SHAPEDINO: SCALING SELF-SUPERVISED REPRESENTATION LEARNING FOR 3D SHAPES

Anonymous authors
Paper under double-blind review

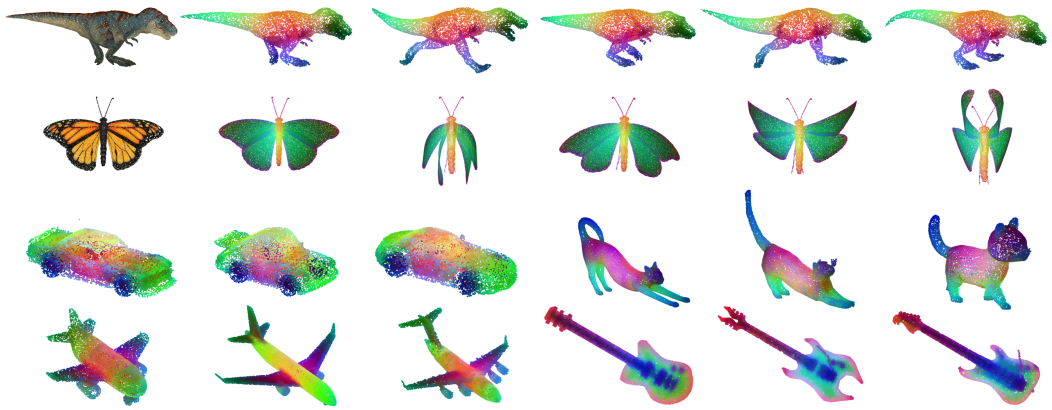

Figure 1: ShapeDINO learns consistent dense 3D representations across both temporal variations of the same object (top) and different instances within the same category (bottom).

ABSTRACT

Large-scale self-supervised learning has enabled general-purpose representations in language and vision. We investigate how far this paradigm can advance reusable representations of 3D shapes. We introduce ShapeDINO, a family of self-supervised point cloud encoders built on a standard vision Transformer backbone, ranging from Tiny to a billion-parameter Giant. ShapeDINO combines global and patch-level self-supervised learning to jointly learn object-level semantics and dense local features. To support pretraining at scale, we curate a dataset of 1.5 million 3D assets through quality filtering, geometric deduplication, and semantic rebalancing. We further distill the largest model into smaller variants to accommodate different computational budgets. Under frozen-feature evaluation, ShapeDINO achieves state-of-the-art performance on multiple classification and part-segmentation benchmarks, matching or surpassing the evaluated weakly supervised baselines. Additionally, we demonstrate that ShapeDINO can be seamlessly integrated into downstream tasks such as representation alignment, point segmentation and completion, yielding consistent performance improvements. These results demonstrate that large-scale self-supervised learning can yield reusable 3D representations that benefit diverse downstream tasks.

1 INTRODUCTION

Learning task-agnostic representations from large-scale unlabeled data has become a powerful paradigm in language and vision. Autoregressive language models acquire transferable capabilities through next-token prediction, while visual encoders such as DINOv2 (Oquab et al., 2023) and DINOv3 (Siméoni et al., 2025) learn general features through self-supervised pretraining. These advances motivate a similar goal for 3D shapes: learning a reusable encoder that captures both object-level semantics and fine-grained local structure, providing a shared foundation for diverse 3D tasks.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Early masked point modeling approaches, such as Point-BERT (Yu et al., 2022) and Point-MAE (Pang et al., 2022), demonstrated the effectiveness of self-supervised pretraining for 3D shapes. A parallel line of multimodal pretraining, including ULIP (Xue et al., 2023), ULIP-2 (Xue et al., 2024), OpenShape (Liu et al., 2023), and Uni3D (Zhou et al., 2024), scaled 3D representation learning through alignment with pretrained image-text models. More recently, Sonata (Wu et al., 2025) advanced point-cloud self-distillation, Concerto (Zhang et al., 2026a) combined it with 2D–3D joint learning, and Utonia (Zhang et al., 2026b) extended pretraining across diverse point-cloud domains. Complementing these directions, we investigate a focused question: how far can object-centric self-supervised pretraining be scaled to learn transferable global and dense local 3D representations?

Addressing this question requires more than increasing model capacity. A reusable shape encoder should learn global semantic while preserving local features for dense prediction. At the same time, multi-crop learning on irregular point clouds can incur repeated spatial grouping, increasing the cost of large-scale training. Expanding web-sourced training data also requires controlling quality and redundancy while maintaining geometric diversity. These challenges motivate a coordinated approach to backbone design, self-supervised training, and data curation.

We introduce ShapeDINO, a family of self-supervised 3D shape encoders ranging from 6.9M to 1.1B parameters. Building on DINOv2 (Oquab et al., 2023), our approach combines a scalable transformer backbone with multi-scale self-distillation at both object and patch levels, jointly learning global and dense local representations. To make multi-crop training efficient, we reuse spatial grouping across crops rather than reconstructing local neighborhoods independently for each crop. We support this training with a curated corpus of 1.5 million 3D assets from Objaverse-XL (Deitke et al., 2023a), emphasizing quality, diversity, and reduced redundancy. We pretrain our largest encoder from scratch and distill its representations into smaller variants, making the resulting model family accessible across different computational budgets.

Frozen-feature evaluations demonstrate the quality of both global and local representations learned by ShapeDINO. Across multiple classification and part-segmentation benchmarks, our models match or surpass the evaluated multimodally pretrained baselines. We assess global representations through kNN and linear probing for classification, and local representations through linear probing for part segmentation. These results show that a single pretrained encoder supports both global recognition and dense prediction without task-specific backbone fine-tuning. We further examine the effects of model scaling and distillation on representation quality.

Beyond frozen-feature evaluation, we investigate transfer to 3D generation, segmentation, and completion. We use ShapeDINO features as representation-alignment targets for training a 3D generation model, and integrate our pretrained encoders into downstream segmentation and completion models. Across all three applications, ShapeDINO improves task performance over the compared pretrained encoders. These findings demonstrate the utility of our representations both as a source of feature-level supervision and as an initialization for downstream learning, extending their benefits to tasks with distinct learning objectives.

In summary, our contributions are threefold:

- We introduce ShapeDINO, a family of self-supervised 3D shape encoders ranging from 6.9M to 1.1B parameters, learning transferable object-level and dense local representations.
- We develop a scalable object-centric pretraining recipe that combines efficient self-distillation with a quality-controlled, diverse corpus of 1.5M 3D assets.
- We systematically evaluate model scaling and distillation, and demonstrate transferability across frozen-feature benchmarks and diverse downstream applications.

2 Related Work

3D Representation Learning Recent 3D representation learning has been largely inspired by successful 2D pre-training paradigms, including masked modeling (He et al., 2022), vision-language contrastive learning (Radford et al., 2021; Zhai et al., 2023; Tschannen et al., 2025), and self-distillation (Zhou et al., 2022; Caron et al., 2021; Oquab et al., 2023; Siméoni et al., 2025). One line of work follows the masked modeling paradigm: Point-BERT (Yu et al., 2022) and Point-

MAE (Pang et al., 2022) learn representations by predicting discrete tokens or reconstructing masked geometry, while Point-M2AE (Zhang et al., 2022), I2P-MAE (Zhang et al., 2023), and ReCon (Qi et al., 2023) further extend this direction with multi-scale modeling, 2D priors, and contrastive objectives. Another line of work leverages semantic priors from images and language through multimodal alignment. ULIP (Xue et al., 2023) aligns 3D representations with pretrained image-text spaces, while ULIP-2 (Xue et al., 2024), OpenShape (Liu et al., 2023), and Uni3D (Zhou et al., 2024) further scale this paradigm in terms of data and model capacity, although the learned representations remain substantially guided by external vision-language priors. Complementary to these approaches, self-distillation methods learn directly from latent-space targets. Sonata (Wu et al., 2025), Concerto (Zhang et al., 2026a), and Utonia (Zhang et al., 2026b) extend this paradigm from scene-level point clouds to cross-modal and multi-domain pre-training, highlighting the potential of learning generalizable 3D representations in feature space.

Scaling 3D Foundation Models Scaling model capacity and training data has driven substantial progress in 2D vision, and this paradigm has recently been extended to 3D. For representation learning, OpenShape (Liu et al., 2023) investigates scaling both data and backbones for multimodal pretraining, while Uni3D (Zhou et al., 2024) adopts a standard ViT architecture and scales 3D encoders to 1B parameters. For 3D segmentation, Point-SAM (Zhou et al., 2025) employs a scalable data engine to train a promptable segmentation model, while P3-SAM (Ma et al., 2025) further scales training to nearly 3.7M segmented 3D objects. Scaling has been explored even more extensively in 3D generation: CLAY (Zhang et al., 2024), TripoSG (Li et al., 2025b), and TRELLIS (Xiang et al., 2025) improve generation quality by scaling generative backbones and training data, while Sparc3D (Li et al., 2026), Direct3D-S2 (Wu et al., 2026), and TRELLIS.2 (Xiang et al., 2026) leverage sparse 3D representations to scale generation resolution to 1024^3 and even 1536^3. Together, these advances highlight model, data, and spatial-resolution scaling as key ingredients for building general-purpose 3D foundation models.

3D Data Scaling and Curation. High-quality and diverse data is essential for large-scale self-supervised learning. In 2D vision, DINOv2 (Oquab et al., 2023) and DINOv3 (Siméoni et al., 2025) construct LVD-142M and LVD-1689M with filtering and deduplication to improve data quality and diversity. Although 3D data has been scaled from ShapeNet (Chang et al., 2015) to web-scale collections such as Objaverse (Deitke et al., 2023b) and Objaverse-XL (Deitke et al., 2023a), these datasets remain noisy and exhibit highly imbalanced category distributions. OpenShape (Liu et al., 2023), TRELLIS (Xiang et al., 2025), and TripoSG (Li et al., 2025b) improve data quality through text filtering, aesthetic scoring, and automated curation pipelines. While the semantic distribution of pre-training 3d data remains less explored, we develop a dedicated curation pipeline for large-scale 3D self-supervised pre-training, inspired by data curation practices in 2D vision.

3 METHOD

We present ShapeDINO in this section as follows. In Sec. 3.1, we first present our scalable backbone architecture and its key design choices. In Sec. 3.2, we build an efficient self-distillation framework for representation learning upon this backbone. Finally, in Sec. 3.3, we describe our data curation pipeline for constructing a high-quality self-supervised pretraining dataset from large-scale and noisy Web 3D data.

3.1 A SCALABLE BACKBONE FOR 3D SHAPES

For large-scale object-level 3D representation learning, we seek a backbone with efficient local encoding, predictable computational cost, and straightforward model scaling. We therefore adopt a compact point tokenizer followed by a vanilla Vision Transformer, separating local patch encoding from global feature modeling while allowing the patch configuration and model capacity to be specified independently.

Point Tokenizer. Given a point cloud $P = [X, N, C] \in \mathbb{R}^{N\times 9}$ containing coordinates X, normals N, and colors C, we follow Uni3D (Zhou et al., 2024) to sample M patch centers S using farthest point sampling $S = \text{FPS}(X, M)$ and group the K nearest neighbors of each center $G = \text{KNN}(S, P, K)$. A residual PointNet-style encoder (Qi et al., 2017) processes these features with shared linear layers and GELU activations. We also remove BatchNorm (Ioffe & Szegedy,

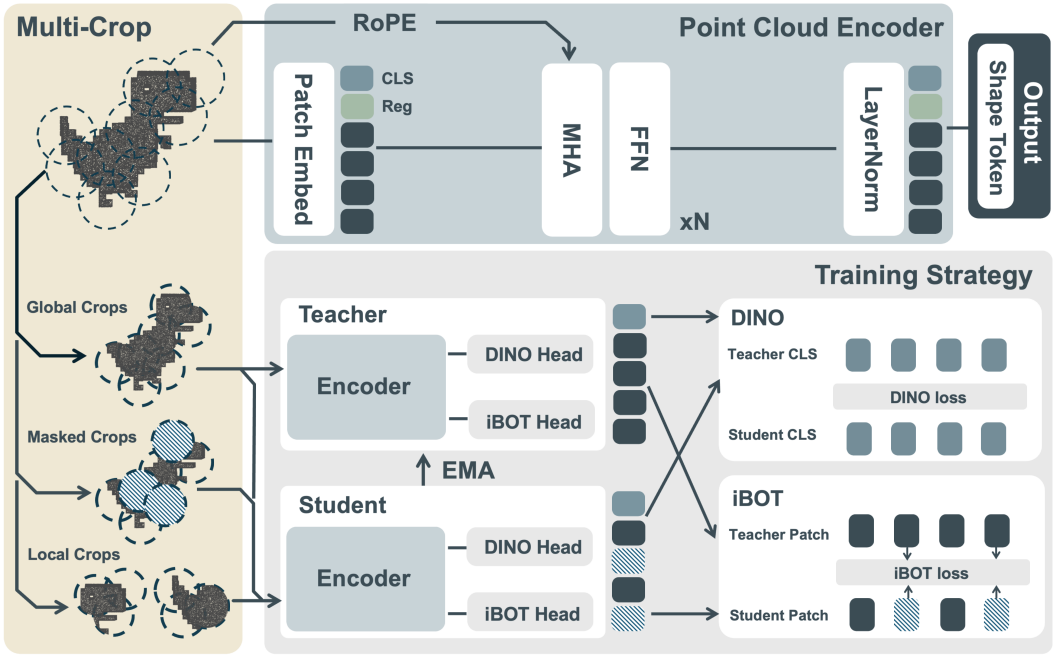

Figure 2: Overview of ShapeDINO, including multi-crop construction, the point-cloud encoder, and the self-distillation training framework.

2015) to simplify the tokenizer and avoid maintaining running normalization statistics during EMA-based self-distillation.

Transformer Backbone. The patch tokens are then processed by a standard Vision Transformer with 3D rotary position embeddings (RoPE), following DINOv3 (Siméoni et al., 2025) and Utonia (Zhang et al., 2026b). RoPE uses patch-center coordinates to rotate attention queries and keys, allowing spatial relationships between patches to be encoded directly during feature interaction. Such a design allows us to scale the Transformer width, depth, and attention-head configuration while keeping the tokenizer and patch construction unchanged, yielding the model family summarized in Table 2.

3.2 Self Distillation for 3D Representation Learning

We follow the training paradigm of DINOv2, combining the global DINO objective with the patch-level iBOT objective to pretrain our encoder on unlabeled 3D shapes. To address the high cost of geometric grouping in point clouds, we design an efficient multi-crop strategy based on shared spatial patches to support large-scale training.

Efficient 3D Multi-Crop. Unlike regular image grids, constructing point-cloud patches requires FPS and kNN operations. Independently patchifying each crop would repeatedly perform these geometric operations and incur substantial computational overhead. To avoid this redundancy, we decouple geometric grouping from multi-crop sampling. Specifically, we apply FPS and KNN only once to the full point cloud to construct a shared grouping G of point patches, and then sample spatial regions containing different numbers of patches to form global and local crops C_v with crop size r_v

$$C_v \sim \mathrm{Crop}(G; r_v), \qquad |C_v| = \lfloor r_v M \rfloor, \qquad v = 1, ..., V \tag{1}$$

Each crop is independently augmented before being fed into the encoder, without repeating the geometric grouping procedure. For efficient batching, we sample a patch count for each type of global and local crop within each per-device mini-batch and share it across all shapes in the batch, while crop locations and data augmentations are still sampled independently for each shape. Consequently, crops of the same type have identical sequence lengths within a batch, enabling dense batched computation without padding. The patch counts are resampled across training iterations and distributed workers, preserving the diversity of multi-scale crops while maintaining computational efficiency.

Teacher–Student Self-Distillation. Following DINOv2, we adopt a teacher–student architecture in which the teacher is updated as an exponential moving average of the student parameters. The teacher processes only unmasked global crops, whereas the student processes both global and local crops, with patch masking applied to the global crops. Training combines the global DINO objective $\mathcal{L}_{\text{DINO}}$ with the patch-level iBOT objective $\mathcal{L}_{\text{iBOT}}$: DINO aligns object-level representations across different crops, while iBOT uses the teacher outputs at corresponding patch locations within the same global crop to supervise the student predictions for masked patches. Together, these objectives encourage crop-consistent global representations and context-aware local representations.

$$\mathcal{L} = \mathcal{L}_{\text{DINO}} + \lambda_{\text{iBOT}}\mathcal{L}_{\text{iBOT}} \tag{2}$$

Knowledge Distillation. Knowledge distillation has been widely used in DINOv2(Oquab et al., 2023) and DINOv3 (Siméoni et al., 2025) to transfer representations from large pretrained models to smaller architectures. While this remains relatively underexplored in existing 3d representation learning methods. We therefore first pretrain our largest ShapeDINO-Giant model from scratch and freeze it as the teacher for distilling the smaller variants. During distillation, we retain the same self-supervised objectives as in pretraining and maintain an EMA copy of each student as the final model. In this way, smaller models benefit from the stronger representation learned by the Giant teacher and can outperform models trained from scratch, while substantially reducing the cost of constructing the full ShapeDINO model family.

3.3 Data Curation for Large Scale 3D Pretraining

High-quality data is critical for scaling self-supervised 3D representation learning. However, web-scale 3D collections are often noisy, highly redundant, and strongly imbalanced. We therefore curate a large-scale training corpus from Objaverse-XL (Deitke et al., 2023a), targeting data quality, diversity, and distributional balance.

Our curation pipeline consists of three stages. First, we filter assets with severe geometric artifacts, such as dominant support planes, detached components, and degenerate structures, which can interfere with point-cloud sampling. Second, we perform geometry-aware deduplication using coarse geometric similarity, removing near-identical assets. Third, we hierarchically cluster the remaining shapes using pretrained OpenShape (Liu et al., 2023) embeddings, and apply cluster-balanced sampling to reduce long-tail bias. Starting from 2.5M assets, the resulting corpus contains approximately 1.5M cleaner, less redundant, and better-balanced shapes. Detailed filtering criteria, duplicate verification, and clustering procedures are provided in the appendix.

4 Experiments

We evaluate the global and local representations learned by ShapeDINO through object classification and part segmentation, respectively, in Section 4.1. We then study model and data scaling in Section 4.2 and ablate key design choices in Section 4.3. We compare against prior self-supervised 3D representation learning methods, including Point-MAE (Pang et al., 2022), Point-M2AE (Zhang et al., 2022), Sonata (Wu et al., 2025), and Utonia (Zhang et al., 2026b), as well as multimodal alignment methods, including OpenShape (Liu et al., 2023), and Uni3D (Zhou et al., 2024). For a fair comparison, we re-evaluate all baselines using their official pretrained checkpoints under the same evaluation protocol. We use kNN and linear probing for object classification and point-wise linear probing for part segmentation. Full evaluation details are provided in the appendix.

4.1 Global and Local Representation Comparison

Global representations. We evaluate global representations on ModelNet40(Wu et al., 2015) and ScanObjectNN PB-T50-RS split (Uy et al., 2019) using frozen encoders, with kNN classification and linear probing as evaluation protocols. As shown in Table 1, ShapeDINO-G consistently outperforms the compared self-supervised baselines under both protocols on both datasets, achieving linear-probing accuracies of 91.3% and 84.1% on ModelNet40 and ScanObjectNN, respectively. Notably, it also achieves the best linear-probing performance among all compared methods, including those pretrained with multimodal alignment. Even the smaller ShapeDINO-B reaches 91.1%

Table 1: Comparison of global and local representations on classification and part segmentation benchmarks. We report mAcc under kNN and linear probing on ModelNet40 and ScanObjectNN, and class mIoU, instance mIoU, and part mIoU under linear probing on ShapeNetPart and PartNetE.

Method	Arch	#Params	#Data	Classification		Segmentation	
				MN40 kNN / LP	ScanObj kNN / LP	SNPart c. / i.mIoU	PartNetE p.mIoU
Weakly supervised							
OpenShape-PB	ViT	32.3M	876K	**89.3 / 90.0**	46.6 / 56.6	78.3 / 79.8	41.1
OpenShape-SC	SpConv	41.3M	876K	89.1 / 89.4	46.1 / 57.1	74.8 / 75.7	38.3
Uni3D	ViT-G	1.02B	876K	84.1 / 87.3	**75.5 / 80.0**	**80.8 / 82.5**	**50.2**
Self-supervised							
Point-MAE	ViT	21.8M	51K	78.7 / 87.8	46.6 / 68.1	80.6 / 82.3	38.4
Point-M2AE	HierViT	12.7M	51K	81.4 / 88.2	52.1 / 64.1	79.7 / 81.3	39.3
Sonata	PTv3	108.5M	140K	59.6 / 78.6	27.9 / 40.2	76.1 / 78.1	32.1
Utonia	PTv3	137.3M	1.2M	80.7 / 88.6	62.5 / 75.3	80.9 / 82.8	45.7
ShapeDINO-Tiny	ViT-Tiny	6.9M	1.5M	80.0 / 89.6	67.8 / 79.3	82.1 / 83.2	48.9
ShapeDINO-B	ViT-B	86.9M	1.5M	81.4 / 91.1	71.0 / 81.4	82.9 / 84.1	50.5
ShapeDINO-G	ViT-G	1.14B	1.5M	**83.0 / 91.3**	**75.4 / 84.1**	**83.4 / 84.5**	**51.3**

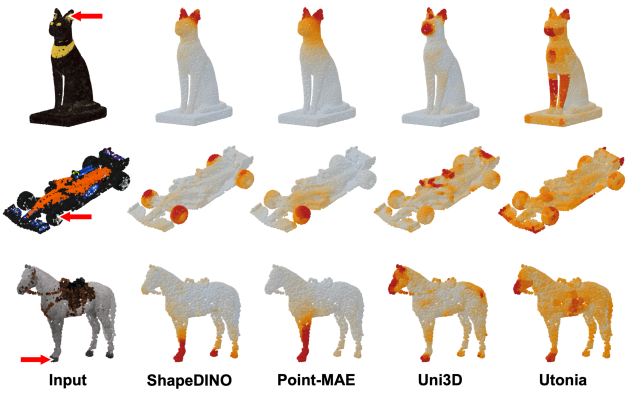

Figure 3: Query conditioned attention map. Warmer colors indicate stronger responses to the selected query patch.

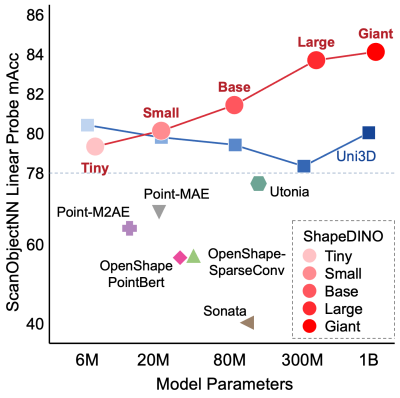

Figure 4: Scaling behavior comparison between different models.

and 81.4%, surpassing prior self-supervised methods by clear margins. In kNN evaluation, multi-modal methods remain advantageous on ModelNet40, while ShapeDINO-G performs comparably to Uni3D on ScanObjectNN. Overall, these results indicate that purely 3D self-supervised pretraining can learn highly linearly separable global representations without relying on image-text supervision.

Local representations. We evaluate local representations on ShapeNetPart (Yi et al., 2016) and PartNetE (Mo et al., 2019) using point-wise linear probing. For point-token-based models, local features are interpolated to the input points using 3NN distance-weighted interpolation; for sparse-voxel encoders, multi-scale voxel features are mapped back to the corresponding input points. We then train only a point-wise linear classifier for part prediction. By avoiding learnable feature-propagation modules and nonlinear decoders, this protocol provides a direct measure of the linear separability of pretrained local features. As shown in Table 1, ShapeDINO-G achieves 83.4/84.5 category-/instance-averaged mIoU on ShapeNetPart and 51.3 mIoU on PartNetE, outperforming all compared methods. Strong performance is retained at smaller scales: ShapeDINO-B also surpasses all baselines across the reported segmentation metrics, while even ShapeDINO-Tiny outperforms them on ShapeNetPart with only 6.9M parameters. These results show that ShapeDINO learns strong local representations in addition to object-level features.

Qualitative analysis. To further examine the structural relationships captured by different representations, we visualize attention responses in Fig. 3. Given a patch on a selected object part, ShapeDINO frequently responds to corresponding or repeated structures elsewhere on the shape, such as multiple wheels or horse hooves, rather than being restricted to the immediate query neigh-

Model	Depth	Dim.	Heads	Head Dim.	Params.	ScanObjectNN	ShapeNetPart
ShapeDINO-Tiny	12	192	4	48	6.9M	67.8 / 79.3	82.1 / 83.2
ShapeDINO-Small	12	384	8	48	22.9M	70.4 / 80.1	82.4 / 84.0
ShapeDINO-Base	12	768	8	96	86.9M	71.0 / 81.4	82.9 / 84.1
ShapeDINO-Large	24	1152	12	96	384.5M	72.8 / 83.7	82.8 / 84.3
ShapeDINO-Giant	40	1536	16	96	1.14B	**75.4 / 84.1**	**83.4 / 84.5**

Table 2: **Model Scaling.** Performance across ShapeDINO with increasing Transformer capacity.

Variant	Scan. ↑	Part.E ↑
Raw 1.5M	58.3	41.2
Balance100K	60.3	40.4
Balance1.5M	**61.2**	**43.4**

Table 3: **Data Ablation.** Effect of dataset balance and scale on ShapeDINO-S scratch.

Variant	Scan. ↑	Part.E ↑
BatchNorm	42.0	26.5
AbsPE	60.7	42.8
Ours	**61.2**	**43.4**

Table 4: **Backbone Design.** Ablation of BatchNorm and position encoding on ShapeDINO-S scratch.

Variant	Scan. ↑	Part.E ↑
Giant	**75.4**	**51.3**
Base Scratch	68.4	48.6
Base Distill	71.0	50.5

Table 5: **Distillation.** Comparison between distillation and training from scratch.

borhood. In comparison, Point-MAE tends to focus more strongly on the queried local region, while Uni3D and Utonia often exhibit broader and less part-specific responses. These results suggest that ShapeDINO captures more coherent part-level structural relationships.

4.2 Scaling Ability

We study the scaling behavior of ShapeDINO by increasing the Transformer capacity from 6.9M to 1.14B parameters. As shown in Table 2, performance improves consistently with model size on both ScanObjectNN and ShapeNetPart. The ScanObjectNN linear-probe mAcc, for example, increases from 79.3 with ShapeDINO-Tiny to 84.1 with ShapeDINO-Giant. Figure 4 places this trend in the context of existing 3D representation models. ShapeDINO shows a clear improvement as capacity increases and continues to benefit from scaling into the billion-parameter regime. This suggests that our self-supervised pretraining recipe can effectively translate increased model capacity into stronger 3D representations.

4.3 Ablation Study

To validate the key components of our approach, we conduct ablation studies on data curation, backbone design, and knowledge distillation. Additional ablations on training strategies and model design are provided in the Appendix.

Data Curation. We study the effects of data curation and scale by training our Small model from scratch on three data variants: 1.5M samples from the raw data, a balanced 100K subset, and the full balanced 1.5M dataset. At the same data scale, the balanced 1.5M dataset consistently outperforms the raw 1.5M data. Increasing the balanced training set from 100K to 1.5M further improves performance, particularly on Part.E. These results demonstrate the importance of both data curation and scale for 3D self-supervised pretraining.

Backbone Design. We ablate two key design choices in our backbone: BatchNorm in the point tokenizer and the positional encoding scheme. Introducing BatchNorm leads to a substantial performance drop, while replacing RoPE with absolute positional embeddings results in consistently lower performance. These results validate our simplified tokenizer and the use of RoPE for 3D representation learning.

Knowledge Distillation. We evaluate knowledge distillation by comparing a Base model trained from scratch with one distilled from our Giant teacher. Distillation improves performance from 68.4 to 71.0 on ScanObjectNN. and from 48.6 to 50.5 on PartNetE, showing that the representations learned by the large teacher can be effectively transferred to a smaller model.

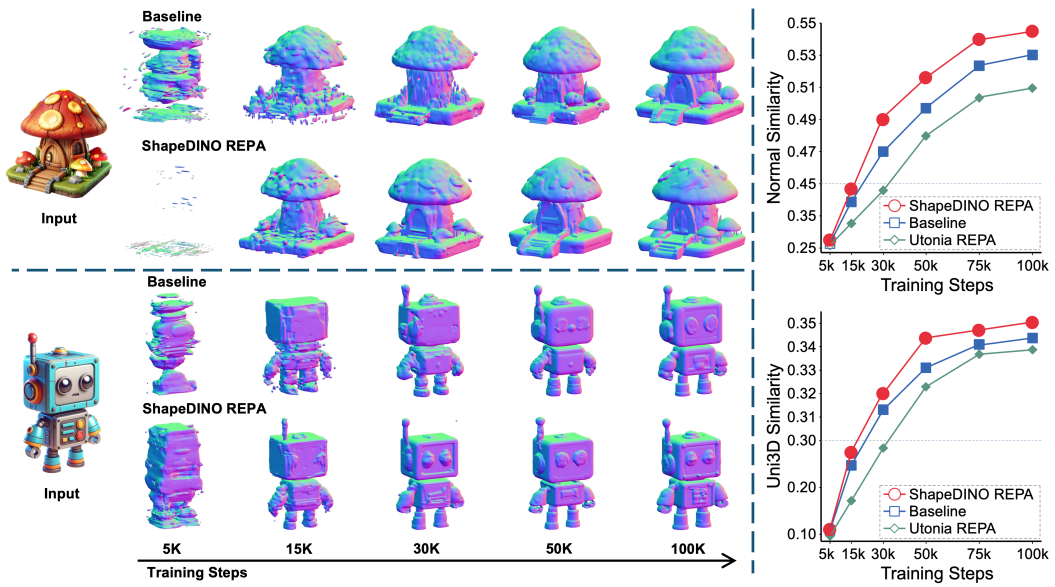

Figure 5: Application to 3D generation via representation alignment. Left: qualitative results of the baseline and ShapeDINO-based REPA at different training steps. Right: Normal Similarity and Uni3D Similarity curves for the baseline and REPA with different representation teachers.

5 Downstream Applications

To further explore the transferability of ShapeDINO, we apply its pretrained representations to a diverse set of downstream applications. Specifically, in Sec. 5.1, we use ShapeDINO as a representation teacher to accelerate the training of 3D generative models; in Sec. 5.2, we employ it as the encoder for promptable 3D segmentation; and in Sec. 5.3, we apply it to point cloud completion.

5.1 Representation Alignment on 3D Shape Generation

Following REPA, we investigate whether ShapeDINO can serve as a representation teacher for 3D generative modeling. REPA aligns the intermediate features of a DiT model with those of a pretrained representation encoder, encouraging the emergence of semantic structure and improving optimization efficiency. Specifically, we use a pretrained TripoSG (Li et al., 2025b) VAE as the latent autoencoder and adopt a rectified-flow DiT based on CraftsMan3D (Li et al., 2025a). We compare three settings: a baseline without representation alignment, REPA using ShapeDINO features, and REPA using Utonia features. We evaluate generation quality using Normal Similarity proposed by ViLearn (Chen et al., 2026) and Uni3D (Zhou et al., 2024) Similarity.

As shown in Fig. 5, using ShapeDINO as the REPA teacher consistently improves both convergence speed and final generation quality. Qualitatively, the model recovers coherent geometry and fine object structures at earlier training stages than the baseline. Quantitatively, ShapeDINO-based REPA achieves higher Normal Similarity and Uni3D Similarity throughout training, reaching comparable quality with fewer optimization steps. In contrast, using Utonia as the teacher yields limited gains and remains below the baseline across training. These results suggest that ShapeDINO provides stronger semantic guidance for 3D shape generation and serves as an effective representation teacher for 3D generative models.

5.2 Promptable 3D Part Segmentation

We further evaluate the transferability of ShapeDINO on promptable 3D part segmentation with Point-SAM (Zhou et al., 2025). Point-SAM extends the promptable segmentation paradigm of SAM (Kirillov et al., 2023) to 3D point clouds by predicting segmentation masks conditioned on point prompts. We replace its point-cloud encoder with ShapeDINO-L while keeping the remaining framework unchanged. Following Point-SAM, we train on object-level dataset including Part-

Table 6: Point-SAM segmentation performance on PartNet-Mobility with different point encoders and initialization strategies. IoU@k denotes IoU with k point prompts.

Encoder	Initialization	IoU@1	IoU@3	IoU@5	IoU@6
Uni3D	Scratch	36.58	51.90	57.53	58.93
	Pretrained	38.93	51.48	55.58	57.38
	Δ Pretrain	+2.35	-0.42	-1.95	-1.55
ShapeDINO-L	Scratch	40.86	55.05	60.12	61.67
	Pretrained	**47.50**	**58.39**	**62.16**	**63.24**
	Δ Pretrain	**+6.64**	**+3.34**	**+2.04**	**+1.57**

Table 7: Point cloud completion results with different point encoders and initialization strategies on PCN, ShapeNet-34, and Unseen-21. We report F1 and CD-ℓ_1 ($\times 10^3$) for PCN, and F1 and CD-ℓ_2 ($\times 10^3$) for ShapeNet-34 and Unseen-21.

Encoder	Initialization	PCN		ShapeNet-34		Unseen-21	
		F1 ↑	CD-ℓ_1 ↓	F1 ↑	CD-ℓ_2 ↓	F1 ↑	CD-ℓ_2 ↓
PoinTr	Scratch	0.751	8.04	0.404	1.97	0.373	3.19
ShapeDINO-S	Random	0.695	8.94	0.432	1.16	0.386	2.52
	Pretrained	**0.781**	**7.48**	**0.454**	**1.12**	**0.397**	**2.32**

Net (Mo et al., 2019) and PartNet-Mobility (Xiang et al., 2020) and evaluate on the PartNet-Mobility test set.

As shown in Table 6, ShapeDINO-L outperforms Uni3D across all IoU metrics even when both encoders are trained from scratch. More importantly, pretrained initialization consistently improves ShapeDINO-L, yielding gains of 1.57–6.64 IoU points across different prompt settings. In contrast, Uni3D pretraining improves IoU@1 by 2.35 points but degrades performance under the remaining settings. With pretrained initialization, ShapeDINO-L further outperforms Uni3D across all metrics, demonstrating that its pretrained representations transfer effectively to promptable 3D part segmentation.

5.3 Point Cloud Completion

We evaluate the transferability of ShapeDINO on point cloud completion with PoinTr (Yu et al., 2021). We replace its original point encoder with ShapeDINO-S while keeping the remaining framework unchanged, and evaluate on PCN (Yuan et al., 2018), ShapeNet-34 (Chang et al., 2015), and Unseen-21.

As shown in Table 7, ShapeDINO-S is already competitive with random initialization, while pretraining consistently improves performance across all benchmarks. On PCN, it increases F1 from 0.695 to 0.781 and reduces CD-ℓ_1 from 8.94 to 7.48, with similar gains on ShapeNet-34 and Unseen-21. The pretrained ShapeDINO-S also outperforms the original PoinTr encoder on all reported metrics, demonstrating effective transfer and generalization to unseen categories.

6 Conclusion

In this work, we introduced ShapeDINO, a family of scalable self-supervised representation models for object-level 3D shape. We showed that combining a scalable Transformer architecture with efficient 3D self-distillation and large-scale data curation enables 3D representation learning to benefit consistently from increased model and data scale. Across a range of global and local representation benchmarks, ShapeDINO achieves strong performance while exhibiting clear scaling behavior. Beyond representation evaluation, we further demonstrated the versatility of the learned features by applying them to representation alignment for 3D generation, as well as point cloud segmentation and completion, where they consistently improve downstream performance. These results suggest that large-scale self-supervised pretraining can produce general and transferable 3D representations, providing a simple and scalable path toward general-purpose 3D foundation encoders.

AI USE STATEMENT

Generative AI tools, primarily ChatGPT and Codex, were used during this work to assist with manuscript writing and editing, including improving clarity and readability, translating and refining technical descriptions, and drafting or revising parts of the paper such as experimental descriptions and figure captions. We also used generative AI for brainstorming and feedback on the presentation and design of experiments. All AI-assisted suggestions, text, and interpretations were reviewed and verified by the authors, and all methodological, experimental, and decisions were made by the authors. We take full responsibility for the final content of this work.

REFERENCES

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9630–9640. IEEE, 2021.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Rui Chen, Jianfeng Zhang, Jing Lin, Xuanyu Yi, Yixun Liang, Guan Luo, Xiu Li, Zeming Li, and Ping Tan. Vilearn: Accelerating training convergence of image-to-3d generation via visibility learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27041–27051, 2026.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36:35799–35813, 2023a.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13142–13153, 2023b.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 15979–15988. IEEE, 2022.

Xianglong He, Zi-Xin Zou, Chia-Hao Chen, Yuan-Chen Guo, Ding Liang, Chun Yuan, Wanli Ouyang, Yan-Pei Cao, and Yangguang Li. Sparseflex: High-resolution and arbitrary-topology 3d shape modeling. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14822–14833. IEEE, 2025.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *2023 IEEE/CVF international conference on computer vision (ICCV)*, pp. 3992–4003. IEEE, 2023.

Weiyu Li, Jiarui Liu, Hongyu Yan, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. Craftsman3d: High-fidelity mesh generation with 3d native diffusion and interactive geometry refiner. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5307–5317. IEEE, 2025a.

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Zhihao Li, Yufei Wang, Heliang Zheng, Yihao Luo, and Bihan Wen. Sparc3d: Sparse representation and construction for high-resolution 3d shapes modeling. *Advances in Neural Information Processing Systems*, 38:118582–118600, 2026.

Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding. *Advances in neural information processing systems*, 36:44860–44879, 2023.

Changfeng Ma, Yang Li, Xinhao Yan, Jiachen Xu, Yunhan Yang, Chunshi Wang, Zibo Zhao, Yanwen Guo, Zhuo Chen, and Chunchao Guo. P3-sam: Native 3d part segmentation. *arXiv preprint arXiv:2509.06784*, 2025.

Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 909–918. IEEE, 2019.

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.

Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pp. 604–621. Springer, 2022.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Zekun Qi, Runpei Dong, Guofan Fan, Zheng Ge, Xiangyu Zhang, Kaisheng Ma, and Li Yi. Contrast with reconstruct: Contrastive 3d representation learning guided by generative pretraining. In *International Conference on Machine Learning*, pp. 28223–28243. PMLR, 2023.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Oriane Siméoni, Huy V Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1588–1597. IEEE, 2019.

Huy V Vo, Vasil Khalidov, Timothée Darcet, Théo Moutakanni, Nikita Smetanin, Marc Szafraniec, Hugo Touvron, Camille Couprie, Maxime Oquab, Armand Joulin, et al. Automatic data curation for self-supervised learning: A clustering-based approach. *arXiv preprint arXiv:2405.15613*, 2024.

Ruicheng Wang, Sicheng Xu, Yue Dong, Yu Deng, Jianfeng Xiang, Zelong Lv, Guangzhong Sun, Xin Tong, and Jiaolong Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details. *Advances in Neural Information Processing Systems*, 38:35928–35959, 2026.

Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Yikang Yang, Jiachen Qian, Siyu Zhu, Xun Cao, Philip Torr, Yao Yao, et al. Direct3d-s2: Gigascale 3d generation made easy with spatial sparse attention. *Advances in Neural Information Processing Systems*, 38:170778–170804, 2026.

Xiaoyang Wu, Daniel DeTone, Duncan Frost, Tianwei Shen, Chris Xie, Nan Yang, Jakob Engel, Richard Newcombe, Hengshuang Zhao, and Julian Straub. Sonata: Self-supervised learning of reliable point representations. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22193–22204. IEEE, 2025.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11094–11104. IEEE, 2020.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21469–21480. IEEE, 2025.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, et al. Native and compact structured latents for 3d generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14419–14429, 2026.

Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1179–1189. IEEE, 2023.

Le Xue, Ning Yu, Shu Zhang, Artemis Panagopoulou, Junnan Li, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, et al. Ulip-2: Towards scalable multimodal pre-training for 3d understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27081–27091. IEEE, 2024.

Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. *arXiv preprint arXiv:2410.06940*, 2024.

Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12498–12507, 2021.

Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19291–19300. IEEE, 2022.

Wentao Yuan, Tejas Khot, David Held, Christoph Mertz, and Martial Hebert. Pcn: Point completion network. In *2018 international conference on 3D vision (3DV)*, pp. 728–737. IEEE, 2018.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952. IEEE, 2023.

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions On Graphics (TOG)*, 43(4):1–20, 2024.

Renrui Zhang, Ziyu Guo, Peng Gao, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, and Hongsheng Li. Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. *Advances in neural information processing systems*, 35:27061–27074, 2022.

Renrui Zhang, Liuhui Wang, Yu Qiao, Peng Gao, and Hongsheng Li. Learning 3d representations from 2d pre-trained models via image-to-point masked autoencoders. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21769–21780. IEEE, 2023.

Yujia Zhang, Xiaoyang Wu, Yixing Lao, Chengyao Wang, Zhuotao Tian, Naiyan Wang, and Hengshuang Zhao. Concerto: Joint 2d-3d self-supervised learning emerges spatial representations. *Advances in Neural Information Processing Systems*, 38:69498–69522, 2026a.

Yujia Zhang, Xiaoyang Wu, Yunhan Yang, Xianzhe Fan, Han Li, Yuechen Zhang, Zehao Huang, Naiyan Wang, and Hengshuang Zhao. Utonia: Toward one encoder for all point clouds. *arXiv preprint arXiv:2603.03283*, 2026b.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *International Conference on Learning Representations (ICLR)*, 2022.

Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. In *International Conference on Learning Representations*, volume 2024, pp. 46766–46782, 2024.

Yuchen Zhou, Jiayuan Gu, Tung Chiang, Fanbo Xiang, and Hao Su. Point-sam: Promptable 3d segmentation model for point clouds. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 56850–56869, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/8f363c981c796cd781729e6c0117cadc-Paper-Conference.pdf.

A Model and Training Details

A.1 Model Configurations

Backbone. In addition to the configure provided in Tab. 2, all variants use GELU MLPs with an expansion ratio of 4, LayerNorm, LayerScale, one CLS token, and 4 register tokens. We adopt non-learned axial 3D RoPE with a base of 100, where the positional rotations are computed from the augmented patch centers and applied to the queries and keys of patch tokens.

Point Tokenizer. Each object is represented by 16,384 points with XYZ coordinates, surface normals, and RGB colors, resulting in a 9-dimensional point input. The point cloud is first centered and normalized to the unit sphere, which is sampled from 81920 points during training. We sample 512 patch centers with FPS and retrieve 64 of their closest neighbors. Point coordinates within each group are expressed relative to the corresponding patch center following PointNet (Qi et al., 2017), while normals and colors are retained.

We utilize the PointNet implementation in SparseFlex (He et al., 2025) with a fixed hidden dimension of 384. It consists of two residual fully connected blocks interleaved with group-wise max pooling, followed by a projection to the hidden dimension of the corresponding Transformer backbone.

Training Heads. We use separate DINO (Caron et al., 2021) and iBOT (Zhou et al., 2022) heads for CLS- and patch-level supervision, respectively. Both heads contain three-layer MLPs with a hidden dimension of 2048. The DINO head uses a 384-dim bottleneck with 32768 prototypes, while the iBOT head uses a 256-dim bottleneck with 16384 prototypes. Bottleneck features are ℓ_2-normalized before the prototype projection. The prototype spaces are shared across model scales, while the input dimensions of the heads follow the corresponding backbone widths.

A.2 Training Configurations

Optimization. We train all models for 50K optimizer steps using AdamW with learning rate of 1e-3 and a constant weight decay of 0.04. The learning rate is linearly warmed up from 1e-5. for the first 5K steps and kept constant afterwards, as suggested in DinoV3 (Siméoni et al., 2025). Training uses BF16 mixed precision and gradient clipping with a maximum norm of 3.0. For pretraining of ShapeDINO-G, the teacher is updated as an exponential moving average (EMA) of the student, with the momentum following a cosine schedule from 0.996 to 1.0. The teacher temperature is linearly warmed up from 0.04 to 0.07 during the first 5K steps and is kept at 0.07 afterwards. For distillation, the pretrained ShapeDINO-G is used as a frozen teacher with a fixed temperature of 0.07, while each student additionally maintains its own EMA model using the same momentum schedule. The student temperature is fixed to 0.1 in all distillation. Refer Table 8 for more training settings and budgets.

Objectives and Masking. Training combines global DINO supervision, patch-level iBOT supervision, and KoLeo regularization on the student global CLS features. The DINO and iBOT objectives both use a weight of 1. Each of the two teacher global views supervises the student global and local views, excluding the matching global-view pair.

For patch-level supervision, we select half of the global-view samples on each rank and assign masking ratios between 0.1 and 0.5. During self-supervised pretraining, the selected student global patch tokens are replaced by a learnable mask token, whereas the teacher always receives the unmasked views. During distillation, the student inputs remain fully unmasked; the same sampled indices are only used to select the patch locations on which distillation is computed. Thus, the patch objective in the distillation stage directly matches corresponding teacher and student representations rather than performing masked-input reconstruction.

Data Augmentation. Global and local crops use the same augmentation recipe, with augmentation parameters sampled independently for each crop. Each crop is first re-centered and normalized using all points contained in the selected patch groups. We then apply random reflection with probability 0.5, a random 3D rotation with probability 0.8, and Gaussian jitter with a standard deviation of 0.005 and clipping at 0.02. Normal and color channels are independently dropped with probability 0.2. Geometric transformations are consistently applied to patch centers, local coordinates, and normals.

Model	Training Method	GPUs	Batch/GPU	Global Batch	Steps	Runtime	GPU-hours
ShapeDINO-Giant	Pretrain	64	24	1536	50K	86.98 h	5566.8
ShapeDINO-Tiny	Distill	16	32	512	50K	31.49 h	503.8
ShapeDINO-Small	Distill	16	32	512	50K	30.36 h	485.7
ShapeDINO-Base	Distill	16	32	512	50K	32.54 h	520.6
ShapeDINO-Large	Distill	32	28	896	50K	50.65 h	1620.8

Table 8: Training resources for ShapeDINO. Runtime denotes the accumulated runtime recorded by the experiment logger and includes data loading, checkpointing, and other training overhead. GPU-hours are estimated by multiplying the recorded runtime by the number of GPUs.

A.3 Multi Crop Construction

Given an input point cloud of 16,384 points, we perform FPS and KNN only once to construct 512 point groups, each containing 64 neighbors. All global and local crops are subsequently sampled from these shared groups. For each crop slot, we first sample a crop ratio r shared by all objects in the local batch of the current rank and set the number of selected patches to $m = \lceil 512r \rceil$. For each object, we independently sample one patch center as the crop anchor and select the m nearest patch centers according to Euclidean distance. We construct two global crops with $r \sim \mathcal{U}(0.4, 1.0)$ and eight local crops with $r \sim \mathcal{U}(0.1, 0.4)$, corresponding to 205–512 and 52–205 patch tokens, respectively.

Sharing m within each crop slot ensures that all objects in the batch have the same sequence length for that view, allowing each crop to be represented as a dense tensor without padding or truncation. Different crop slots independently sample their ratios and can therefore have different sequence lengths. The crop ratio is shared only within a local rank and does not require synchronization across GPUs.

The teacher and student share the same geometric groups, global-crop indices, and augmentations. Consequently, their global patch tokens have a direct correspondence along the crop, object, and patch dimensions, which enables straightforward patch-level distillation. The teacher processes only the two global crops, whereas the student processes both global crops and all eight local crops. This construction both avoids repeated geometric grouping and preserves explicit teacher–student patch correspondence.

A.4 Efficient Implementation

Our training code is implemented with PyTorch Lightning. We use DeepSpeed ZeRO Stage 2 for Giant model pretraining and the S/B/L distillation training, while the Tiny model is distilled with standard DDP. All student models use BF16 mixed precision, while EMA model states are maintained in FP32. Activation checkpointing is applied to each Transformer block for ShapeDINO-G and the S/B/L students, and is disabled for the Tiny model due to its small size.

A.5 Training Resources

All trainings are conducted on a slurm cluster with NVIDIA A800-SXM4-80GB GPUs with NVLink. Table 8 summarizes the training resources for each model, as well as the batch size. ShapeDINO-G is pretrained on 64 GPUs for approximately 3.6 days. Distillation of the Tiny, S, and B models uses 16 GPUs and takes approximately 31.5, 30.4, and 32.5 hours, respectively, while the L model is distilled on 32 GPUs for approximately 50.6 hours.

B Dataset Construction and Curation

We construct our pretraining corpus through a multi-stage curation pipeline for ObjaverseXL (Deitke et al., 2023a), which consists of geometric quality filtering, geometric deduplication, and feature-space distribution rebalancing.

B.1 Data Sources and Preprocessing

Our data comes from ObjaverseXL (Deitke et al., 2023a). We first perform format unification, following the data preprocessing method proposed by TRELLIS (Xiang et al., 2025), we use blender to load each asset and export GLB format asset, set motion to the first frame if existed. Then we perform point sampling of 81920 points with normal and color.

B.2 Quality Filtering

We first perform filter following TripoSG (Li et al., 2025b), and we also filter data with large planar, which finally return 260K objects to us. The remained 3D collections still contain multi-object scenes that are undesirable for object-centric representation learning. We therefore apply an additional geometry-based filter based on the spatial connectivity of sampled surfaces.

For each 81920-point cloud, we normalize the coordinates by its longest bounding-box extent and voxelize all points on a 64^3 occupancy grid. We then compute connected components using 26-connectivity. Let n_k denote the number of sampled points assigned to the k-th connected component and N the total number of points. We define its relative surface mass as $r_k = \frac{n_k}{N}$. A component is considered significant if $r_k \geq 0.02$, and an object is retained only if it contains exactly one significant component.

Since the criterion is defined in terms of sampled surface mass rather than mesh volume or ground-truth object count, we use it as a conservative proxy for single-object content rather than an exact multi-object detector.

B.3 Geometric Deduplication

We next remove geometric near-duplicates from the quality-filtered corpus. Exhaustively comparing millions of dense point clouds is computationally prohibitive, so we adopt a two-stage procedure that combines descriptor-based candidate retrieval with stricter point-level geometric verification.

For each object, we deterministically subsample 4096 XYZ points and construct 65536 random non-self point pairs. Their Euclidean distances are accumulated into a 128-bin, L1-normalized D2 shape histogram over $[0, \sqrt{3}]$. The D2 descriptor is used only for high-recall candidate retrieval and does not directly determine whether two assets are duplicates.

We build a FAISS IVF-Flat index with 4096 inverted lists and use nprobe = 32 to retrieve the top-20 descriptor neighbors for each object. After removing self-matches and merging reciprocal candidates, this procedure yields approximately 35.5M unique candidate pairs.

For each candidate pair (P, Q), we compute bidirectional nearest-neighbor distances on the corresponding 4096 point sets:

$$d_{P\to Q}(p) = \min_{q\in Q} |p - q|2, \qquad d_{Q\to P}(q) = \min_{p\in P} |q - p|_2. \tag{3}$$

A pair is considered a duplicate only if more than 95% of points in both directions lie within 0.03 normalized units and the 95th-percentile nearest-neighbor distance in both directions is also below 0.03. The distance threshold is defined after normalizing the longest bounding-box extent to one. We do not perform ICP alignment, exhaustive rotation search, or reflection matching; the procedure therefore targets evident geometric duplicates under the canonical orientations of the source assets rather than arbitrary-pose shape equivalence.

To avoid transitive over-merging, we do not take the connected-component closure of the verified duplicate graph. Instead, we apply a greedy direct-edge star assignment, such that every removed sample must have a directly verified duplicate edge to its retained representative. This prevents chains of locally similar shapes from being merged into a single group through single-linkage effects.

Overall, geometric deduplication removes $261{,}352$ assets and produces a pool of $2{,}101{,}010$ training objects. A small set of 16 objects for which valid D2 descriptors could not be constructed is excluded from candidate retrieval and conservatively retained in the resulting pool.

B.4 Distribution Rebalancing

Despite quality filtering and geometric deduplication, web-scale 3D collections remain strongly long-tailed, with common semantic and appearance regions heavily overrepresented. Since our corpus does not provide reliable ground-truth category annotations, we rebalance the dataset in a pretrained feature space rather than enforcing class-balanced sampling, following Vo et al. (2024).

Specifically, we extract representations using a pretrained OpenShape PointBERT model. For each object, we deterministically sample 10000 XYZ-RGB points, center the XYZ coordinates, normalize them by the maximum radial distance, and clip RGB values to $[0, 1]$. OpenShape produces a 1280-dimensional representation, which we further perform L2-normalize. As OpenShape representations jointly encode geometry, appearance, and semantic information inherited from pretraining, the resulting clusters should be interpreted as regions in a pretrained feature space rather than purely geometric categories.

We construct a three-level hierarchy over all $2{,}101{,}010$ deduplicated objects using FAISS k-means. The object embeddings are first partitioned into $100{,}000$ fine clusters. Their centroids are subsequently clustered into $10{,}000$ middle-level clusters, whose centroids are further grouped into $1{,}000$ coarse clusters.

We manually set the final training budget to 1.5M objects and recursively allocate capped, approximately uniform quotas throughout the hierarchy. Allocation is performed from coarse to middle and finally fine clusters, after which samples are drawn without replacement within each fine cluster.

This procedure primarily suppresses highly populated regions of the pretrained feature space while retaining low-density regions whenever capacity permits. At the coarse level, the cluster-size Gini coefficient decreases from 0.479 to 0.346, while the maximum cluster size is reduced from $12{,}724$ to $2{,}433$. The resulting subset contains exactly $1{,}500{,}000$ unique training objects. We emphasize that this procedure rebalances OpenShape feature-space density and does not assume a one-to-one correspondence between individual clusters and ground-truth semantic categories.

C Evaluation Protocols

C.1 Classification Evaluation Settings

For classification evaluation, we use a fixed input resolution for each dataset. For ModelNet40 (Wu et al., 2015), we sample 10,000 points from each shape, while for ScanObjectNN (Uy et al., 2019) PB-T50-RS we directly use the 2,048 input points provided by the dataset. Since different pretrained models expose different types of global representations, we refer their official evaluation code to extract feature for classification. For our ShapeDINO, we concatenate the CLS tokens from 4 encoder layers: layers 3/6/9/12 for ViT-Tiny/S/B, 6/12/18/24 for ViT-L, and 10/20/30/40 for ViT-G.

For KNN evaluation, we directly use the frozen object-level representations and retrieve nearest neighbors based on cosine similarity, following DINOv2 (Oquab et al., 2023). Predictions are obtained using unweighted majority voting without temperature scaling or learnable parameters, and we set $k = 20$ for all models. For linear probing, we train a single linear classifier on top of the frozen representation using cross-entropy loss and AdamW. We use a batch size of 256, a learning rate of 1e-3, and train the classifier for 300 epochs.

We report overall accuracy (allAcc) and mean class accuracy (mAcc). OA measures the fraction of correctly classified samples over the entire test set, while mAcc first computes the classification accuracy independently for each category and then averages over all evaluated categories.

C.2 Segmentation Evaluation Settings

For ShapeNetPart (Yi et al., 2016), we deterministically sample 2,048 points from each shape for evaluation. For PartNetE (Mo et al., 2019), the original point clouds reconstructed from depth projections are highly non-uniform; we therefore preprocess each object using farthest point sampling (FPS) to obtain 16,384 points for evaluation. To evaluate local representations, we concatenate patch tokens extracted from multiple backbone layers, following each method's offical evaluation settings. we concatenate the patch tokens from 4 encoder layers: layers 3/6/9/12 for ViT-Tiny/S/B,

Table 9: Full results of classification performance on ModelNet40 and ScanObjectNN, measured by mAcc / allAcc (%) under kNN and linear probing. Bold and underlined values denote the best and second-best results within each supervision category, respectively.

Method	Arch	#Params	#Data	ModelNet40		ScanObjectNN	
				kNN mAcc / allAcc	Linear Probe mAcc / allAcc	kNN mAcc / allAcc	Linear Probe mAcc / allAcc
Weakly supervised							
OpenShape-PB	ViT	32.3M	876K	**89.3 / 92.2**	**90.0 / 90.5**	46.6 / 53.3	56.6 / 57.5
OpenShape-SC	SpConv	41.3M	876K	<u>89.1</u> / <u>91.7</u>	<u>89.4</u> / 89.3	46.1 / 50.3	57.1 / 58.0
Uni3D-Tiny	ViT-Tiny	1.02B	876K	81.9 / 87.0	87.2 / **90.5**	**75.6** / <u>77.5</u>	**80.4 / 81.3**
Uni3D-S	ViT-S	1.02B	876K	83.8 / 88.2	87.7 / 89.6	73.1 / 76.0	79.8 / <u>80.7</u>
Uni3D-B	ViT-B	1.02B	876K	82.4 / 87.3	87.7 / <u>90.1</u>	75.4 / 77.1	79.4 / 80.4
Uni3D-L	ViT-L	1.02B	876K	83.6 / 88.5	86.7 / 89.9	73.4 / 76.8	78.3 / 80.2
Uni3D-G	ViT-G	1.02B	876K	84.1 / 88.6	87.3 / <u>90.1</u>	<u>75.5</u> / **78.6**	<u>80.0</u> / **81.3**
Self-supervised							
Point-MAE	ViT	21.8M	51K	78.7 / 85.5	87.8 / 90.4	46.6 / 53.9	68.1 / 70.6
Point-M2AE	HierViT	12.7M	51K	<u>81.4</u> / <u>87.2</u>	88.2 / 91.5	52.1 / 58.8	64.0 / 67.1
Sonata	PTv3	108.5M	140K	59.6 / 69.3	78.6 / 83.0	27.9 / 35.6	40.2 / 48.1
Utonia	PTv3	137.3M	1.2M	80.7 / 84.2	88.6 / 90.2	62.5 / 66.1	75.3 / 77.1
ShapeDINO-Tiny	ViT-Tiny	6.9M	1.5M	80.0 / 85.3	89.6 / 91.5	67.8 / 70.7	79.3 / 80.3
ShapeDINO-S	ViT-S	6.9M	1.5M	80.4 / 85.7	90.6 / 92.1	70.4 / 72.8	80.1 / 81.6
ShapeDINO-B	ViT-B	86.9M	1.5M	<u>81.4</u> / 86.5	91.1 / 92.5	71.0 / 73.6	81.4 / 82.6
ShapeDINO-L	ViT-L	86.9M	1.5M	79.8 / 85.9	**91.4 / 92.9**	<u>72.8</u> / <u>75.3</u>	<u>83.7</u> / <u>84.9</u>
ShapeDINO-G	ViT-G	1.14B	1.5M	**83.0 / 88.0**	<u>91.3</u> / <u>92.8</u>	**75.4 / 77.0**	**84.1 / 85.1**

6/12/18/24 for ViT-L, and 10/20/30/40 for ViT-G. For patch-based models, the resulting token features are interpolated to the input point resolution using 3NN interpolation before being fed into a point-wise linear classifier. For architectures that provide native point-level feature mappings, such as OpenShape-SparseConv (Liu et al., 2023) and Utonia (Zhang et al., 2026b), we instead use their native inverse mapping to recover features at the input-point resolution.

We evaluate all models using linear probing with a single point-wise linear classifier while keeping the pretrained backbone frozen. ShapeNetPart and PartNetE use object batch sizes of 32 and 8, respectively. For both datasets, the linear probe is optimized using AdamW with a learning rate of 1e-3 and zero weight decay for 100 epochs.

For ShapeNetPart, we report class mIoU and instance mIoU. Instance mIoU first computes the mean IoU over valid parts for each shape and then averages over all test shapes, while class mIoU first averages shape-level IoUs within each object category and subsequently averages across categories. For PartNetE, we report part mIoU, where IoU is computed for each semantic part over objects containing that part and then macro-averaged across all valid parts.

D Additional Quantitative Results

We provide the complete quantitative results for classification and part segmentation, complementing the condensed comparisons presented in the main paper. All results follow the evaluation protocols described in Sec. C. Weakly supervised methods are reported separately from self-supervised methods due to their additional semantic supervision.

D.1 Full Classification Comparison

Table 9 reports the full classification results on ModelNet40 (Wu et al., 2015) and ScanObjectNN (Uy et al., 2019) under both kNN and linear probing, including mAcc and allAcc. ShapeDINO consistently outperforms prior self-supervised methods across the two datasets and evaluation protocols. On ScanObjectNN, ShapeDINO-G achieves 75.4/77.0 mAcc/allAcc with kNN and 84.1/85.1 with linear probing. On ModelNet40, ShapeDINO-G reaches 83.0/88.0 under kNN, while ShapeDINO-L obtains the highest linear-probing performance of 91.4/92.9. The improvements are consistent across both accuracy metrics, demonstrating the high quality of the learned object-level representations.

Table 10: Full results of segmentation performance on ShapeNetPart and PartNetE under linear probing. We report class mIoU / instance mIoU for ShapeNetPart, and part mIoU for PartNetE. Bold and underlined values denote the best and second-best results within each supervision category, respectively.

Method	Arch	#Params	#Data	ShapeNetPart		PartNetE
				Linear Probe		Linear Probe
				Class mIoU	Instance mIoU	Part mIoU
Weakly supervised						
OpenShape-PB	ViT	31.7M	876K	78.3	79.8	41.0
OpenShape-SC	SpConv	41.3M	876K	74.8	75.7	38.2
Uni3D-Tiny	ViT-Tiny	6.4M	876K	79.3	81.8	46.4
Uni3D-S	ViT-S	22.8M	876K	78.9	81.1	47.9
Uni3D-B	ViT-B	89.0M	876K	79.8	81.4	47.4
Uni3D-L	ViT-L	307.4M	876K	**80.9**	<u>82.3</u>	<u>49.5</u>
Uni3D-G	ViT-G	1.02B	876K	<u>80.8</u>	**82.5**	**50.2**
Self-supervised						
Point-MAE	ViT	29.0M	51K	80.6	82.3	44.1
Point-M2AE	HierViT	15.3M	51K	79.7	81.3	44.9
Sonata	PTv3	108.5M	140K	76.1	78.1	32.1
Utonia	PTv3	137.3M	1.2M	80.9	82.8	45.7
ShapeDINO-Tiny	ViT-Tiny	6.9M	1.5M	82.1	83.2	48.9
ShapeDINO-S	ViT-S	22.9M	1.5M	82.4	84.0	49.8
ShapeDINO-B	ViT-B	86.9M	1.5M	<u>82.9</u>	84.1	50.5
ShapeDINO-L	ViT-L	384.6M	1.5M	82.8	<u>84.3</u>	<u>50.9</u>
ShapeDINO-G	ViT-G	1.14B	1.5M	**83.4**	**84.5**	**51.3**

D.2 Full Segmentation Comparison

Table 10 presents the full linear-probing results on ShapeNetPart (Yi et al., 2016) and PartNetE (Mo et al., 2019). All ShapeDINO variants improve over previous self-supervised methods across class mIoU, instance mIoU, and part mIoU. ShapeDINO-G achieves 83.4 class mIoU and 84.5 instance mIoU on ShapeNetPart, together with 51.3 part mIoU on PartNetE. These results show that the learned representations transfer effectively not only to object-level recognition but also to dense part-level prediction.

E Additional Qualitative Results

We provide additional qualitative analyses of the dense representations learned by ShapeDINO. Figure 6 projects point-wise features onto the first three principal components and maps them to RGB values. Across diverse ShapeNet55 categories, the features exhibit spatially coherent patterns while maintaining clear variations across different geometric regions, indicating a structured organization of local 3D features.

Figure 7 further visualizes feature-based keypoint correspondence. We select query points on the first shape and transfer them to subsequent shapes via cosine nearest-neighbor matching in the learned feature space. The correspondences remain largely consistent under substantial pose and shape variations, showing that ShapeDINO captures stable local structures across different observations and instances.

F Ablation Details and Additional Experiments

Ablation Experiments Details For all ablation experiments, we train ShapeDINO-Small from scratch using 16 GPUs with a batch size of 40, while keeping the remaining training and evaluation settings unchanged unless otherwise specified.

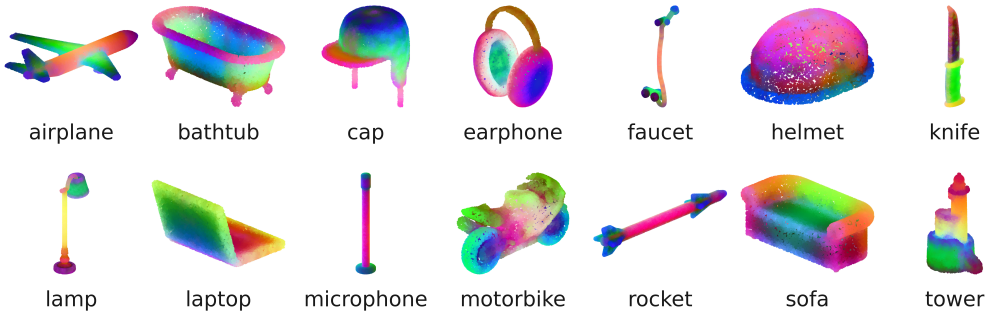

Figure 6: PCA visualization of ShapeDINO features on ShapeNet55. The first three principal components of point-wise features are mapped to RGB.

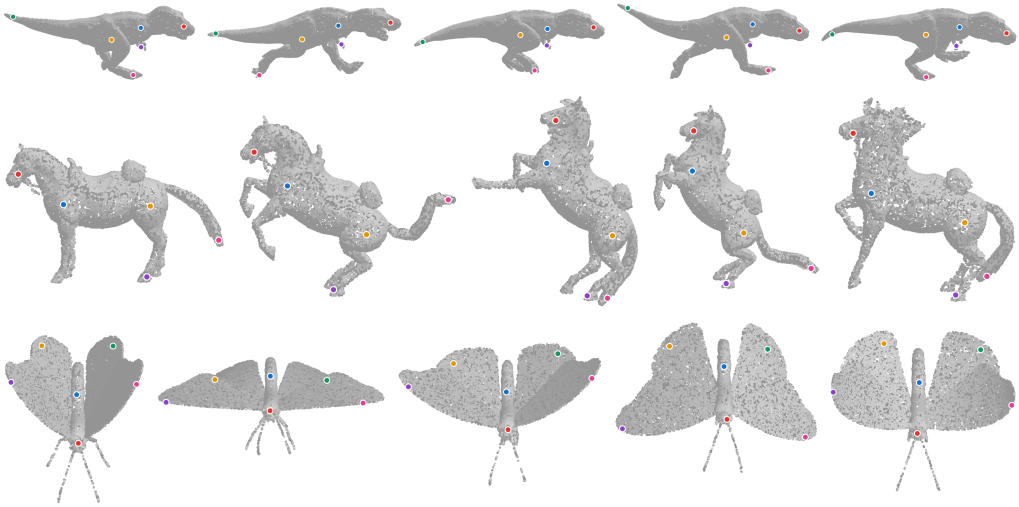

Figure 7: Selected keypoints in the first frame are matched to each subsequent frame by cosine nearest-neighbor on ShapeDINO representation. Matching colors indicate corresponding points.

Additional Ablations We further investigate the effects of multi-crop construction and training objectives. Reducing the number of local views to 4 only slightly affects ScanObjectNN kNN accuracy, but leads to a larger drop on PartNetE. The local crop design has a more pronounced impact: using larger local crops substantially degrades both metrics, and removing crop-specific augmentation results in the largest performance drop. Finally, removing the iBOT objective slightly improves object-level kNN accuracy but noticeably decreases PartNetE p.mIoU, indicating that patch-level supervision is particularly beneficial for learning transferable dense representations.

G Details on Applications

Here we provide the details of our application experiments.

G.1 Representation Alignment on 3D Generation Model

Implementation Details We implement an image-conditioned rectified-flow generation model based on CraftsMan3D (Li et al., 2025a) framework and use a pretrained frozen TripoSG (Li et al., 2025b) VAE as the shape autoencoder. We train the generation model on TRELLIS500K (Xiang et al., 2025) dataset. We perform dataset preprocess, watertight, point sampling, and condition images rendering for the whole dataset following (Zhang et al., 2024). For each shape, we sample 16,384 surface points with normals and encode them into 1,024 latent tokens of dimension 64. The

Table 11: Additional ablations on multi-crop construction and training objectives. All our abltation are conducted on training ShapeDINO-S from scratch.

Variant	ScanObjectNN kNN ↑	PartNetE p.mIoU ↑
Ours	<u>61.2</u>	**43.4**
4 Local Views	61.0	<u>41.0</u>
Larger Local Crop Size	51.5	36.6
w/o Crop Aug.	43.7	22.0
w/o iBOT	**62.2**	39.4

DiT is based on CraftsMan3D (Li et al., 2025a), which is a PixArt-style DiT with 32 Transformer blocks, a hidden dimension of 1,024, and 16 attention heads. Image conditions are extracted using CLIP ViT-L/14 (Radford et al., 2021) and DINOv2-L (Oquab et al., 2023) at a resolution of 518×518. Both image backbones are frozen, while the linear projection applied to the DINOv2 features remains trainable. We use a condition dropout probability of 0.1 for classifier-free guidance.

Representation Alignment. Following REPA (Yu et al., 2024), we incorporate our learned 3D representations into the generator by supervising intermediate DiT tokens with a frozen ShapeDINO ViT-L EMA teacher. To establish explicit spatial correspondence between the vae latents and the 3D representations, the TripoSG VAE and ShapeDINO share the same 1,024 sampled query indices during offline preprocessing. For each query center, the teacher constructs a local patch using 64 nearest-neighbor points and extracts the normalized patch token from its final layer. This produces a one-to-one correspondence between the VAE latent tokens and the teacher features without requiring any post-hoc token matching. Both the VAE posterior and the teacher features are precomputed and cached, avoiding additional VAE encoding or teacher inference during generator training. We set the weight of representation alignment loss to 0.5, with the final loss of:

$$\mathcal{L} = \mathcal{L}_{diffusion} + 0.5 * \mathcal{L}_{REPA} \tag{4}$$

Training and Inference. We train the DiT with AdamW using a learning rate of 1e-4, and zero weight decay. We use a 20K-step warmup followed by a constant learning rate. Training is performed on 32 GPUs with a per-GPU batch size of 40, resulting in a global batch size of 1,280, using BF16 mixed precision and DeepSpeed ZeRO Stage 2. Timesteps are sampled using a logit-normal distribution with a FlowMatch Euler scheduler with shift 3.0. At inference time, we use 30 Euler sampling steps with a classifier-free guidance scale of 7.5. Meshes are extracted using TripoSG hierarchical geometry extraction, progressively refining the field resolution from 256^3 to 512^3.

Evaluation. We evaluate different training checkpoints on a fixed set of 200 objects using identical conditioning images and sampling seeds across methods. We measure semantic consistency using Uni3D image shape similarity. Specifically, the conditioning image is encoded by EVA02-E-14-plus, while the generated shape is encoded by Uni3D-g using 10,000 surface points, and we report the cosine similarity between their normalized global embeddings. We additionally evaluate local geometric consistency using rendered-normal similarity proposed by ViLearn (Chen et al., 2026). We first estimate normals from the condition image using MoGe2 (Wang et al., 2026) and render normal maps of the generated mesh from 512 viewpoints. DINOv2-L (Oquab et al., 2023) patch features are extracted from both normal maps, and cosine similarity is computed over valid foreground patches. We report the maximum mean similarity across all rendered viewpoints.

G.2 Promptable 3D Part Segmentation

Model Architecture. We build our promptable 3D part segmentation model upon Point-SAM, replacing its original point-cloud encoder with our pretrained ShapeDINO backbone. We use the ViT-L configuration with 24 Transformer blocks, a feature dimension of 1,152, and 12 attention heads. Each input point cloud contains 10,000 points and is grouped into 512 patches using FPS, with 64 nearest neighbors for each patch center.

The resulting 1,152-dimensional patch features are projected to 256 dimensions through a learnable linear layer and fed into the original Point-SAM prompt encoder and mask decoder. The mask

decoder uses a two-layer Two-Way Transformer with 256-dimensional features, 8 attention heads, and an MLP dimension of 2,048. The encoder, feature projection, and segmentation modules are jointly fine-tuned rather than using the pretrained representation as a frozen feature extractor.

Training. For each training sample, we randomly select two ground-truth part masks and perform six iterative prompting and refinement rounds. These iterations include both prompt-based updates and mask refinement, and therefore do not correspond to a fixed number of point clicks. The segmentation objective combines sigmoid focal loss, Dice loss, and an IoU prediction regression loss, with the Dice loss weighted by 2. We train on the combined PartNet (Mo et al., 2019)/ShapeNet (Chang et al., 2015) and PartNet-Mobility (Xiang et al., 2020) training sets and use the PartNet/ShapeNet validation configuration for evaluation during training. We use AdamW with an initial learning rate of 3e-4 and weight decay of 0.1. The learning rate is linearly warmed up for the first 3,000 optimizer updates and decayed by a factor of 0.1 at 30K and 40K updates. Training runs for 50K optimizer updates with a batch size of 16 per GPU on 8 NVIDIA A800-SXM4-80GB GPUs, resulting in a global batch size of 128. We use BF16 mixed precision and gradient checkpointing. The recorded end-to-end training time is approximately 16.3 hours, including validation, checkpointing, and logging overhead.

G.3 Point Cloud Completion

Model Details We adopt PoinTr as the downstream completion framework and replace its original DGCNN and Transformer encoder with our pretrained ShapeDINO encoder, while retaining the query generator, eight-layer Transformer decoder, and folding-based completion head. We use the ShapeDINO-S configuration with 12 Transformer blocks, a feature dimension of 384, and 8 attention heads.

Training Details Each partial input contains 2,048 points and is grouped into 128 patches with 32 points per patch. Missing normal and color channels are zero-filled. Before encoding, the partial point cloud is centered and normalized by its maximum radius. The corresponding patch centers are subsequently transformed back to the original completion coordinate system before being passed to the decoder. A learnable linear projection adapts the encoder patch features to the PoinTr decoder, and all components are jointly fine-tuned end-to-end.

Following the inclusive epoch loop of the implementation, we train for 300 epochs on PCN and 200 epochs on ShapeNet55 and ShapeNet34. The final completed point clouds contain 16,384 points for PCN and 8,192 points for ShapeNet55/34, including the partial input.

H Limitations

Our study has several limitations. First, we primarily focus on object-centric 3D point clouds, with the model design, pretraining data, and evaluations centered on individual objects. It remains unclear how well the same training paradigm extends to large-scale indoor or outdoor scenes involving clutter, multiple interacting objects, and substantially larger spatial extents. Second, although our pretrained representations transfer well to real scanned data, the pretraining distribution does not fully capture the diversity of real-world sensing conditions, such as severe occlusion, sensor noise, non-uniform sampling, and incomplete observations. Robustness under such conditions therefore remains underexplored. Finally, despite our efficient implementation and knowledge distillation, large-scale 3D pretraining still requires substantial computation due to point sampling, neighborhood construction, and large Transformer backbones. Improving the data and computational efficiency of large-scale 3D representation learning remains an important direction for future work.