Adapt-As-You-Walk Through the Clouds: Training-Free Online Test-Time Adaptation of 3D Vision-Language Foundation Models

Mehran Tamjidi [*1], **Hamidreza Dastmalchi** [*2], **Mohammadreza Alimoradijazi**[3], **Ali Cheraghian**[4], **Aijun An**[2], **Morteza Saberi**[1]

[1]School of Computer Science, University of Technology Sydney, Sydney, Australia
[2]Department of Electrical Engineering and Computer Science, York University, Toronto, Canada
[3]Business school, The University of New South Wales, Sydney, Australia,
[4]School of Engineering, Macquarie University, Sydney, Australia
[mehran.tamjidi, morteza.saberi]@uts.edu.au, [hrd, aan]@yorku.ca, reza.moradi@unsw.edu.au, ali.cheraghian@mq.edu.au

Abstract

3D Vision-Language Foundation Models (VLFMs) have demonstrated strong generalization and zero-shot recognition capabilities in open-world point cloud processing tasks. However, their performance often degrades in practical scenarios where data are noisy, incomplete, or drawn from distributions that differ from the training data. To address this challenge, we propose Uni-Adapter, a novel training-free online test-time adaptation (TTA) strategy for 3D VLFMs based on dynamic prototype learning. Uni-Adapter maintains a 3D cache that stores class-specific cluster centers as prototypes, which are continuously updated to capture intra-class variability under heterogeneous data distributions. These dynamic prototypes serve as anchors for cache-based logit computation through similarity scoring. In parallel, a graph-based label smoothing module models inter-prototype similarities to enforce label consistency among related prototypes. Finally, predictions from the original 3D VLFM and the refined 3D cache are unified through entropy-weighted aggregation to ensure reliable adaptation. Without retraining, Uni-Adapter effectively mitigates distribution shifts and achieves state-of-the-art performance across diverse 3D benchmarks and multiple 3D VLFMs, improving performance on ModelNet-40C by 10.55%, ScanObjectNN-C by 8.26%, and ShapeNet-C by 4.49% over the source 3D VLFMs.

Code — https://mehran-tam.github.io/Uni-Adapter

Introduction

3D Vision-Language Foundation Models (VLFMs), such as Uni3D (Zhou et al. 2024), have introduced remarkable potential in multimodal point cloud processing tasks. Pre-trained on web-scale text-image-point cloud triplets, these models learn cross-modal representations in a shared embedding space, enabling zero-shot recognition of novel point cloud categories. Despite the strength of these models, VLFMs encounter critical limitations in real-world scenarios where acquired point clouds often suffer from severe noise, sparsity, and low resolution due to sensor constraints and environmental factors. Domain adaptation and generalization aim to address these distribution shifts by bridging the gap between source and target domains.

Among existing adaptation approaches for VLFMs, test-time adaptation (TTA) (Sharifdeen et al. 2025; Karmanov et al. 2024; Huang et al. 2025) offers a particularly effi-

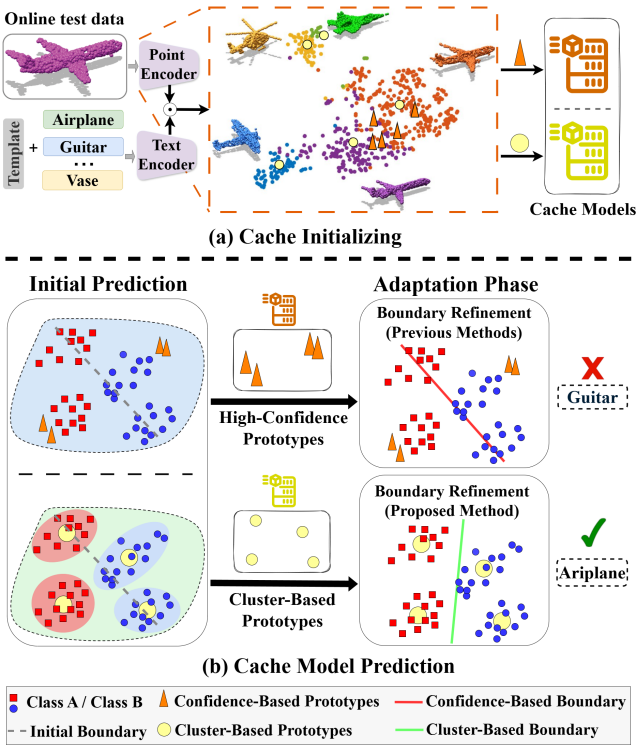

Figure 1: (a) t-SNE of Uni3D embeddings for the airplane class in ModelNet40-C shows clear intra-class clustering patterns. Confidence-based prototypes (triangles) cache only high-confidence samples, while cluster-based prototypes (circles) represent distribution modes via online clustering. (b) In the toy example, confidence-based caching leads to incorrect boundaries due to poor mode coverage, whereas cluster-based caching captures diverse patterns and enables correct predictions.

[*]These authors contributed equally.

cient solution, requiring no labeled target data while enabling dynamic adjustment to unseen conditions. Existing TTA methods for VLFMs can be broadly categorized into training-based and training-free approaches. Training-based TTA methods adapt to the target domain by updating a subset of model parameters (Osowiechi et al. 2024) or soft prompts (Shu et al. 2022; Yoon et al. 2024; Sharifdeen et al. 2025) at test time. These methods typically optimize objectives like prediction entropy minimization across augmented views of test samples (Shu et al. 2022), or employ additional auxiliary losses (Sharifdeen et al. 2025) to guide adaptation. While effective in reducing domain shift, these methods often require iterative backpropagation, making them computationally demanding and less suited for real-time deployment. In contrast, training-free TTA methods, such as recent cache-based approaches (Karmanov et al. 2024; Sun et al. 2025), avoid parameter tuning by dynamically caching high-confidence features. These embeddings refine predictions via feature similarity, enabling lightweight, scalable adaptation for real-time and streaming scenarios.

While mainly designed for 2D VLFMs (Radford et al. 2021a), cache-based strategies are underexplored in 3D VLFMs, with only a few early attempts to design effective cache modules. Recently, Point-Cache (Sun et al. 2025) introduced a TTA framework for 3D VLFMs, proposing a dual-cache structure of global and local caches. Both caches are built from high-confidence test samples, assuming these samples sufficiently represent the full data distribution. However, this assumption is often violated in practice—particularly for 3D data—where each semantic class can exhibit significant structural diversity. As illustrated in Figure 1(a), features corresponding to a single class (e.g., "airplane") form multiple distinct clusters in the feature space, reflecting different structural modes. Consequently, high-confidence prototypes typically capture only a subset of these variations, leading to suboptimal adaptation performance. This limitation is shown in the top of Figure 1(b), where cached high-confidence prototypes (triangular markers) lead to incorrect decision boundaries.

To overcome the limitations of prior confidence-based cache strategies, we propose Uni-Adapter (Unified 3D Adapter), a novel online TTA framework for 3D VLFMs. Our approach employs a cluster-based caching strategy that dynamically stores and updates cluster centers, ensuring comprehensive coverage of the underlying feature distribution. This design is visualized in the lower part of Figure 1(b), where yellow circular markers denote the cached cluster centers used as prototypes. These prototypes offer a more faithful representation of the distribution, leading to improved affinity calculations and more robust adaptation.

To enable clustering-based prototyping in the test-time setting, we adopt an online clustering strategy, where incoming test samples incrementally update class-specific cluster centers, serving as prototypes. Each class maintains multiple cluster-based prototypes within a unified cache, ensuring comprehensive coverage of diverse data distribution modes. This strategy captures intra-class variability and prevents over-reliance on a few dominant patterns. Moreover, we observe that the performance of existing cache-based models is affected by noisy pseudo-labels, allowing misclassified samples to contaminate the cache. To address this issue, we construct a similarity graph over cached prototypes and apply graph-based label smoothing to refine their labels. This enables effective label propagation across similar prototypes, mitigating noisy pseudo-labels and yielding a more reliable, adaptive cache. We solve the resulting Laplacian system using the conjugate gradient method for its efficiency and scalability for large, sparse systems. Finally, we fuse the original 3D VLFM scores and 3D cache logits using entropy-driven confidence weighting to derive the final prediction.

In summary, the contributions of the proposed method are as follows: **1)** We introduce a cluster-based caching strategy that employs multiple cluster centers per class to capture intra-class variability, enabling adaptation to diverse test distributions. **2**) We apply graph-based label smoothing over cache prototypes, using inter-prototype similarities to refine noisy pseudo-labels and improve cache-based adaptation under distribution shift. **3**) We conduct extensive experiments to validate our approach on different 3D VLFMs across diverse benchmarks—including corrupted datasets such as ShapeNet-C (Mirza et al. 2023), ModelNet-40C (Sun et al. 2022), and ScanObjectNN-C (Mirza et al. 2023), as well as clean datasets (large-scale and small-scale)—achieving new state-of-the-art results.

Related Work

3D Vision-Language Foundation Models (3D VLFMs) have demonstrated transformative potential in advancing point cloud understanding by bridging semantic representations from large-scale image-text datasets to 3D data (Zhu et al. 2023; Chen et al. 2023; Xue et al. 2024). For instance, Uni3D (Zhou et al. 2024), ULIP (Xue et al. 2023), ULIP-2 (Xue et al. 2024), and OpenShape (Liu et al. 2023) employ contrastive learning on extensive datasets of paired image, text, and point cloud data to achieve robust cross-modal feature alignment. These pre-trained 3D VLFMs exhibit strong zero-shot capabilities and geometric semantic perception across diverse tasks. However, their performance is often hindered by domain gaps, limiting generalization to real-world and dynamic scenarios.

Test-Time Adaptation (TTA) focuses on dynamically adapting model predictions to novel domains without requiring target annotations or access to the source data (Niu et al. 2023; Boudiaf et al. 2022). Early TTA methods, designed for vision-only models, adapted parameters via post-hoc regularization during inference. For instance, TENT (Wang et al. 2020), SHOT (Liang, Hu, and Feng 2020), and MEMO (Zhang, Levine, and Finn 2022) minimize the entropy of the softmax prediction distribution to boost confidence and generalization to the downstream domains. With advancements in VLFMs, recent TTA approaches leverage text modalities to enhance generalization. TPT (Shu et al. 2022) and DiffTPT (Feng et al. 2023) combine entropy minimization with fine-tuning a learnable prompt for each test sample. SCAP (Zhang et al. 2025) optimizes both image and text prompts for TTA. While effective, these methods require costly gradient backpropagation at the test time. In contrast, TDA (Karmanov et al. 2024), COSMIC (Huang et al.

2025), and PointCache (Sun et al. 2025) use cached high-confidence prototypes to refine VLFM predictions through similarity-based scoring. However, relying solely on confident samples can miss distribution modes, and noisy prototypes can lead to suboptimal performance.

Test-Time Point Cloud Adaptation has gained significant traction in improving the generalization of 3D point cloud analysis across tasks, including recognition (Sun et al. 2025; Wang et al. 2024; Shim, Kim, and Yang 2024), segmentation (Zhao et al. 2025; Zou et al. 2024), registration (Hatem, Qian, and Wang 2023), object detection (Lin et al. 2024; Chen et al. 2024; Yuan et al. 2024), and scene completion (Jang et al. 2025). These approaches can be divided into two distinct groups. The first group modifies model parameters and employs training during inference. For instance, MATE (Mirza et al. 2023) adapts encoder parameters through self-training, and Bahri et al. (Bahri et al. 2025) adapt normalization layers using TENT (Wang et al. 2021). The second group employs methods that avoid parameter updates. Specifically, BFTT3D (Wang et al. 2024) integrates source representations using a non-parametric adapter, whereas CloudFixer (Shim, Kim, and Yang 2024) and 3DD-TTA (Dastmalchi et al. 2025) adapt input point clouds through geometric transformations guided by diffusion models. However, these methods are often designed for smaller-scale models and face challenges when applied to large, multi-modal 3D models. PointCache (Sun et al. 2025), closely related to our work, adapts VLFMs using global and local caches built from high-confidence predictions and applies k-means to summarize local patch features. In contrast, our Uni-Adapter performs online, confidence-weighted clustering at the class level to capture diverse distribution modes in 3D data.

Proposed Method

Background

3D VLFMs (Xue et al. 2024; Liu et al. 2023; Zhou et al. 2024) use separate encoders to map point clouds, images, and text into a shared, aligned feature space. A text encoder $\mathrm{E_T}$, typically based on CLIP (Radford et al. 2021b), encodes class prompts, while a transformer-based point encoder $\mathrm{E_P}$, adapted with a point tokenizer, encodes 3D point clouds. In zero-shot classification, a generic prompt $r =$ "a point cloud of a" is prepended to the ith class name $y_i \in \mathcal{Y}$, where $\mathcal{Y} = \{y_1, \ldots, y_K\}$ denotes the set of K class names. The resulting textual inputs $\{r, y_i\}$ are encoded as $\mathbf{w}_i = \mathrm{E_T}(\{r, y_i\}) \in \mathbb{R}^d$, and d is the embedding dimension. Given a point cloud $\mathbf{X} \in \mathbb{R}^{L\times 3}$, its embedding $\mathbf{f} = \mathrm{E_P}(\mathbf{X}) \in \mathbb{R}^d$ is compared to $\mathbf{w}_i$ via cosine similarity, and the probability distribution is given as follows:

$$p(y_i|\mathbf{X}) = \frac{\exp(\mathrm{sim}(\mathbf{w}_i, \mathbf{f})/\tau)}{\sum_{j=1}^{K} \exp(\mathrm{sim}(\mathbf{w}_j, \mathbf{f})/\tau)} \tag{1}$$

where $\mathrm{sim}(\cdot,\cdot)$ denotes the cosine similarity and τ is the temperature controlling the sharpness of the distribution.

Uni-Adapter Method

The overall framework of the proposed method is illustrated in Fig. 2. It integrates similarity scores from point cloud-to-text comparisons with those from the cache model. The cache model learns 3D prototypes using an *online prototyping* module and dynamically refines noisy prototype assignment through *prototype reassignment* module. It then computes cache scores based on the affinity between the input point cloud representation and the stored prototypes. Finally, these scores are unified based on the entropy of the predictions to obtain the final similarity score. The following sections provide a detailed description of each component.

Online Prototyping Module. We adopt an *online clustering* strategy, termed *online prototyping*, to dynamically capture diverse modes of the data distribution. This module incrementally updates a set of class-specific prototypes. The goal is to associate each incoming point cloud feature with a representative prototype and update it accordingly.

At time step t, a point cloud $\mathbf{X}_t$ is encoded as $\mathbf{f}_t = \mathrm{E_P}(\mathbf{X}_t) \in \mathbb{R}^d$. We first predict the class k by computing cosine similarities between $\mathbf{f}_t$ and class embeddings $\{\mathbf{w}_i\}_{i=1}^K$:

$$s_i^{\text{main}} = \mathrm{sim}(\mathbf{w}_i, \mathbf{f}_t), \quad k = \arg\max_i s_i^{\text{main}}. \tag{2}$$

Each class k maintains up to N prototypes, denoted by $\{\mathbf{c}_{k,j} \in \mathbb{R}^d\}_{j=1}^{N_k}$, where $N_k \le N$. Given the predicted class k, we select the most similar prototype:

$$n = \arg\max_{1\le j\le N_k} \mathrm{sim}(\mathbf{f}_t, \mathbf{c}_{k,j}). \tag{3}$$

If an empty slot exists ($N_k < N$), it is initialized with $\mathbf{f}_t$. Otherwise, the selected prototype $\mathbf{c}_{k,n}$ is updated using a confidence-weighted moving average:

$$\mathbf{c}_{k,n}^{\text{new}} = \frac{\alpha_t \mathbf{f}_t + b_{k,n}\alpha_{k,n}\mathbf{c}_{k,n}^{\text{old}}}{\alpha_t + b_{k,n}\alpha_{k,n}}, \tag{4}$$

where $b_{k,n}$ is the number of past updates to the prototype, and α_t, $\alpha_{k,n}$ are the confidence scores of the incoming sample and the cached prototype, respectively. These scores are derived from prediction entropy as:

$$\alpha_t = \exp(-\beta \cdot H_t), \quad \alpha_{k,n} = \exp(-\beta \cdot H_{k,n}), \tag{5}$$

where β is a scaling factor, and H_t and $H_{k,n}$ denote the entropy of the softmax over similarities with text embeddings. Specifically, H_t is computed from feature $\mathbf{f}_t$, and $H_{k,n}$ from prototype $\mathbf{c}_{k,n}$, both compared against $\{\mathbf{w}_i\}_{i=1}^K$.

Prototype Reassignment Module. While online prototyping maintains representative prototypes, it remains sensitive to noisy pseudo-labels. To improve label reliability, we introduce a *prototype reassignment* module that smooths pseudo-labels across similar prototypes via graph-based regularization. To refine pseudo-labels based on semantic relationships, we require two components: (1) a similarity matrix capturing prototype relationships, and (2) the initial soft pseudo-labels to be updated. These soft pseudo-labels, given by the model's softmax probabilities over class logits, are stored in $\mathbf{Z}^{(0)} \in \mathbb{R}^{M\times K}$, where each row corresponds to a prototype and contains its class probabilities.

Let $M = \sum_{k=1}^{K} N_k$ denote the total number of active prototypes across all classes, where $N_k \le N$ is the number

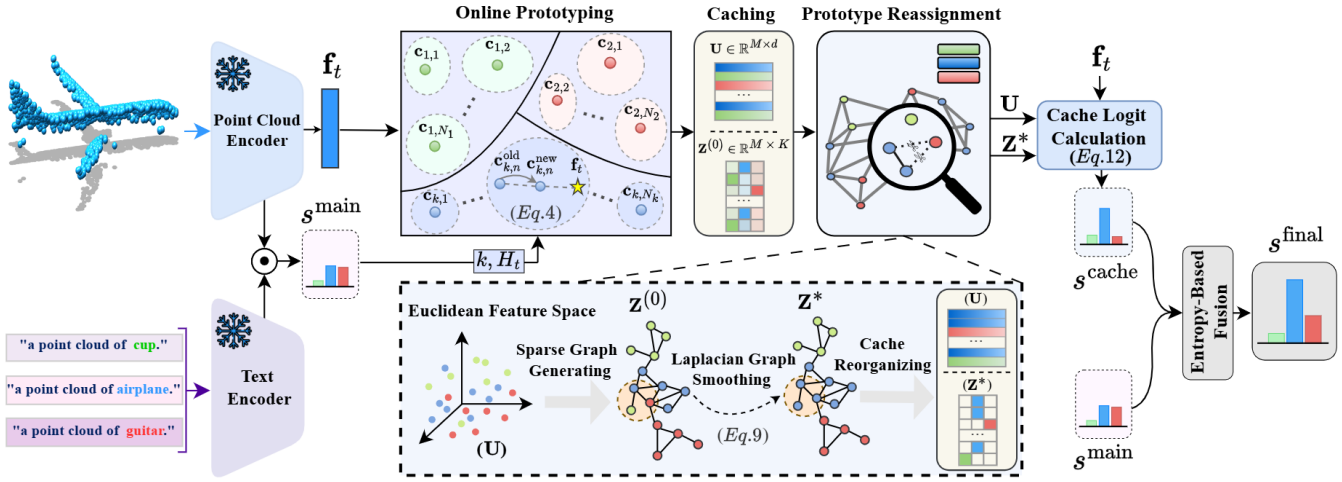

Figure 2: Method Overview. Given a test point cloud $\mathbf{X}_t \in \mathbb{R}^{L\times 3}$, our method extracts a point cloud feature $\mathbf{f}_t$ via a point cloud encoder. The 3D cache is updated via online Prototyping, where cluster centers serve as 3D prototypes. The Prototype Reassignment module refines these prototypes, and their affinity with $\mathbf{f}_t$ is computed to obtain $\mathbf{s}^{\text{cache}}$. Finally, the prediction logit $\mathbf{s}^{\text{final}}$ is obtained by fusing $\mathbf{s}^{\text{cache}}$ and the model's base output $\mathbf{s}^{\text{main}}$ using entropy-driven confidence weighting.

of prototypes for class k. We collect all prototype features into a matrix $\mathbf{U} = [\mathbf{c}_{1,1}; \ldots; \mathbf{c}_{K,N_K}] \in \mathbb{R}^{M\times d}$, where each row is a ℓ_2-normalized prototype. The similarity matrix is computed as:

$$\mathbf{A} = \mathbf{U}\mathbf{U}^\top \in \mathbb{R}^{M\times M}. \tag{6}$$

We apply a threshold $\gamma \in [0, 1]$ to remove weak connections and obtain a sparse matrix $\hat{\mathbf{A}}$ by zeroing out values below γ. From $\hat{\mathbf{A}}$, we compute the degree matrix $\mathbf{D}$, a diagonal matrix where each diagonal entry $\mathbf{D}_{mm}$ is the sum of the m-th row in $\hat{\mathbf{A}}$. The normalized graph Laplacian is then:

$$\mathbf{L}_{\text{norm}} = \mathbf{I} - \mathbf{D}^{-1/2}\hat{\mathbf{A}}\mathbf{D}^{-1/2}. \tag{7}$$

This refinement is formulated as the following optimization:

$$\mathbf{Z}^* = \arg\min_{\mathbf{Z}} \|\mathbf{Z} - \mathbf{Z}^{(0)}\|_F^2 + \lambda_{\text{reg}} \cdot \text{Tr}(\mathbf{Z}^\top \mathbf{L}_{\text{norm}}\mathbf{Z}), \tag{8}$$

where $\lambda_{\text{reg}} > 0$ balances fidelity to the initial predictions and label smoothness across the graph. This objective has a closed-form solution:

$$\mathbf{Z}^* = (\mathbf{I} + \lambda_{\text{reg}}\mathbf{L}_{\text{norm}})^{-1}\,\mathbf{Z}^{(0)}. \tag{9}$$

Finally, we convert $\mathbf{Z}^*$ into a one-hot label matrix by keeping the maximum entry in each row:

$$\mathbf{Z}^*_{m,\hat{i}} = \begin{cases} 1 & \text{if } \hat{i} = \arg\max_i \mathbf{Z}^*_{m,i}, \\ 0 & \text{otherwise} \end{cases} \quad \text{for } m = 1, \ldots, M. \tag{10}$$

It should be noted that, to reduce computational overhead ($\mathcal{O}(M^3)$), we solve Equation 9 using the conjugate gradient method (Hestenes, Stiefel et al. 1952), which reduces the complexity to $\mathcal{O}(\rho \cdot \text{nnz}(\mathbf{L}_{\text{norm}}))$, where ρ is the number of iterations and $\text{nnz}(\cdot)$ denotes the number of non-zero entries.

Cache Logit Calculation. Each prototype now has a one-hot class label encoded in $\mathbf{Z}^* \in \{0, 1\}^{M\times K}$. Given an input feature $\mathbf{f}_t \in \mathbb{R}^d$, we compute its cosine similarity to all prototypes as: $\mathbf{U}\mathbf{f}_t \in \mathbb{R}^M$. To ensure that the similarity scores are not biased by the number of prototypes per class, we normalize by the count of prototypes assigned to each class. Specifically, we compute a diagonal normalization matrix:

$$\mathbf{\Lambda} = \text{diag}\left(\left(\frac{1}{\sum_{m=1}^{M}\mathbf{Z}^*_{m,i}}\right)_{i=1}^{K}\right) \in \mathbb{R}^{K\times K}, \tag{11}$$

where the i-th diagonal entry rescales the summed similarity for class i by the number of associated prototypes. The cache-based logits are then computed as:

$$\mathbf{s}^{\text{cache}} = \mathbf{\Lambda}\,\mathbf{Z}^{*\top}\,(\mathbf{U}\mathbf{f}_t) \in \mathbb{R}^K. \tag{12}$$

This yields class-wise average similarities between $\mathbf{f}_t$ and the prototypes. The resulting $\mathbf{s}^{\text{cache}}$ is fused with main logits for robust classification.

Entropy-Based Fusion. We combine the predictions from the source VLFM and the cache model by fusing their logits. The fusion is performed using entropy-based weighting:

$$\boldsymbol{s}^{\text{final}} = \frac{H_{\text{cache}} \cdot \boldsymbol{s}^{\text{main}} + H_t \cdot \boldsymbol{s}^{\text{cache}}}{H_{\text{cache}} + H_t}. \tag{13}$$

Here, H_t and H_{cache} are the entropies of the softmax over the main and cache logits, respectively. The fusion adaptively weights each source by its confidence, favoring the more certain modality.

Experiments

Experimental Setup

Datasets. We evaluate our approach under distribution shifts using ModelNet-40C (Sun et al. 2022), ShapeNet-C (Mirza et al. 2023), and ScanObjectNN-C (Mirza et al. 2023), which introduce 15 types of synthetic corruptions, including density variations, noise, and geometric transformations, each with five severity levels. To further assess the generalization capability on unseen data, we conduct

	Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	Avg.
	Source-Only	57.37	54.01	70.21	61.91	60.69	51.74	52.39	67.50	74.87	72.40	71.02	63.97	58.95	47.24	22.93	59.15
Training	TENT (ICLR21)	61.46	58.30	65.28	56.36	65.08	51.62	52.51	65.44	75.24	72.36	72.44	61.43	58.18	48.09	28.44	59.48
Training	SHOT (ICML20)	61.50	58.42	65.27	56.68	64.66	51.54	52.84	65.48	75.57	72.81	72.29	61.14	58.14	48.46	28.44	59.55
Training	SAR (ICLR23)	61.51	58.18	65.40	56.77	64.79	51.86	53.40	66.37	75.93	72.49	72.08	62.28	59.12	49.31	30.71	60.01
Training	DUA (CVPR22)	61.26	57.90	65.11	56.76	64.79	51.38	53.08	65.44	75.16	72.93	72.41	60.85	58.35	48.50	28.65	59.50
Training	MEMO (NIPS22)	57.38	54.08	70.24	61.92	60.71	51.76	52.39	67.59	74.88	72.48	71.03	63.98	59.30	47.36	22.94	59.20
Training	TPT* (NIPS22)	60.65	57.20	76.32	61.15	63.47	55.39	55.78	71.36	74.77	75.20	73.01	65.41	60.97	47.60	18.10	61.02
Training-Free	LAME (CVPR22)	57.33	54.09	70.10	61.63	60.94	51.94	52.27	67.50	75.16	72.45	71.27	63.90	59.04	47.33	22.61	59.17
Training-Free	3DD-TTA (WACV25)	60.53	59.76	63.53	64.71	67.91	50.73	51.09	59.36	66.98	64.26	58.67	59.36	53.77	36.10	32.37	56.06
Training-Free	CloudFixer † (ECCV24)	65.44	65.84	63.90	68.11	**72.77**	52.80	53.81	52.88	63.21	59.36	61.18	56.16	52.18	29.09	24.60	56.09
Training-Free	T3A (NIPS21)	**69.40**	**70.26**	41.89	63.33	70.74	61.26	59.27	72.20	79.33	77.59	78.36	71.07	65.51	49.59	32.01	64.12
Training-Free	TDA* (CVPR24)	62.20	61.63	75.16	65.52	67.18	57.46	57.94	70.95	76.94	74.43	73.26	67.46	63.17	50.69	30.39	63.63
Training-Free	Point-Cache* (CVPR25)	64.34	64.87	73.95	68.31	71.68	62.84	65.19	73.22	77.80	77.15	75.77	69.77	68.31	54.78	32.98	66.73
	Uni-Adapter (Ours)	66.82	65.52	**78.32**	**72.25**	72.04	**65.60**	**66.61**	**77.51**	**80.63**	**79.05**	**79.30**	**75.29**	**73.38**	**56.92**	**36.26**	**69.70**

Table 1: Top-1 accuracy (%) on ModelNet40-C under distribution shifts using Uni3D-Large (batch size = 1). Source-Only shows performance without adaptation. Best and second-best are in **bold** and underline. * denotes VLFM-based TTA methods.

	Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	Avg.
	Source-Only	60.33	55.75	65.95	65.02	59.04	59.41	60.23	79.06	71.07	75.62	73.87	76.82	63.22	2.37	1.06	57.92
Training	TENT (ICLR21)	59.88	54.30	58.31	61.14	57.78	58.69	60.17	79.30	72.69	75.83	74.54	77.08	63.23	2.97	**2.54**	57.23
Training	SHOT (ICML20)	59.96	54.32	58.37	61.14	57.67	58.81	60.19	79.23	72.53	75.90	74.57	77.09	63.34	3.02	2.51	57.24
Training	SAR (ICLR23)	59.86	54.09	58.59	60.84	57.48	58.65	60.09	79.04	73.23	74.78	71.39	77.04	61.88	2.69	1.40	56.74
Training	DUA (CVPR22)	59.85	54.37	58.26	61.21	57.89	58.71	60.15	79.11	72.40	72.55	75.91	77.05	63.24	2.97	2.53	57.08
Training	MEMO (NIPS22)	60.33	55.76	66.02	65.02	59.04	59.42	60.23	79.01	70.92	75.62	73.95	76.82	63.22	2.41	1.09	57.92
Training	TPT* (NIPS22)	62.87	56.63	69.20	64.70	59.10	58.29	60.43	**81.59**	75.23	76.93	74.56	**80.52**	63.02	2.17	1.25	59.10
Training-Free	LAME (CVPR22)	60.43	55.89	66.04	65.12	59.09	59.42	60.33	79.13	71.16	75.74	74.08	76.99	63.35	2.31	1.02	58.01
Training-Free	3DD-TTA (WACV25)	65.78	64.16	55.00	61.75	**68.05**	55.06	56.20	74.20	67.35	68.06	61.19	73.01	56.29	2.50	0.97	55.30
Training-Free	CloudFixer † (ECCV24)	65.57	**65.30**	58.15	**69.53**	63.65	55.02	56.71	69.89	58.67	65.65	65.64	70.36	55.09	3.75	2.46	57.24
Training-Free	T3A (NIPS21)	60.60	53.12	20.70	44.19	49.31	46.35	44.37	70.31	63.01	64.86	63.83	68.24	52.60	1.00	0.87	46.89
Training-Free	TDA* (CVPR24)	62.75	58.95	68.33	67.14	62.09	61.28	62.56	79.00	71.44	75.93	74.79	77.17	64.44	3.82	1.81	59.43
Training-Free	Point-Cache* (CVPR25)	62.63	56.71	66.51	65.85	61.15	59.79	61.49	75.89	69.47	72.61	70.81	73.82	63.41	3.64	1.67	57.70
	Uni-Adapter (Ours)	**66.89**	62.23	**71.38**	68.62	64.15	**67.42**	**67.33**	80.76	**75.69**	**78.11**	**77.01**	79.64	**70.14**	**4.36**	2.47	**62.41**

Table 2: Top-1 accuracy (%) on ShapeNet-C under distribution shifts using Uni3D-Large (batch size = 1). Source-Only shows performance without adaptation. Best and second-best are in **bold** and underline. * denotes VLFM-based TTA methods.

experiments on the test splits of ModelNet40 (Wu et al. 2015), ShapeNetCore-v2 (Chang et al. 2015), and ScanObjectNN (Uy et al. 2019), as well as large-scale 3D datasets such as OmniObject3D (Wu et al. 2023) (216 classes) and Objaverse-LVIS (Deitke et al. 2023) (1,156 classes), designed to evaluate generalization across diverse categories.

Baselines. To evaluate our Uni-Adapter and ensure a fair comparison, we implemented twelve diverse baselines spanning both training-free and training-based TTA approaches. Specifically, we evaluate TENT (Wang et al. 2020), SHOT (Liang, Hu, and Feng 2020), SAR (Niu et al. 2023), DUA (Mirza et al. 2022), MEMO (Zhang, Levine, and Finn 2022), LAME (Boudiaf et al. 2022), T3A (Iwasawa and Matsuo 2021), CloudFixer (Shim, Kim, and Yang 2024), 3DD-TTA (Dastmalchi et al. 2025), TPT (Shu et al. 2022), TDA (Karmanov et al. 2024), and Point-Cache (Sun et al. 2025). While CloudFixer, 3DD-TTA, and Point-Cache are designed for 3D point clouds, the remaining methods originate from the 2D domain and are adapted for 3D data. For CloudFixer, we use only its generative model and guidance, without updating the source model, denoted as CloudFixer †. Note that TPT and TDA are developed specifically for 2D VLFMs, whereas Point-Cache is natively built for 3D VLFMs.

Implementation. We use ULIP-2 (Xue et al. 2024), OpenShape (Liu et al. 2023), and Uni3D-Large (Zhou et al. 2024) as 3D VLFMs. Test-time adaptation is performed on a single sample. For Graph-Based Label Smoothing, we set the sparsity threshold $\gamma = 0.5$ to retain strong correlations in the adjacency matrix and the confidence decay parameter $\beta = 10$ to balance diversity and confidence when updating cluster centers. Each target sample has 1024 points, except Objaverse-LVIS with 10,000. All experiments use corruption severity level 5 on a single NVIDIA RTX 4090 GPU.

	Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	Avg.
	Source-Only	29.78	25.99	40.62	45.96	30.64	33.05	34.42	56.28	47.16	54.22	54.04	56.80	43.55	9.98	8.61	38.07
Training	TENT (ICLR21)	29.78	26.16	49.91	51.46	30.65	33.22	36.32	55.08	45.27	52.50	53.18	55.77	44.92	7.06	3.79	38.34
Training	SHOT (ICML20)	29.60	26.85	51.12	52.32	31.33	33.73	37.20	56.80	45.78	54.57	54.22	56.31	45.27	6.72	3.96	39.05
Training	SAR (ICLR23)	29.08	27.54	42.17	44.58	31.33	32.36	34.77	56.28	44.41	52.84	54.22	55.59	44.06	9.64	9.64	37.90
Training	DUA (CVPR22)	29.95	27.37	41.65	44.92	30.81	32.53	34.08	56.63	45.09	53.87	54.22	55.77	43.89	9.81	9.47	38.00
Training	MEMO (NIPS22)	29.77	26.16	49.91	51.46	30.64	33.22	36.32	55.08	45.27	52.50	53.18	55.77	44.92	7.05	3.78	38.34
Training	TPT* (NIPS22)	30.04	28.43	40.95	46.76	32.68	35.55	34.39	56.60	50.66	53.81	54.39	60.30	42.70	11.13	6.07	38.96
Training-Free	LAME (CVPR22)	29.60	26.85	51.12	52.32	31.33	33.73	37.18	56.80	45.78	54.56	54.22	56.29	45.27	6.71	3.96	39.05
Training-Free	3DD-TTA (WACV25)	32.19	30.81	27.71	39.59	34.60	25.82	26.51	45.61	38.04	36.14	33.39	45.09	33.05	8.61	4.47	30.78
Training-Free	CloudFixer † (ECCV24)	**36.83**	33.22	36.14	48.19	37.69	27.54	30.46	45.44	40.28	38.73	38.21	46.13	35.28	10.67	**11.70**	34.43
Training-Free	T3A (NIPS21)	34.94	32.70	34.08	43.37	33.22	36.49	36.32	**62.65**	**55.42**	61.46	**62.31**	**64.03**	**56.97**	9.47	8.61	42.14
Training-Free	TDA* (CVPR24)	31.33	28.40	52.67	53.36	32.36	36.32	40.45	56.80	46.82	55.25	55.76	57.66	49.40	8.09	4.65	40.62
Training-Free	Point-Cache* (CVPR25)	30.98	27.19	**55.25**	56.45	33.73	40.79	43.03	59.50	49.23	60.07	56.63	57.66	49.05	8.26	4.30	42.13
	Uni-Adapter (Ours)	35.28	**37.69**	53.35	**59.55**	**39.07**	**42.16**	**52.49**	60.58	51.97	**61.61**	60.24	60.92	54.38	**20.82**	4.81	**46.33**

Table 3: Top-1 accuracy (%) on ScanObjectNN-C using Uni3D-Large (batch size = 1). * denotes VLFM-based TTA. Source-Only shows performance without adaptation. Best and second-best are in **bold** and underline.

Method	Small-Scale Data			Large-Scale Data	
	ModelNet	SONN	ShapeNet	O-LVIS	Omni3D
ULIP-2	71.23	52.49	69.53	30.26	26.36
+Point-Cache	74.53	58.52	69.74	**32.36**	29.38
+Uni-Adapter	**76.47**	**58.84**	**70.81**	31.83	**30.37**
O-Shape	84.56	55.94	74.53	46.15	34.09
+Point-Cache	84.04	62.48	78.51	46.05	34.45
+Uni-Adapter	**85.44**	**63.12**	**79.99**	**48.07**	**37.21**
Uni3D	78.84	59.55	79.90	46.20	32.06
+Point-Cache	83.43	62.27	78.51	47.13	33.22
+Uni-Adapter	**83.96**	**64.03**	**81.23**	**47.49**	**35.95**

Table 4: Performance of Uni-Adapter on 3D VLFMs across clean datasets (batch size = 1). Objaverse-LVIS uses 10,000 points; others use 1,024. SONN denotes ScanObjectNN.

Results

Robustness Against Distribution Shifts. We evaluate Uni-Adapter on ModelNet-40C (Table 1) and ShapeNet-C (Table 2) across diverse corruption types. While most training-based baselines adapted from the 2D domain show only marginal gains, methods such as T3A, TDA, and Point-Cache yield relatively stronger improvements. We also observe that, although 3D input adaptation methods (Shim, Kim, and Yang 2024; Dastmalchi et al. 2025) address specific types of corruption, their deployment on novel test instances may introduce excessive generation errors and greater deviations from the source domain, ultimately amplifying distribution shift. Moreover, methods designed for VLFMs demonstrate stronger capabilities in reducing modality gaps. While test-time methods like T3A, TDA, and Point-Cache outperform traditional 2D baselines on ModelNet-40C, Uni-Adapter surpasses all alternatives, particularly under significant domain shifts, with improvements reaching 10.55%. On the more diverse ShapeNet-C benchmark, characterized by higher intra-class variance, methods relying solely on confident samples (e.g., T3A, Point-Cache) lead to suboptimal generalization or performance degradation. In contrast, Uni-Adapter, by integrating both confidence and diversity, effectively mitigates these challenges and enhances the source model performance by 4.49%.

Effectiveness Under Severe Distribution Shifts. To further evaluate the robustness of our Uni-Adapter, we conduct experiments on ScanObjectNN-C, a challenging variant of ScanObjectNN. ScanObjectNN consists of real-world 3D scans that often contain background clutter and partial observations. ScanObjectNN-C introduces additional corruptions to simulate more severe real-world disturbances. As shown in Table 3, our method outperforms the source model by 8.26%, showing strong generalization to significant distribution shifts and robustness to real-world 3D corruptions.

Generalization. Uni-Adapter demonstrates superior generalization on unseen, uncorrupted datasets of varying scales (see Table 4). On three small-scale datasets—ModelNet, ScanObjectNN, and ShapeNet—Uni-Adapter outperforms Point-Cache, improving performance across all three source 3D VLFMs and establishing new state-of-the-art baselines. On large-scale 3D benchmarks, including OmniObject3D and Objaverse-LVIS, which feature a diverse and realistic spectrum of object categories, Uni-Adapter consistently yields absolute performance gains.

Inference Efficiency. We evaluate the inference throughput of Uni-Adapter on the ModelNet40 dataset, where throughput (t/s) refers to the number of test instances processed per second. Table 5 shows that Uni-Adapter incurs a smaller drop in throughput compared to zero-shot inference. This overhead is primarily due to the additional operations involved in online prototyping, prototype Reassignment, and logit computation. These findings highlight that Uni-Adapter is a more efficient approach, achieving substantial accuracy improvements with minimal computational overhead relative to other cache-based baseline methods.

Method	Uni3D	OpenShape	ULIP-2
Zero-shot	**39.19**	**15.90**	**23.94**
TDA	36.02	14.43	21.78
Point-Cache	9.73	9.74	11.11
Uni-Adapter (Ours)	36.93	15.06	22.67

Table 5: Throughput (t/s) comparison of 3D VLFMs and cache baselines on ModelNet40-C (batch size = 1). Results are averaged over test samples on an RTX 4090 GPU.

Ablation Study

Effectiveness of Different Components of Uni-Adapter: Table 6 evaluates the effectiveness of Uni-Adapter's components. Starting with Online Prototyping as the only component (row #1), we observe a significant improvement over the source model (row #0) across all benchmarks. In row #2, adding Prototype Reassignment further reduces the performance gap, improving ModelNet-40C by 1.22%, with similar trends across other datasets. By leveraging correlated features in the dynamic adjacency graph, this module refines pseudo-labels, mitigates inconsistencies, and dynamically corrects errors. The Prototype Reassignment module introduces negligible overhead by employing a conjugate gradient solver that converges quickly on sparse systems.

#	Online Prototyping	Prototype Reassignment	ModelNet-40C	SONN-C	ShapeNet-C
0	✗	✗	59.15	38.07	57.92
1	✓	✗	68.48	45.12	61.79
2	✓	✓	**69.70**	**46.33**	**62.41**

Table 6: Ablation of Uni-Adapter components. Metric: top-1 accuracy; #0 is the source model without adaptation.

Cluster-Based Cache vs. Confidence-Based Cache: We evaluated the effectiveness of our online clustering strategy for learning 3D prototypes by comparing it to a confidence-based cache that retains only the most confident prototypes. Replacing the cluster-based cache in Uni-Adapter with the confidence-based method, we observed that the former consistently outperforms the latter across corruption types in the ShapeNet-C dataset (Fig. 3). This improvement demonstrates that online clustering generates more diverse prototypes and better captures the underlying data distribution modes, resulting in a more robust decision boundary.

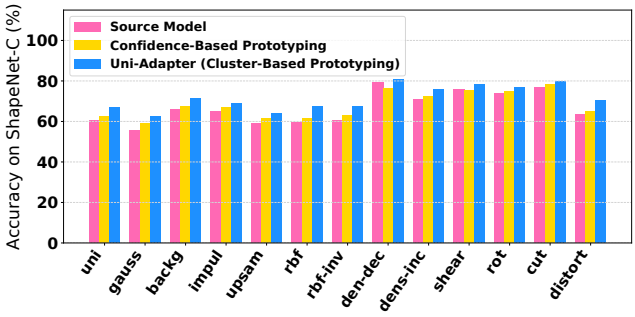

Figure 3: Cluster- vs. confidence-based caches in Uni-Adapter on ShapeNet-C. Cluster-based caching gives higher accuracy by capturing diverse modes, while confidence-based caching misses much of the class distribution.

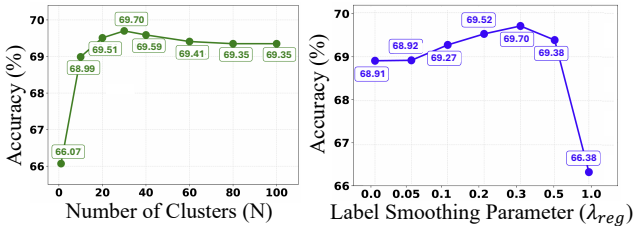

Figure 4: Ablation on the number of cluster centers (N) and label smoothing (λ_{reg}) for ModelNet-40C.

Number of Cluster Centers: Fig. 4 shows the effect of selecting the right number of cluster centers. Too few clusters fail to represent class distributions and limit the effectiveness of Prototype Reassignment module, which requires sufficient features. Too many clusters introduce noise, weakening confidence-based clustering. Our findings suggest that cache size of 30 balances diversity and confidence, with the trade-off controlled by confidence decay hyperparameter β.

Graph-based Label Smoothing: We illustrate the impact of the graph-based label smoothing λ_{reg} on the performance of the Uni-Adapter (Fig. 4). Varying λ_{reg} from 0 to 1 controls the refinement intensity. When λ_{reg} is close to 0, the smoothing effect becomes negligible, resulting in minimal pseudo-label refinement and degraded performance. This emphasizes the importance of our Prototype Reassignment component in enhancing adaptation. Conversely, increasing λ_{reg} enhances refinement but can overly smooth labels, diminishing their original influence. We find that $\lambda_{reg} = 0.3$ balances effective smoothing with label integrity.

Conclusion

In this paper, we introduce Uni-Adapter, a novel training-free test-time adaptation framework tailored for 3D Vision-Language Foundation Models (VLFMs). Unlike prior confidence-based approaches, Uni-Adapter leverages cluster-based prototypes to capture the multiple mode distribution present in 3D data, enabling more accurate and robust adaptation to real-world variation. By incorporating online prototyping, graph-based prototype reassignment, and entropy-weighted fusion, our method effectively mitigates the challenges of noisy pseudo-labels and preserves semantic consistency across diverse target domains. Extensive experiments demonstrate that Uni-Adapter sets a new state of the art in test-time performance for 3D VLFMs, offering an efficient solution for dynamic and resource-constrained environments. However, Uni-Adapter faces performance instability during the transient cache initialization phase. Future work may incorporate lightweight self-supervised training using contrastive losses or prototype consistency objectives to improve prototype stability and early-stage adaptation.

References

Bahri, A.; Yazdanpanah, M.; Noori, M.; Oghani, S. D.; Cheraghalikhani, M.; Osowiechi, D.; Beizaee, F.; Hakim, G. A. V.; Ben Ayed, I.; and Desrosiers, C. 2025. Test-Time Adaptation in Point Clouds: Leveraging Sampling Variation with Weight Averaging. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, 266–275.

Boudiaf, M.; Mueller, R.; Ben Ayed, I.; and Bertinetto, L. 2022. Parameter-free online test-time adaptation. In *CVPR*.

Chang, A. X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*.

Chen, R.; Liu, Y.; Kong, L.; Zhu, X.; Ma, Y.; Li, Y.; Hou, Y.; Qiao, Y.; and Wang, W. 2023. Clip2scene: Towards label-efficient 3d scene understanding by clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7020–7030.

Chen, Z.; Meng, J.; Baktashmotlagh, M.; Zhang, Y.; Huang, Z.; and Luo, Y. 2024. Mos: Model synergy for test-time adaptation on lidar-based 3d object detection. *arXiv preprint arXiv:2406.14878*.

Dastmalchi, H.; An, A.; Cheraghian, A.; Rahman, S.; and Ramasinghe, S. 2025. Test-Time Adaptation of 3D Point Clouds via Denoising Diffusion Models. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, 1566–1576.

Deitke, M.; Schwenk, D.; Salvador, J.; Weihs, L.; Michel, O.; VanderBilt, E.; Schmidt, L.; Ehsani, K.; Kembhavi, A.; and Farhadi, A. 2023. Objaverse: A Universe of Annotated 3D Objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13142–13153.

Feng, C.-M.; Yu, K.; Liu, Y.; Khan, S.; and Zuo, W. 2023. Diverse data augmentation with diffusions for effective test-time prompt tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2704–2714.

Hatem, A.; Qian, Y.; and Wang, Y. 2023. Point-tta: Test-time adaptation for point cloud registration using multitask meta-auxiliary learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16494–16504.

Hestenes, M. R.; Stiefel, E.; et al. 1952. Methods of conjugate gradients for solving linear systems. *Journal of research of the National Bureau of Standards*, 49(6): 409–436.

Huang, F.; Jiang, J.; Jiang, Q.; Li, H.; Khan, F. N.; and Wang, Z. 2025. COSMIC: Clique-Oriented Semantic Multi-space Integration for Robust CLIP Test-Time Adaptation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 9772–9781.

Iwasawa, Y.; and Matsuo, Y. 2021. Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34: 2427–2440.

Jang, H.-K.; Kim, J.; Kweon, H.; and Yoon, K.-J. 2025. TALoS: Enhancing semantic scene completion via test-time adaptation on the line of sight. *Advances in Neural Information Processing Systems*, 37: 74211–74232.

Karmanov, A.; Guan, D.; Lu, S.; El Saddik, A.; and Xing, E. 2024. Efficient Test-Time Adaptation of Vision-Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14162–14171.

Liang, J.; Hu, D.; and Feng, J. 2020. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *ICML*.

Lin, H.; Zhang, Y.; Niu, S.; Cui, S.; and Li, Z. 2024. Monotta: Fully test-time adaptation for monocular 3d object detection. In *European Conference on Computer Vision*, 96–114. Springer.

Liu, M.; Shi, R.; Kuang, K.; Zhu, Y.; Li, X.; Han, S.; Cai, H.; Porikli, F.; and Su, H. 2023. Openshape: Scaling up 3d shape representation towards open-world understanding. *Advances in neural information processing systems*, 36: 44860–44879.

Mirza, M. J.; Micorek, J.; Possegger, H.; and Bischof, H. 2022. The norm must go on: Dynamic unsupervised domain adaptation by normalization. In *CVPR*.

Mirza, M. J.; Shin, I.; Lin, W.; Schriebl, A.; Sun, K.; Choe, J.; Possegger, H.; Kozinski, M.; Kweon, I. S.; Yoon, K.-J.; et al. 2023. MATE: Masked Autoencoders are Online 3D Test-Time Learners. In *ICCV*.

Niu, S.; Wu, J.; Zhang, Y.; Wen, Z.; Chen, Y.; Zhao, P.; and Tan, M. 2023. Towards stable test-time adaptation in dynamic wild world. In *ICLR*.

Osowiechi, D.; Noori, M.; Vargas Hakim, G.; Yazdanpanah, M.; Bahri, A.; Cheraghalikhani, M.; Dastani, S.; Beizaee, F.; Ayed, I.; and Desrosiers, C. 2024. WATT: Weight average test time adaptation of CLIP. *Advances in neural information processing systems*, 37: 48015–48044.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021a. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 8748–8763. PMLR.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021b. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, 8748–8763. PMLR.

Sharifdeen, A.; Munir, M. A.; Baliah, S.; Khan, S.; and Khan, M. H. 2025. O-TPT: Orthogonality Constraints for Calibrating Test-time Prompt Tuning in Vision-Language Models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 19942–19951.

Shim, H.; Kim, C.; and Yang, E. 2024. CloudFixer: Test-Time Adaptation for 3D Point Clouds via Diffusion-Guided Geometric Transformation. In *European Conference on Computer Vision*, 454–471. Springer.

Shu, M.; Nie, W.; Huang, D.-A.; Yu, Z.; Goldstein, T.; Anandkumar, A.; and Xiao, C. 2022. Test-time prompt tuning for zero-shot generalization in vision-language models. *Advances in Neural Information Processing Systems*, 35: 14274–14289.

Sun, H.; Ke, Q.; Cheng, M.; Wang, Y.; Li, D.; Gou, C.; and Cai, J. 2025. Point-Cache: Test-time Dynamic and Hierarchical Cache for Robust and Generalizable Point Cloud Analysis. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 1263–1275.

Sun, J.; Zhang, Q.; Kailkhura, B.; Yu, Z.; Xiao, C.; and Mao, Z. M. 2022. Benchmarking robustness of 3d point cloud recognition against common corruptions. *arXiv preprint arXiv:2201.12296*.

Uy, M. A.; Pham, Q.-H.; Hua, B.-S.; Nguyen, T.; and Yeung, S.-K. 2019. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, 1588–1597.

Wang, D.; Shelhamer, E.; Liu, S.; Olshausen, B.; and Darrell, T. 2020. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*.

Wang, D.; Shelhamer, E.; Liu, S.; Olshausen, B.; and Darrell, T. 2021. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*.

Wang, Y.; Cheraghian, A.; Hayder, Z.; Hong, J.; Ramasinghe, S.; Rahman, S.; Ahmedt-Aristizabal, D.; Li, X.; Petersson, L.; and Harandi, M. 2024. Backpropagation-free network for 3d test-time adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 23231–23241.

Wu, T.; Zhang, J.; Fu, X.; Wang, Y.; Ren, J.; Pan, L.; Wu, W.; Yang, L.; Wang, J.; Qian, C.; Lin, D.; and Liu, Z. 2023. OmniObject3D: Large-Vocabulary 3D Object Dataset for Realistic Perception, Reconstruction and Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 803–814.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3D ShapeNets: A Deep Representation for Volumetric Shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Xue, L.; Gao, M.; Xing, C.; Martín-Martín, R.; Wu, J.; Xiong, C.; Xu, R.; Niebles, J. C.; and Savarese, S. 2023. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1179–1189.

Xue, L.; Yu, N.; Zhang, S.; Panagopoulou, A.; Li, J.; Martín-Martín, R.; Wu, J.; Xiong, C.; Xu, R.; Niebles, J. C.; et al. 2024. Ulip-2: Towards scalable multimodal pre-training for 3d understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 27091–27101.

Yoon, H. S.; Yoon, E.; Tee, J. T. J.; Hasegawa-Johnson, M.; Li, Y.; and Yoo, C. D. 2024. C-TPT: Calibrated Test-Time Prompt Tuning for Vision-Language Models via Text Feature Dispersion. *arXiv preprint arXiv:2403.14119*.

Yuan, J.; Zhang, B.; Gong, K.; Yue, X.; Shi, B.; Qiao, Y.; and Chen, T. 2024. Reg-TTA3D: Better Regression Makes Better Test-Time Adaptive 3D Object Detection. In *European Conference on Computer Vision*, 197–213. Springer.

Zhang, C.; Xu, K.; Liu, Z.; Peng, Y.; and Zhou, J. 2025. Scap: Transductive test-time adaptation via supportive clique-based attribute prompting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 30032–30041.

Zhang, M.; Levine, S.; and Finn, C. 2022. Memo: Test time robustness via adaptation and augmentation. In *NeurIPS*.

Zhao, J.; Jiang, H.; Song, H.; Xiao, J.; and Gong, D. 2025. Dˆ 3CTTA: Domain-Dependent Decorrelation for Continual Test-Time Adaption of 3D LiDAR Segmentation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 11864–11874.

Zhou, J.; Wang, J.; Ma, B.; Liu, Y.-S.; Huang, T.; and Wang, X. 2024. Uni3d: Exploring unified 3d representation at scale. In *International Conference on Learning Representations (ICLR)*.

Zhu, X.; Zhang, R.; He, B.; Guo, Z.; Zeng, Z.; Qin, Z.; Zhang, S.; and Gao, P. 2023. Pointclip v2: Prompting clip and gpt for powerful 3d open-world learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2639–2650.

Zou, T.; Qu, S.; Li, Z.; Knoll, A.; He, L.; Chen, G.; and Jiang, C. 2024. Hgl: Hierarchical geometry learning for test-time adaptation in 3d point cloud segmentation. In *European Conference on Computer Vision*, 19–36. Springer.