloud sample, TTT adapts the network weights *independently* for each pointcloud sample by using the self-supervised objective and then performs inference with the adapted network weights. Recently, MATE [33] proposes to use Masked Auto Encoding (MAE) [38] task as a self-supervised objective for adaptation to OOD pointclouds at test-time, for the task of pointcloud classification. However, this method was designed for the task of point cloud classification and it was only tested on synthetic domain shifts.

In this work, we propose the first TTT algorithm for the task of 3D semantic segmentation, TTT-KD, which models 2D $\rightarrow$ 3D KD from foundation models as a self-supervised objective. During training, our method receives a 3D pointcloud as input and a set of 2D images of the same scene with point-pixel correspondences. A 3D backbone processes the 3D pointcloud generating a set of 3D per-point features. These 3D features are then used to predict the semantic label of the points and also to perform 2D $\rightarrow$ 3D KD from a 2D foundation model, DINOv2 [37] in most of our experiments. At test-time, given a test pointcloud with its corresponding images, we adapt the network's weights by taking several gradient descent steps on the self-supervised task of KD. Since the 3D backbone has learned a joint feature space for the main segmentation task and the self-supervised KD task, improving predictions on the KD task leads to large improvements in the semantic segmentation task. Our algorithm does not make any assumption of the target domain, and therefore, it is able to adapt to it by processing individual scenes one at a time. Our extensive evaluation shows that our algorithm not only leads to large improvements for OOD datasets (see Fig. 1), **up to 17 points in mIoU**, but also provides significant improvements on in-distribution datasets, **up to 8.5 points in mIoU**.

2. Related work

Our TTT-KD is related to works that study Unsupervised Domain Adaptation (UDA), Test-Time Adaptation (TTA), and more closely to works that propose methodologies for Test-Time Training (TTT).

Unsupervised domain adaptation. UDA aims to train a network in order to bridge the gap between the source and target domains while having access to labeled data from the source domain and unlabeled data from the target domain. For the task of pointcloud classification, PointDAN [41] proposes to learn domain invariant features between the source and target domain with the help of adversarial feature alignment [12]. Liang *et al*. [23] propose to learn an invariant feature space for the source and target domain by using self-supervision. More specifically, they propose to predict the masked local structures by estimating cardinality, position, and normals for the points in the source and target domains, while Shen *et al*. [49] propose to use implicit functions coupled with pseudo-labeling for UDA. For 3D object detection, some approaches [31, 57] rely on statistical normalization of the anchor boxes in the source and target domain for UDA. Lehner and Gasperini *et al*. [22] propose to use adversarial augmentations, while pseudo-labeling is employed by [10, 29, 42, 61]. The task of 3D semantic segmentation has also been studied extensively in the context of UDA. Yi *et al*. [62] propose to synthesize canonical domain points, making the sparse pointcloud dense, before performing segmentation. Saltori *et al*. [45] propose a pseudo-labeling approach by mixing the source and target domains. xMUDA [20] proposes a multi-modal (two-stream) learning approach between 3D and 2D networks, they perform UDA by minimizing the discrepancy between the feature

space of the two streams and self-training through pseudo-labeling. Some other approaches also rely on a multi-modal setup and achieve UDA by increasing the number of samples used from 2D features by increasing the 3D to 2D correspondences [39], employing contrastive learning [60] or leveraging SAM [21] for obtaining reliable dense 2D annotations [4]. Although UDA offers an efficient solution for adaptation to distribution shifts, still it requires advanced knowledge about the target distribution and requires access to the unlabeled data as well. However, in real-world scenarios, such resources are often unavailable or impractical. Distribution shifts can occur *on-the-fly* and can be unpredictable. Thus, a more practicable solution is to adapt the network weights whenever changing data distributions are encountered, which is put forward by TTT.

Test-time adaptation. TTA does not alter the training procedure of the network but instead proposes post hoc regularization for adaptation to distribution shifts at test-time. For the image domain, some approaches rely on statistical correction to adapt the network at test-time, generally by adapting the means and variance estimates (of the Batch Normalization layer [18]) to the OOD test data [25, 32]. TENT [54] proposes to adapt to distribution shifts at test-time by minimizing the Shannon Entropy [48] of predictions and adapts only the scale and shift parameters of the normalization layers in the network. The problem of TENT [54] to require larger batch sizes is solved by MEMO [63], which augments a single sample multiple times and minimizes the marginal output distribution over the augmented samples. Niu *et al*. [36] proposes a sharpness-aware entropy minimization method for adaptation to distribution shifts in the wild. One group of TTA methods also rely on self-training. T3A [19] casts TTA as a prototype learning problem and replaces a classifier learned on the source dataset with pseudo-prototypes generated on the fly for the test batch. AdaContrast [6] uses self-distillation with contrastive learning and a momentum encoder to adapt to distribution shifts on the fly. Self-distillation has also been explored by other methods [51, 53] in the context of TTA. MM-TTA [50] uses 2D-3D multi-modal training for test-time adaptation but only adapts batch normalization affine parameters with a pseudo-labeling strategy. Liang *et al*. [24] also relies on pseudo-labeling and entropy minimization of individual predictions but also encourages maximizing the entropy over predicted classes over the entire dataset. CoTTA [56] also relies on pseudo-labeling and proposes continual test-time adaptation, where they learn different distribution shifts at test-time in a continual manner. Similarly, other continual TTA methods include [9, 35]. Another group of methods also relies on the consistency of predictions [3], or statistics between the train and test data distributions [26, 34]. We port AdaContrast [6], DUA [32] and TENT [54] to the task of 3D semantic segmentation and, together with MM-TTA [50], choose them as representative methods for TTA, since they cover a large variety of TTA techniques: adaptation via batch normalization statistic, adaptation by updating a reduced number of parameters, self-distillation methods, and multimodal approaches. Empirically, our TTT-KD outperforms these methods comprehensively on all the benchmarks we test on.

Test-time training. TTT first proposes to train the network jointly for the main downstream task (*e.g*., 3D Semantic Segmentation in our case) and a self-supervised objective. At test-time, it adapts the network weights for a single OOD sample as it is encountered by using the self-supervised objective – usually by taking multiple gradient steps for each sample. TTT methods are usually strictly *inductive* in nature, *i.e*., they adapt the network weights on a single sample only, whereas, TTA methods do not adhere to this restriction. For the image domain, there are two TTT works that differ in the self-supervised objectives they employ for adaptation. The first TTT [52] method (which popularized the name) employs rotation prediction [14] as its self-supervised task. Unfortunately, this approach is difficult to adapt to 3D semantic segmentation, since 3D scenes do not have a canonical orientation, and training with random SO(3) rotations usually leads to a degradation in the resulting performance. The second TTT approach, TTT-MAE [11], uses the task of image reconstruction through masked auto-encoders (MAE) [15]. MATE [33] proposes a TTT method, which also employs the MAE objective (PointMAE [38] for pointclouds) for adapting to distribution shifts in pointcloud classification.

In this paper, we propose a TTT method which models (2D $\rightarrow$ 3D) Knowledge Distillation from foundation models (*e.g*., DINOv2 [37]) as a self-supervised objective for adaptation to distribution shifts at test-time, for the task of 3D semantic segmentation. Similar to other TTT works, our TTT-KD is also strictly inductive in nature and adapts on a single pointcloud sample by performing multiple gradient steps for effective adaptation. As we will show in the experiments section, our self-supervised objective is well suited for the task of 3D semantic segmentation, outperforming other TTT methods based on a MAE objective [33].

2D $\rightarrow$ 3D knowledge distillation. KD from neural networks for images has been used in the past in the context of 3D semantic segmentation to improve ID model performance [17, 27, 28, 30, 40, 46]. In this paper, we suggest instead using KD in the context of TTT to adapt to distribution shifts on the fly by taking advantage of the generalization abilities of pre-trained vision foundation models.

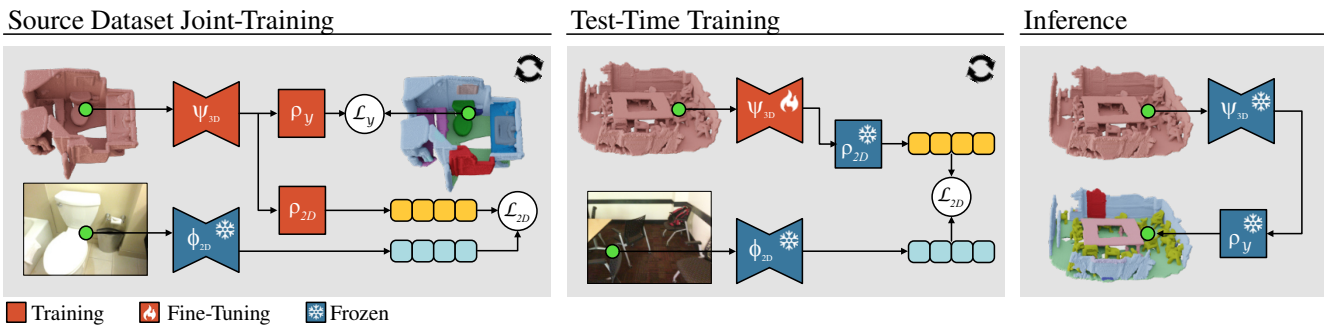

Figure 2. Given paired image-pointcloud data of a 3D scene, TTT-KD , during **joint-training**, optimizes the parameters of a point or voxel-based 3D backbone, ψ_{3D}, followed by two projectors, $\rho_{\mathcal{Y}}$ and ρ_{2D}. While $\rho_{\mathcal{Y}}$ predicts the semantic label of each point, ρ_{2D} is used for knowledge distillation from a frozen 2D foundation model, ϕ_{2D}. During **test-time training**, for each test scene, we perform several optimization steps on the self-supervised task of knowledge distillation to fine-tune the parameters of the 3D backbone. Lastly, during **inference**, we freeze all parameters of the model to perform the final prediction. By improving on the knowledge distillation task during TTT, the model adapts to out-of-distribution 3D scenes different from the source data the model was initially trained on.

3. Methods

Our algorithm jointly trains a 3D model on the semantic segmentation task and 2D $\rightarrow$ 3D KD as a secondary self-supervised task. Then, for each scene during testing, we perform a few steps of gradient descent on the KD task before we freeze the model to perform the final prediction on the segmentation task. In this section, we explain the three phases of our method: *Joint Training*, *Test-Time Training*, and *Inference* (see Fig. 2).

3.1. Input

Our method assumes as input sets of the form $(\mathcal{X}, \mathcal{F}, \mathcal{Y}, \mathcal{I}, \mathcal{U})$, where $\mathcal{X} \in \mathbb{R}^{N\times 3}$ are the spatial coordinates of the N points representing the scene, $\mathcal{F} \in \mathbb{R}^{N\times F}$ are the features associated with each point, $\mathcal{Y} \in \{0,1\}^{N\times C}$ are per-point semantic labels, $\mathcal{I} \in \mathbb{R}^{I\times W\times H\times 3}$ are a set of I images of the same scene, and $\mathcal{U} \in \mathbb{R}^{I\times N\times 2}$ are the pixel coordinates of each pair of point in $\mathcal{X}$ and image in $\mathcal{I}$. Note that not all points are projected on all images, and some points of the scene might not be projected on any image.

3.2. Joint training

3D backbone. During training, we process each point-cloud $\mathcal{X}$ with a 3D backbone ψ_{3D} to generate semantically relevant 3D features per-point, F^{3D}. Our method is agnostic to the backbone used and works, as we will show later, with voxel-based and point-based architectures.

2D foundation model. At the same time, we process all images of the 3D scene, $\mathcal{I}$, with a model ϕ_{2D} capable of generating semantically relevant 2D features, F^{2D}. This foundation model is pre-trained in a self-supervised manner on millions of images and remains fixed during the whole training procedure. As we will show in the ablation studies, our method is also agnostic to the foundation model used and can be used with any off-the-shelf foundation model.

Learning objective. Our learning objective is a multi-task objective where, from the 3D features F^{3D}, we aim to predict the semantic label of each point, $\hat{\mathcal{Y}}$, and the associated average 2D feature $\hat{F}^{2D}$ over all the images. Therefore, our algorithm minimizes a combination of two losses:

$$\mathcal{L}_{\mathcal{Y}} = \mathbb{E}_{x\sim\mathcal{X}}\left[-\sum_{c}^{C} \mathcal{Y}_{x,c}\log(\hat{\mathcal{Y}}_{x,c})\right]$$

$$\mathcal{L}_{2D} = \mathbb{E}_{x\sim\mathcal{X},i\sim\mathcal{I}}\left[-\frac{\hat{F}^{2D}_{x}}{\|\hat{F}^{2D}_{x}\|}\cdot\frac{F^{2D}_{i}(\mathcal{U}_{x,i})}{\|F^{2D}_{i}(\mathcal{U}_{x,i})\|}\right]$$

where $\mathcal{L}_{\mathcal{Y}}$ is the cross-entropy loss between the predicted labels $\hat{\mathcal{Y}}$ and the ground truth labels $\mathcal{Y}$, and $\mathcal{L}_{2D}$ is the knowledge distillation loss defined as the expected cosine similarity between the normalized per point features $\hat{F}^{2D}$ and image features F^{2D}, sampled at the pixel position defined by the mapping $\mathcal{U}$. To estimate $\mathcal{L}_{\mathcal{Y}}$ during training, we compute the average cross-entropy loss of all the points within the batch. However, since estimating $\mathcal{L}_{2D}$ is more expensive, we randomly sample points x and images i to fill a certain budget per batch.

Feature projection. In order to learn a common 3D feature space F^{3D} with these competing objectives without hampering the predictions on the main task, we transform the 3D features to $\hat{\mathcal{Y}}$ and $\hat{F}^{2D}$ with two separate projectors, $\rho_{\mathcal{Y}}$ and ρ_{2D} respectively. In practice, these projectors are two simple Multi-layer Perceptron (MLP). During training, we optimize the parameters of the 3D backbone, ψ_{3D}, and the two projectors, $\rho_{\mathcal{Y}}$ and ρ_{2D}, whilst the parameters of the foundation model, ϕ_{2D}, remain fixed.

3.3. Test-time training

Contrary to the standard testing phase in other algorithms, in which the parameters of the model are frozen, our algorithm, for each OOD scene, slightly modifies the parameters of the model before performing the final prediction. In particular, we freeze the parameters of the projectors $\rho_{\mathcal{Y}}$ and ρ_{2D}, and fine-tune all parameters of ψ_{3D} while fixing the mean and standard deviation of the batch normalization layers. In particular, we perform several gradient descent steps minimizing the knowledge distillation loss, $\mathcal{L}_{2D}$, for which no labels are required. Since both projectors have learned to perform predictions from a common feature space, F_{3D}, and both projectors aim to predict semantically relevant information, modifying these features to improve $\mathcal{L}_{2D}$ also improves the predictions on the primary segmentation task. Since we process a single scene at a time, contrary to existing test-time adaptation approaches that rely on large batches, we do not update the mean and standard deviation of the batch normalization layers. Therefore, we are not forced to synthetically increase the batch size with data augmentations, which might be prohibitive for large scenes composed of millions of points.

3.4. Inference

Once the test-time training phase has finished, we freeze all parameters of our model and perform the final prediction on the segmentation task. Following previous works [33, 52], we experiment with two variants of our method:

Offline (TTT-KD). In this setup, we perform several gradient descent steps for each test scene independently. Once the TTT phase has finished, we predict the per-point class for the current scene and then we discard the parameter updates before processing the next test scene.

Online (TTT-KD-O). In this setup, we only perform one optimization step for each test scene but we keep the parameter updates between consecutive scenes. Although this approach does not fully adapt to a single scene, it requires less computational resources while, as we will show later, achieving significant improvements over the baselines.

4. Results

In this section, we describe the experiments carried out to validate our methods. In particular, we tested our TTT-KD algorithm on two different 3D semantic segmentation setups: indoor and outdoor 3D semantic segmentation. While indoor 3D semantic segmentation provides an ideal setup for our algorithm, in which each pointcloud is paired with multiple images, the outdoor 3D semantic segmentation experiment presents a more challenging setup in which only a single image is paired with each pointcloud.

4.1. Indoor 3D semantic segmentation

In this section, first, we describe the datasets used, then our experimental setup, and lastly, the results.

4.1.1 Datasets

In our experiments, we use three different datasets of real indoor 3D scenes, SCANNET [8], S3DIS [1], and MATTERPORT3D [5]. These datasets are composed of several 3D scans of rooms from different buildings for which the reconstructed 3D pointcloud and a set of 2D images per pointcloud are available. We follow the standard train, validation, and test splits of the datasets in our experiments. For each point in the 3D scan, 3D coordinates, $[x, y, z]$, its normal, $[n_x, n_y, n_z]$, and color, $[r, g, b]$, are used as input.

4.1.2 Experimental setup

In this section, we describe our experimental setup. Additional details are provided in the supplementary material.

Tasks. We focus on two types of evaluations for our TTT-KD: ID and OOD. For ID evaluation, we train a model on the train split of a dataset and perform TTT on the test split on the same dataset, while for OOD, TTT is performed on the test split of all other datasets in our evaluation setup. We report results by using the mean Intersection over Union (mIoU) evaluation metric for semantic segmentation. For OOD evaluations, since different datasets differ on the semantic labels used, we evaluate only the classes in which both the train and test datasets share.

Models. Our experiments use two different 3D backbones: a voxel-based and a point-based architecture. As our voxel-based backbone, we choose the commonly used Minkowski34C [7]. The point-based backbone is taken from Hermosilla *et al*. [16] which is based on kernel point convolutions with Gaussian correlation functions. As our foundation model, we use DINOv2 [37], in particular, the ViT-L/14 model with an embedding size of 1024 features.

Testing. During training, we randomly rotate scenes along the up vector. Therefore, during testing, we accumulate the logits over 8 predictions of the same scene but rotated with different angles, covering 360 degrees. In the TTT phase, we use Stochastic Gradient Descent (SGD) without momentum, a learning rate of 1, and perform 100 optimization steps for each rotated scene on the offline version of our algorithm, but only one for the online version.

Baselines. We train our models on the main segmentation task, *Source-Only* , and also use knowledge distillation as

a secondary objective, *Joint-Train* . We compare their performance to the offline version of our algorithm, *TTT-KD* , and the online version, *TTT-KD-O* . Since, in the literature, there is no TTT method proposed for the task of 3D semantic segmentation, we port several TTA works from the image domain and a TTT method from the 3D shape classification field. Specifically, we compare our method with *TENT* [54], *DUA* [32], *AdaContrast* [6], and *TTT-MAE* [33]. While TENT uses entropy minimization at test-time to optimize affine parameters of batch normalization layers, DUA only updates the mean and standard deviation of these batch normalization layers. AdaContrast , on the other hand, leverages self-distillation under different degrees of data augmentations together with contrastive learning and momentum encoder to adapt the parameters of the model during testing. Additionally, TTT-MAE uses MAE as the self-supervised task in a TTT setup. Lastly, for OOD experiments, as an upper bound, we provide the performance of a model that has been trained on the same dataset as the test set on the shared classes tested in the experiment, *Oracle* .

4.1.3 Results

The main experimental results and comparisons with all other methods are provided in Tab. 1. In the following, we explain these results in detail.

Joint-training. First, we compare the performance of a Source-Only model with our Joint-Train strategy. Our results in Tab. 1 show that for all datasets and both 3D backbones, joint training always provides an improvement over the Source-Only model. In some cases, this improvement is minor, such as in SCANNET or MATTERPORT3D with an improvement of 0.8 mIoU, but for other datasets the improvement is larger, as in S3DIS or in MATTERPORT3D $\rightarrow$ SCANNET with an improvement of 2.2 and 3.6 mIoU respectively. We conjecture that the reason for the improvement is the KD task acts as an additional regularizer.

In-distribution. When testing on ID data, our algorithm provides significant improvements for all three datasets and all 3D backbones. Our algorithm presents an improvement of 2.9 and 3.7 for SCANNET , of 8.5 and 5.6 for S3DIS , and 3.5 and 2.7 for MATTERPORT3D . Moreover, although smaller, the online version of our algorithm, TTT-KD-O , also presents significant gains on all datasets.

Out-of-distribution. When we look at the performance of the Source-Only models when tested on OOD data, as expected, the performance drops significantly when compared with an Oracle model trained on ID data, with large drops in performance as in MATTERPORT3D $\rightarrow$ SCANNET with 24.5 or in S3DIS $\rightarrow$ SCANNET with 30.1. Our TTT-KD algorithm, on the other hand, is able to reduce this gap, increasing significantly the performance of all models and even obtaining better performance than the Oracle model as in the SCANNET $\rightarrow$ MATTERPORT3D experiment. Again, our online version, TTT-KD-O , also provides significant improvements but is smaller than our offline version.

Comparison to baselines. When compared to TENT , DUA , and AdaContrast , although these baselines can provide some adaptation, TTT-KD has a clear advantage, surpassing them by a large margin. We can see that TENT is not suited for the task of semantic segmentation, since it does not provide improvement in many of the configurations. We hypothesize this is due to the mean and standard deviation of the batch normalization layers, which TENT computes independently for each test batch. Since we are testing each scene independently, these estimates are not representative of the OOD data, leading to a degradation of performance. We can also see that DUA performs better than TENT , since it accumulates these parameters over several scenes, but still fails for some configurations. Lastly, we can see that the more complex method, AdaContrast , performs better than both DUA and TENT but falls behind our TTT-KD . Fig. 3 presents some qualitative results of these methods.

Importance of self-supervised task. When we substitute our KD task with another self-supervised task, TTT-MAE , we can see that the model is not able to provide the same level of adaptation as our TTT-KD , highlighting the importance of KD in the adaptation process.

Backbone agnostic. When we analyze the performance of our method on different backbones, we see a consistent improvement in all setups. This indicates that our method is independent of the 3D backbone used.

4.2. Outdoor 3D semantic segmentation

In this section, first, we briefly describe the datasets used, then the experimental setup, and lastly, the results.

4.2.1 Datasets

In our experiments, we use two different autonomous driving datasets of real outdoor 3D scenes, A2D2 [13] and SEMANTICKITTI [2]. While the 3D pointclouds are obtained with LiDAR scans, the images are obtained from different cameras mounted on the vehicle. Following Jaritz *et al*. [20], only the 2D images obtained from the front camera of the vehicle are used, and the 3D pointcloud is cropped by selecting only visible points from this camera. For each point in the 3D pointcloud, only 3D coordinates are used.

Table 1. Our method achieves large improvements not only on OOD data (□) but also on ID setups (□), surpassing existing methods by a large margin. Moreover, these results show that our algorithm is backbone agnostic, achieving comparable results for a point-based backbone, PNE [16], and a voxel-based backbone, Mink [7].

Train	Method	Test					
		ScanNet		S3DIS		Matterport3D	
		PNE [16]	Mink [7]	PNE [16]	Mink [7]	PNE [16]	Mink [7]
ScanNet	Oracle	–	–	75.7	77.4	53.9	52.3
	Source-Only	73.5	72.9	65.8	67.3	49.1	48.3
	Joint-Train	74.3 ↑ 0.8	73.9 ↑ 1.0	66.5 ↑ 0.7	71.5 ↑ 4.2	50.4 ↑ 1.3	48.7 ↑ 0.4
	TENT [54]	71.1 ↓ 2.4	68.8 ↓ 4.1	46.8 ↓ 19.0	70.5 ↑ 3.2	47.1 ↓ 2.0	44.1 ↓ 4.2
	DUA [32]	73.9 ↑ 0.4	73.1 ↑ 0.2	64.0 ↓ 1.5	70.6 ↑ 3.3	48.9 ↓ 0.2	46.9 ↓ 1.4
	AdaContrast [6]	73.8 ↑ 0.3	72.4 ↓ 0.5	67.2 ↑ 1.4	72.3 ↑ 5.1	50.2 ↑ 1.1	48.4 ↑ 0.1
	TTT-MAE [33]	74.1 ↑ 0.6	73.9 ↑ 1.0	69.4 ↑ 3.6	73.5 ↑ 6.2	49.2 ↑ 0.1	46.7 ↓ 1.6
	TTT-KD-O	75.5 ↑ 2.0	74.7 ↑ 1.8	**72.4** ↑ **6.6**	**73.7** ↑ **6.4**	53.6 ↑ 4.5	51.3 ↑ 3.0
	TTT-KD	**76.4** ↑ **2.9**	**76.6** ↑ **3.7**	70.4 ↑ 4.6	73.1 ↑ 5.8	**56.6** ↑ **7.5**	**55.3** ↑ **7.0**
S3DIS	Oracle	84.6	84.2	–	–	64.9	66.0
	Source-Only	54.5	54.9	63.2	65.9	46.1	42.1
	Joint-Train	55.5 ↑ 1.0	56.1 ↑ 1.2	65.4 ↑ 2.2	66.8 ↑ 0.9	47.0 ↑ 0.9	42.8 ↑ 0.7
	TENT [54]	56.0 ↑ 1.5	54.6 ↓ 0.3	53.0 ↓ 10.2	66.1 ↑ 0.2	45.6 ↓ 0.5	43.4 ↑ 1.3
	DUA [32]	59.0 ↑ 4.5	57.6 ↑ 2.7	67.3 ↑ 4.1	65.5 ↓ 0.4	46.7 ↑ 0.6	44.1 ↑ 2.0
	AdaContrast [6]	58.0 ↑ 3.5	57.5 ↑ 2.6	65.4 ↑ 2.2	65.6 ↑ 0.3	46.7 ↑ 0.6	46.4 ↑ 4.3
	TTT-MAE [33]	58.5 ↑ 4.0	57.2 ↑ 2.3	64.1 ↑ 0.9	65.4 ↓ 0.5	45.1 ↓ 1.0	41.8 ↓ 0.3
	TTT-KD-O	65.0 ↑ 10.5	64.1 ↑ 9.2	68.8 ↑ 5.6	68.7 ↑ 2.8	50.1 ↑ 4.0	49.2 ↑ 7.1
	TTT-KD	**69.9** ↑ **14.4**	**68.4** ↑ **13.5**	**71.7** ↑ **8.5**	**71.5** ↑ **5.6**	**53.2** ↑ **7.1**	**50.9** ↑ **8.8**
Matt3D	Oracle	73.5	72.9	77.9	78.5	–	–
	Source-Only	49.0	45.4	59.2	58.6	55.2	53.8
	Joint-Train	52.6 ↑ 3.6	50.6 ↑ 5.2	59.9 ↑ 0.7	63.7 ↑ 5.1	56.0 ↑ 0.8	55.4 ↑ 1.6
	TENT [54]	50.3 ↑ 1.3	47.5 ↑ 2.1	52.3 ↓ 6.9	65.0 ↑ 6.4	54.1 ↓ 1.1	54.7 ↑ 0.9
	DUA [32]	52.7 ↑ 3.7	50.7 ↑ 5.3	64.5 ↑ 5.3	63.8 ↑ 5.2	56.0 ↑ 0.8	54.7 ↑ 0.9
	AdaContrast [6]	55.2 ↑ 6.2	53.4 ↑ 8.0	60.8 ↑ 1.6	69.7 ↑ 11.1	56.0 ↑ 0.8	54.3 ↑ 0.5
	TTT-MAE [33]	51.1 ↑ 2.1	43.4 ↓ 2.0	63.1 ↑ 3.9	64.9 ↑ 6.3	54.5 ↓ 0.7	53.2 ↓ 0.6
	TTT-KD-O	59.4 ↑ 10.4	57.6 ↑ 12.2	64.4 ↑ 5.2	70.9 ↑ 12.3	57.8 ↑ 2.6	56.2 ↑ 2.4
	TTT-KD	**64.0** ↑ **15.0**	**62.6** ↑ **17.2**	**66.8** ↑ **7.6**	**75.4** ↑ **16.8**	**58.7** ↑ **3.5**	**56.5** ↑ **2.7**

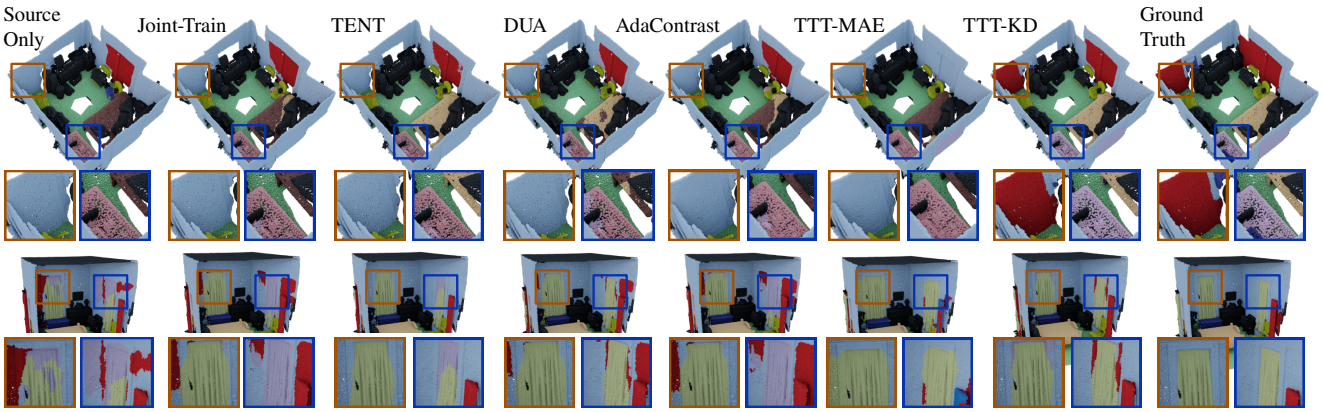

Figure 3. Qualitative results for two different OOD tasks. The top row presents results for Matterport3D → ScanNet , while the bottom row presents results for ScanNet → Matterport3D . Although other methods are able to slightly adapt to the domain shifts, our TTT-KD algorithm provides more accurate predictions.

Table 2. Results for the outdoor 3D semantic segmentation tasks. Our TTT-KD algorithm significantly reduces the domain gap for OOD compared to other methods.

	A2D2 → KITTI
Oracle	73.8
Source-Only	35.8
Joint-Train	41.6 ↑ 5.8
TENT [54]	36.6 ↑ 0.8
DUA [32]	35.5 ↓ 0.3
AdaContrast [6]	40.3 ↑ 4.5
MM-TTA [50]	42.5 ↑ 6.7
TTT-MAE [33]	39.1 ↑ 3.3
TTT-KD-O	**52.0** ↑ **16.2**
TTT-KD	49.7 ↑ 13.9

4.2.2 Experimental setup

For the task of outdoor 3D semantic segmentation, we use the same experimental setup as other UDA (xMUDA [20]) and TTA methods (MM-TTA [50]). For additional details, we refer the reader to Jaritz *et al*. [20].

Tasks. We focus on a well-established and challenging task to measure the robustness of a model to OOD data using mIoU as our metric. In this task, we train a model on the training set of the A2D2 dataset and perform TTT on the test set of the SEMANTICKITTI . Since the LiDAR scan used in the target domain is of higher resolution than the one used in the source domain, this task aims to measure the robustness of the 3D model to OOD data.

Model. As our 3D backbone, we use the same sparse convolution architecture as previous work [20, 50]. As our foundation model, we use again DINOv2 [37].

Testing. For testing, we use the same configuration as in the indoor 3D semantic segmentation tasks. However, due to the large size of the dataset, we reduce the number of rotations to 4, and the number of TTT of our offline version to 25. Moreover, we reduce the learning rate to 0.1.

Baselines. In this experiment, we use the same baselines as in the indoor setup. Additionally, we also compare to the 3D backbone of the 2D-3D multi-modal TTA method *MM-TTA* [50]. Unfortunately, since no implementation is available for this method, we were not able to include it in our indoor experiments.

4.2.3 Results

The main results of this experiment are presented on Tab. 2. Further, we analyze these results in detail.

Joint-training. As in the indoor tasks, our Joint-Train strategy provides a significant improvement over the Source-Only model. Moreover, we can see that it is able to match and even surpass most of the baselines without performing any adaptation during testing, confirming that our KD secondary task acts as a regularizer, improving the generalization of the model.

Out-of-distribution. When we analyze the performance of the Source-Only models when tested on OOD, we see again a significant performance drop when compared with an Oracle model trained on ID data. Our TTT-KD algorithm, on the other hand, presents a large performance increase when compared to the Source-Only . The domain gap is reduced even more by our TTT-KD-O , achieving an increase of 45 % mIoU. We hypothesize that this is due to the reduced number of TTT iterations of our offline version when compared to the number of iterations used previously.

Comparison to baselines. When we compare our TTT-KD and TTT-KD-O algorithms to the baselines, we can see similar results to the ones obtained for the indoor setup, where our methods surpass them by a large margin. Additionally, when compared to the TTA method for outdoor 3D semantic segmentation, MM-TTA , we can see that our TTT-KD and TTT-KD-O algorithms also surpass it by almost 10 mIoU points.

4.3. Ablation studies

In the supplementary materials, together with additional experiments and a discussion of the limitations of our method, we provide extensive ablation studies to investigate the effect of different design choices. In particular, we investigate the effect of the number of TTT steps, the effect of the number of available point-image pairs, the effect of the foundation model used, the effect of incorporating a stride in the online version of our algorithm, the effect of incorporating momentum in the SGD optimizer, and analyze the computational cost of our TTT.

5. Conclusions

Our TTT-KD is the first test-time training method proposed for the task of 3D semantic segmentation, which proposes to use knowledge distillation from foundation models as a self-supervised auxiliary objective to adapt the network weights individually for each test sample as it is encountered. Our experiments show that TTT-KD can be used with any off-the-shelf foundation model and multiple different 3D backbones. Furthermore, our method provides impressive performance gains while adapting to both in-distribution and out-of-distribution test samples when evaluated on multiple different benchmarks.

References

[1] I. Armeni, S. Sax, A. R Zamir, and S. Savarese. Joint 2d-3d-semantic data for indoor scene understanding, 2017. 5

[2] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019. 6

[3] Malik Boudiaf, Romain Mueller, Ismail Ben Ayed, and Luca Bertinetto. Parameter-free online test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8344–8353, 2022. 3

[4] Haozhi Cao, Yuecong Xu, Jianfei Yang, Pengyu Yin, Shenghai Yuan, and Lihua Xie. Mopa: Multi-modal prior aided domain adaptation for 3d semantic segmentation, 2024. 2, 3

[5] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017. 5

[6] Dian Chen, Dequan Wang, Trevor Darrell, and Sayna Ebrahimi. Contrastive test-time adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022. 3, 6, 7, 8

[7] C. Choy, J. Y. Gwak, and S. Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1, 5, 7

[8] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *Proc. CVPR*, 2017. 5

[9] Mario Döbler, Robert A Marsden, and Bin Yang. Robust mean teacher for continual and gradual test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7704–7714, 2023. 3

[10] Christian Fruhwirth-Reisinger, Michael Opitz, Horst Possegger, and Horst Bischof. FAST3D: Flow-Aware Self-Training for 3D Object Detectors. In *Proc. BMVC*, 2021. 2

[11] Yossi Gandelsman, Yu Sun, Xinlei Chen, and Alexei Efros. Test-time training with masked autoencoders. *Advances in Neural Information Processing Systems*, 35:29374–29385, 2022. 3

[12] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. *JMLR*, 17(59):1–35, 2016. 2

[13] Jakob Geyer, Yohannes Kassahun, Mentar Mahmudi, Xavier Ricou, Rupesh Durgesh, Andrew S. Chung, Lorenz Hauswald, Viet Hoang Pham, Maximilian Mühlegg, Sebastian Dorn, Tiffany Fernandez, Martin Jänicke, Sudesh Mirashi, Chiragkumar Savani, Martin Sturm, Oleksandr Vorobiov, Martin Oelker, Sebastian Garreis, and Peter Schuberth. A2D2: audi autonomous driving dataset, 2020. 6

[14] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018. 3

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 3

[16] P. Hermosilla. Point neighborhood embeddings, 2023. 1, 5, 7

[17] Wenbo Hu, Hengshuang Zhao, Li Jiang, Jiaya Jia, and Tien-Tsin Wong. Bidirectional projection network for cross dimensional scene understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2021. 3

[18] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proc. ICML*, 2015. 3

[19] Yusuke Iwasawa and Yutaka Matsuo. Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34:2427–2440, 2021. 3

[20] M. Jaritz, T.-H. Vu, R. de Charette, E. Wirbel, and P. Pérez. xMUDA: Cross-modal unsupervised domain adaptation for 3D semantic segmentation. In *Proc. CVPR*, 2020. 2, 6, 8

[21] 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:2304.02643*, 2023. 3

[22] Alexander Lehner, Stefano Gasperini, Alvaro Marcos-Ramiro, Michael Schmidt, Mohammad-Ali Nikouei Mahani, Nassir Navab, Benjamin Busam, and Federico Tombari. 3d-vfield: Adversarial augmentation of point clouds for domain generalization in 3d object detection. In *Proc. CVPR*, 2022. 2

[23] Hanxue Liang, Hehe Fan, Zhiwen Fan, Yi Wang, Tianlong Chen, Yu Cheng, and Zhangyang Wang. Point cloud domain adaptation via masked local 3d structure prediction. In *Proc. ECCV*. Springer, 2022. 2

[24] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International conference on machine learning*, pages 6028–6039. PMLR, 2020. 2, 3

[25] Hyesu Lim, Byeonggeun Kim, Jaegul Choo, and Sungha Choi. Ttn: A domain-shift aware batch normalization in test-time adaptation. *arXiv preprint arXiv:2302.05155*, 2023. 3

[26] Wei Lin, Muhammad Jehanzeb Mirza, Mateusz Kozinski, Horst Possegger, Hilde Kuehne, and Horst Bischof. Video Test-Time Adaptation for Action Recognition. In *Proc. CVPR*, 2023. 3

[27] 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3

[28] 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:2104.04687*, 2021. 3

[29] Zhipeng Luo, Zhongang Cai, Changqing Zhou, Gongjie Zhang, Haiyu Zhao, Shuai Yi, Shijian Lu, Hongsheng Li, Shanghang Zhang, and Ziwei Liu. Unsupervised Domain Adaptive 3D Detection with Multi-Level Consistency. In *Proc. CVPR*, 2021. 2

[30] Anas Mahmoud, Jordan Hu, Tianshu Kuai, Ali Harakeh, Liam Paull, and Steven Waslander. Self-supervised image-to-point distillation via semantically tolerant contrastive loss. 2023. 3

[31] Dušan Malić, Christian Fruhwirth-Reisinger, Horst Possegger, and Horst Bischof. Sailor: Scaling anchors via insights into latent object representation. In *Proc. WACV*, 2023. 2

[32] M. Jehanzeb Mirza, Jakub Micorek, Horst Possegger, and Horst Bischof. The Norm Must Go On: Dynamic Unsupervised Domain Adaptation by Normalization. In *Proc. CVPR*, 2022. 3, 6, 7, 8

[33] M. J. Mirza, I. Shin, W. Lin, A. Schriebl, K. Sun, J. Choe, M. Kozinski, H. Possegger, I. S. Kweon, K.-J. Yoon, and H. Bischof. Mate: Masked autoencoders are online 3d test-time learners. *Proc. ICCV*, 2023. 2, 3, 5, 6, 7, 8

[34] M. Jehanzeb Mirza, Pol Jane Soneira, Wei Lin, Mateusz Kozinski, Horst Possegger, and Horst Bischof. ActMAD: Activation Matching to Align Distributions for Test-Time Training. In *Proc. CVPR*, 2023. 3

[35] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yaofo Chen, Shijian Zheng, Peilin Zhao, and Mingkui Tan. Efficient Test-Time Model Adaptation without Forgetting. In *Proc. ICML*, 2022. 3

[36] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. In *Internetional Conference on Learning Representations*, 2023. 3

[37] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 2, 3, 5, 8

[38] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pages 604–621. Springer, 2022. 2, 3

[39] D. Peng, Y. Lei, W. Li, P. Zhang, and Y. Guo. Sparse-to-dense feature matching: Intra and inter domain cross-modal learning in domain adaptation for 3d semantic segmentation. In *Proc. ICCV*, 2021. 3

[40] Gilles Puy, Spyros Gidaris, Alexandre Boulch, Oriane Siméoni, Corentin Sautier, Patrick Pérez, Andrei Bursuc, and Renaud Marlet. Three pillars improving vision foundation model distillation for lidar. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[41] Can Qin, Haoxuan You, Lichen Wang, C-C Jay Kuo, and Yun Fu. PointDAN: A Multi-Scale 3D Domain Adaption Network for Point Cloud Representation. In *NeurIPS*, 2019. 2

[42] C. Saltori, S. Lathuilière, N. Sebe, E. Ricci, and F. Galasso. Sf-uda3d: Source-free unsupervised domain adaptation for lidar-based 3d object detection. In *Proc. i3dv*, 2020. 2

[43] C. Saltori, F. Galasso, G. Fiameni, N. Sebe, E. Ricci, and F. Poiesi. Cosmix: Compositional semantic mix for domain adaptation in 3d lidar segmentation. In *Proc. ECCV*, 2022.

[44] 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. Springer, 2022.

[45] C. Saltori, F. Galasso, G. Fiameni, N. Sebe, F. Poiesi, and E. Ricci. Compositional semantic mix for domain adaptation in point cloud segmentation. *IEEE TPAMI*, 2023. 2

[46] Corentin Sautier, Gilles Puy, Spyros Gidaris, Alexandre Boulch, Andrei Bursuc, and Renaud Marlet. Image-to-lidar self-supervised distillation for autonomous driving data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[47] A. Shaban, J. Lee, S. Jung, X. Meng, and B. Boots. Lidar-uda: Self-ensembling through time for unsupervised lidar domain adaptation. In *Proc. ICCV*, 2023. 2

[48] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. 3

[49] Yuefan Shen, Yanchao Yang, Mi Yan, He Wang, Youyi Zheng, and Leonidas J Guibas. Domain Adaptation on Point Clouds via Geometry-Aware Implicits. In *Proc. CVPR*, 2022. 2

[50] I. Shin, Y.-H. Tsai, B. Zhuang, S. Schulter, B. Liu, S. Garg, I. S. Kweon, and K.-J. Yoon. Mm-tta: Multi-modal test-time adaptation for 3d semantic segmentation. In *Proc. CVPR*, 2022. 2, 3, 8

[51] Samarth Sinha, Peter Gehler, Francesco Locatello, and Bernt Schiele. Test: Test-time self-training under distribution shift. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023. 3

[52] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR, 2020. 3, 5

[53] Devavrat Tomar, Guillaume Vray, Behzad Bozorgtabar, and Jean-Philippe Thiran. Tesla: Test-time self-learning with automatic adversarial augmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2023. 3

[54] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully Test-time Adaptation by Entropy Minimization. In *Proc. ICLR*, 2020. 3, 6, 7, 8

[55] Peng-Shuai Wang. Octformer: Octree-based transformers for 3d point clouds. *ACM Transactions on Graphics (SIGGRAPH)*, 2023. 1

[56] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7201–7211, 2022. 2, 3

[57] Yan Wang, Xiangyu Chen, Yurong You, Li Erran Li, Bharath Hariharan, Mark Campbell, Kilian Q Weinberger, and Wei-Lun Chao. Train in Germany, Test in The USA: Making 3D Object Detectors Generalize. In *Proc. CVPR*, 2020. 2

[58] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 1

[59] X. Wu, Z. Tian, X. Wen, B. Peng, X. Liu, K. Yu, and H. Zhao. Towards large-scale 3d representation learning with multi-dataset point prompt training, 2023. 2

[60] B. Xing, X. Ying, R. Wang, J. Yang, and T. Chen. Cross-modal contrastive learning for domain adaptation in 3d semantic segmentation. In *Proc. AAAI*, 2023. 3

[61] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. ST3D: Self-training for Unsupervised Domain Adaptation on 3D Object Detection. In *Proc. CVPR*, 2021. 2

[62] L. Yi, B. Gong, and T. Funkhouser. Complete & label: A domain adaptation approach to semantic segmentation of lidar point clouds. In *Proc. CVPR*, 2021. 2

[63] Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. *Advances in Neural Information Processing Systems*, 35: 38629–38642, 2022. 3