Point-MaDi: Masked Autoencoding with Diffusion for Point Cloud Pre-training

Xiaoyang Xiao[1], **Runzhao Yao**[1], **Zhiqiang Tian**[2], **Shaoyi Du**[1*]
[1] State Key Laboratory of Human-Machine Hybrid Augmented Intelligence,
National Engineering Research Center for Visual Information and Applications,
and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University
[2] School of Software Engineering, Xi'an Jiaotong University
lopeture@stu.xjtu.edu.cn, rzy3320@163.com
zhiqiangtian@xjtu.edu.cn, dushaoyi@xjtu.edu.cn

Abstract

Self-supervised pre-training is essential for 3D point cloud representation learning, as annotating their irregular, topology-free structures is costly and labor-intensive. Masked autoencoders (MAEs) offer a promising framework but rely on explicit positional embeddings, such as patch center coordinates, which leak geometric information and limit data-driven structural learning. In this work, we propose **Point-MaDi**, a novel **Point** cloud **Ma**sked autoencoding **Di**ffusion framework for pre-training that integrates a dual-diffusion pretext task into an MAE architecture to address this issue. Specifically, we introduce a center diffusion mechanism in the encoder, noising and predicting the coordinates of both visible and masked patch centers without ground-truth positional embeddings. These predicted centers are processed using a transformer with self-attention and cross-attention to capture intra- and inter-patch relationships. In the decoder, we design a conditional patch diffusion process, guided by the encoder's latent features and predicted centers to reconstruct masked patches directly from noise. This dual-diffusion design drives comprehensive global semantic and local geometric representations during pre-training, eliminating external geometric priors. Extensive experiments on ScanObjectNN, ModelNet40, ShapeNetPart, S3DIS, and ScanNet demonstrate that Point-MaDi achieves superior performance across downstream tasks, surpassing Point-MAE by 5.50% on OBJ-BG, 5.17% on OBJ-ONLY, and 4.34% on PB-T50-RS for 3D object classification on the ScanObjectNN dataset. Codes are available at https://github.com/YangParky/Point-MaDi.

1 Introduction

Driven by advances in LiDAR and depth-sensing technologies, point clouds have become a fundamental data representation in applications such as autonomous driving, robotics, and virtual reality, owing to their ability to capture fine-grained geometric details of objects and environments. More recently, supervised learning has significantly advanced 3D computer vision by introducing 3D-centric methods that directly operate on raw point clouds for tasks such as object classification [34, 35, 51, 11], semantic segmentation [34, 25], and object detection [33, 26]. These approaches typically rely on large-scale annotated datasets to achieve high performance. However, unlike 2D images arranged in regular grids, point clouds lack a consistent topology, making the annotation process both expensive and labor-intensive. Labeling 3D data [4, 56, 2, 6, 60, 47] often requires expert knowledge to accurately capture complex geometrical structures, which limits the scalability and generalization ability of supervised approaches. Self-supervised learning (SSL) [29, 7, 54, 14] has emerged as a promising

*Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

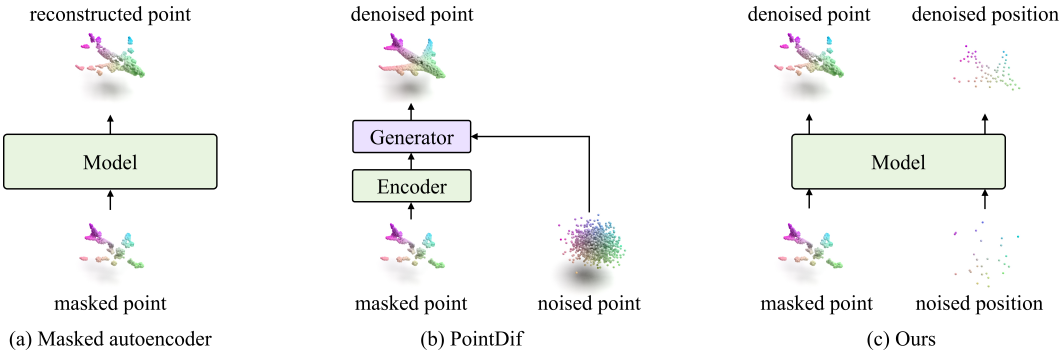


Figure 1: Comparison between different pretext tasks. (a) Masked autoencoders reconstruct masked point patches. (b) PointDif uses a conditional point generator to guide the point-to-point generation from noisy input. (c) Our Point-MaDi denoises noisy masked patches and reconstruct their centers.

alternative, enabling the extraction of generalizable representations from unlabeled point clouds through the design of various pretext tasks, including generative- and contrastive-based objectives.

Among them, diffusion probabilistic models (DPMs) [15, 28, 32] have recently emerged as a powerful paradigm for 3D point cloud representation learning, owing to their ability to model complex data distributions through iterative denoising processes. Unlike contrastive learning [57, 69, 1, 61] aligns views of a point cloud via maximizing the similarities of positive pairs to capture global semantic consistency, or reconstruction-based methods [31, 65, 9, 13, 67, 36, 37] like masked autoencoders (MAEs) [14, 31], which mask patches and reconstruct geometry using an encoder-decoder architecture. DPMs generate data step-wise from Gaussian noise and learn to reverse this process, potentially capturing rich semantic and geometric structures across scales. Despite their strengths, existing diffusion-based methods mainly rely on global context aggregation or predefined conditioning mechanisms, such as class labels or auxiliary features, to guide the denoising process. Recent studies [70, 19] have begun to address these challenges by integrating diffusion frameworks into MAEs; this structure naturally complements diffusion models: the encoder can operate on partially observed data, while the decoder can progressively recover masked content from latent noise, aligning well with the denoising objective of DPMs. Nonetheless, directly combining MAE and diffusion remains nontrivial, as current MAEs inject geometric priors, such as patch center embeddings, that leak explicit positional information into the encoder, hindering the objective of learning structure purely from data-driven cues [42, 49, 3]. This motivates our central question: **How can we design a diffusion-based pre-training framework that mitigates geometric information leakage while enhancing the modeling of local geometric structures for faithful reconstruction?**

Intuitively, if positional embeddings are particularly provided or noised, the model is forced to infer global spatial relationships solely from the visual and contextual content of the point cloud, encouraging a deeper understanding of its structure and semantics. Considering this, we propose **Point-MaDi**, a novel **Point** cloud **Ma**sked autoencoding **Di**ffusion framework. It introduces a dual-diffusion pretext task that applies diffusion to both centers and local patches, predicting centers and reconstructing patches to enforce robust modeling of comprehensive representations without external geometric cues. Fig. 1 illustrates how Point-MaDi contrasts with existing pretext tasks: while MAEs reconstruct masked patches using provided positional embeddings, PointDif advances this paradigm by formulating pre-training as a conditional point-to-point generation task. Our proposed Point-MaDi further instantiates this by performing center denoising in the encoder and patch reconstruction in the decoder, enabling geometry-aware self-supervised learning without positional cues.

To this end, we first group the point cloud into patches and apply random masking strategies to create visible and masked regions, as in traditional MAEs. In the encoder, a center diffusion process applies noise to visible and masked patch centers and tasks the model with denoising them, eliminating reliance on ground-truth positional embeddings. This process, implemented via iterative sampling, forces the encoder to model global spatial relationships by inferring center positions from partial observations. Visible patches undergo self-attention to capture intra-patch relationships, while masked patches leverage cross-attention with visible patches to model inter-patch dependencies, refining the predicted centers. In the decoder, a patch diffusion process reconstructs masked patches from latent noise, conditioned on the encoder's latent representations of visible patches and the predicted centers.

This reconstruction is optimized using Chamfer Distance, ensuring high-fidelity recovery of local structures, particularly in sparse point clouds. By integrating center diffusion for global modeling and patch diffusion for local reconstruction, Point-MaDi encourages the encoder to learn robust, context-aware representations while enabling the decoder to focus on fine-grained geometric details.

Our main contributions of this work are as follows:

- We propose Point-MaDi, a novel self-supervised pre-training framework for point clouds that disentangles positional encoding and geometry modeling via two dedicated diffusion processes, effectively mitigating positional shortcut leakage.
- The center diffusion process adds noise to both visible and masked patch centers and predicts clean coordinates, replacing explicit positional embeddings and promoting high-level spatial understanding; the patch diffusion process, conditioned on visible features and predicted centers, reconstructs noisy masked patches via step-wise denoising to recover local geometry.
- Extensive experiments on ScanObjectNN, ModelNet40, ShapeNet, S3DIS, and ScanNet demonstrate that Point-MaDi significantly outperforms existing methods in classification, segmentation, and detection tasks, validating its effectiveness and generalizability.

2 Related Work

Self-supervised point cloud representation learning. Self-supervised Learning (SSL) has achieved remarkable success in many fields such as NLP and computer vision. It aims to learn useful representations from the massive unlabeled data by defining a pre-text task [29, 17, 54] through image/patch operations, with no external labels. Recent methods have explored the potential and strengths of SSL in the point cloud domain. A prominent approach in SSL [57, 69] is contrastive learning, which encourages representations of augmented versions of an instance to be more similar compared to different inputs. PointContrast [57] pioneers contrastive learning by performing point-level invariant mapping learning. Several recent works integrate point cloud representations with other modalities, such as vision [1, 20, 46, 61] and language [66, 63, 59, 23] to facilitate the learning of transferable 3D point cloud representations. CrossPoint [1] combines intra-modal and cross-modal contrastive learning, enforcing invariance to point cloud augmentations and aligning 2D image features with point cloud prototypes. More recently, masked prediction [14] has re-attracted attention trying to recover the original input from a masked version since the introduction of the Vision Transformer (ViT) [10]. Point-Bert [62] extends the BERT-style [7] pre-training strategy to point cloud transformers via discrete Variational Autoencoder [39]. Point-MAE [31] brings the masked autoencoder idea to point cloud tasks by randomly masking input point patches and reconstructing them. Afterward, PointM2AE [65] and I2P-MAE [67] adopt hierarchical MAE transformers to extract fine-grained and higher-level semantic features of 3D shapes. Subsequent works [9] mainly enrich the model's comprehension of point cloud geometric structures using cross-modal knowledge. For instance, ACT [9] utilized a pre-trained ViT as a teacher model to acquire knowledge from other modalities. Joint-MAE [13] proposes a 2D-3D joint MAE framework to reconstruct the masked two modalities. ReCon++ [37] is similar to ReCon [36], both employing a generative framework (MAE-based) and incorporating contrastive learning through ensemble distillation. Our Point-MaDi leverages masked autoencoding with diffusion, and by progressively denoising noised positional centers, it reduces reliance on positional shortcuts while enhancing geometric awareness.

Diffusion probabilistic models. Diffusion probabilistic models [15, 8], also known as score-based models [43, 44], have gained significant attention in computer vision for their ability to generate high-fidelity images. DPMs begin by using an evolving Stochastic Differential Equation (SDE) to gradually add Gaussian noise to real data, turning complex data into a Gaussian distribution. Then, a time-reversed SDE maps Gaussian noise back into high-quality samples, guided by a network using the score function [45] with multiple steps. It has demonstrated superior performance in various generative fields, including image generation [28, 40, 32, 71, 18, 64], video generation [41, 12], and speech generation [21]. However, when adopted in the 3D domain, a number of works focus on 3D generation [24, 58, 27, 30, 55, 19]. Applying DPMs to 3D point cloud pre-training remains underexplored due to the challenges posed by the irregular sampling patterns of point clouds in 3D space. [24] proposed an autoencoder architecture with a DPM as a decoder. DiffPMAE [19] introduces a two-stage architecture that denoises the masked regions conditioned on the latent representation from the first stage. The most related work to ours is PointDif [70], which aggregates latent features via a condition aggregation network to guide iterative denoising of noisy point

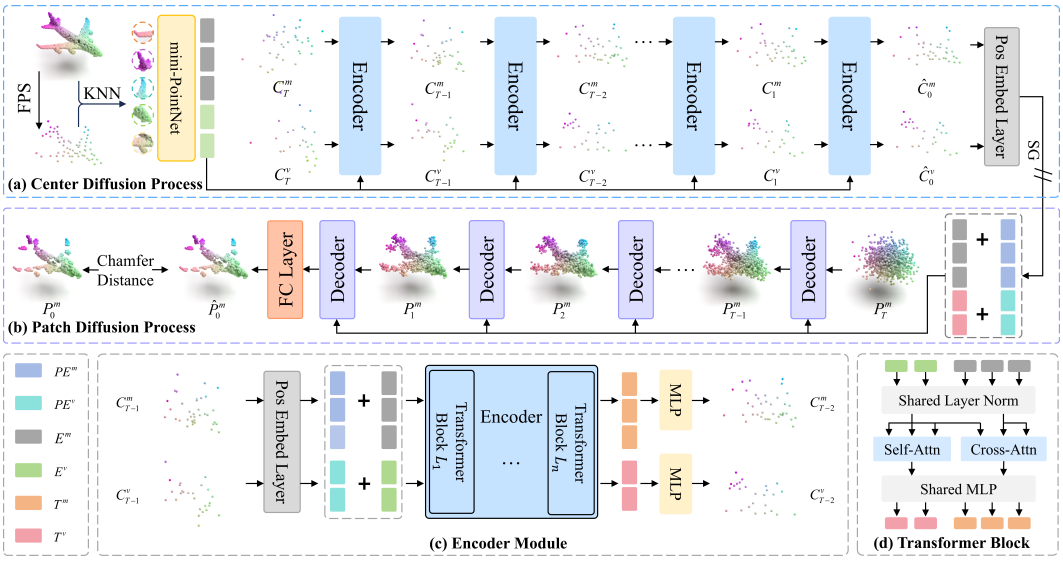


Figure 2: The pipeline of our Point-MaDi framework. The encoder adopts a center diffusion process, where noise is added to the centers of both visible and masked patches. A multilayer perceptron (MLP) maps the noisy centers to positional embeddings, and both these and visible patch embeddings are input to a transformer to predict the clean centers. The decoder performs a patch diffusion process, conditioned on the visible tokens and the predicted centers, to progressively denoise and reconstruct the masked patches. The chamfer distance guides the patch reconstruction, while the stop gradient (SG) operation prevents leakage of ground-truth masked patch positions during pre-training.

clouds for pre-training. However, our Point-MaDi differs in several key aspects. First, unlike PointDif's holistic point-to-point denoising, we introduce a joint diffusion process that simultaneously models patch centers and coordinates, capturing both structural and local geometric priors for enhanced robustness. Second, instead of relying on complex modules like PointDif's separate encoder, aggregator, and diffusion model, we unify the point encoding and decoding process with a bi-jective network design, improving efficiency and reconstruction fidelity.

3 Point-MaDi

3.1 Overview of Point-MaDi

The proposed framework contains two modules, each driven by distinct diffusion models for specific tasks. Fig. 2 illustrates the end-to-end diffusion process of Point-MaDi. For a given clean point cloud, the encoder module partitions the input into g patches and trains a diffusion model that iteratively adds noise to the visible and masked patch centers. Then a denoising process predicts clean centers for all patches. The decoder module, conditioned on the encoder's visible tokens and predicted patch centers, aims to reconstruct the noisy masked patches supervised by Chamfer Distance. This dual-diffusion design enables Point-MaDi to learn both sparse structural and dense geometric representations.

3.2 Point cloud processing

Point patches generation. Following previous works [62, 31], we divide point clouds into overlapping point patches via Farthest Point Sampling (FPS) and K-Nearest Neighbors (KNN) algorithm. Formally, given the input point cloud $X \in \mathbb{R}^{n\times 3}$ with three-dimensional (x, y, z coordinates) n number of points, we first choose the centers $C \in \mathbb{R}^{g\times 3}$ for g number of groups through FPS. For centers C, we adopt the KNN to select k nearest points and obtain local geometric groups $P \in \mathbb{R}^{g\times k\times 3}$.

$$C = \mathrm{FPS}(X), \quad P = \mathrm{KNN}(X, C), \tag{1}$$

To eliminate global location bias, each patch in P is normalized by subtracting its corresponding center coordinates. The resulting centered patches are treated as sub-clouds and served as a sequence of localized point representations treated like words in NLP or image patches in vision.

Patch masking. For the point patches, we select a predefined mask ratio $m \in (0, 1)$ and deploy random masking, outputting the visible point patches $P^v \in \mathbb{R}^{(g-r)\times k\times 3}$ and masked point patches

$P^m \in \mathbb{R}^{r\times k\times 3}$, where $r = \lfloor g \cdot m \rfloor$ is the number of masked patches, $g - r$ is the number of visible patches, and $\lfloor \cdot \rfloor$ denotes the floor function. The corresponding centers are denoted as $C^v \in \mathbb{R}^{(g-r)\times 3}$ and $C^m \in \mathbb{R}^{r\times 3}$, respectively. We also obtain a binary indicator of whether the patch is masked.

Embedding. The visible point patches P^v are embedded through a simplified PointNet which employs 1×1 convolutions and max pooling to generate the visible tokens E^v.

$$E^v = \text{PointNet}(P^v), \quad E^v \in \mathbb{R}^{(g-r)\times d}. \tag{2}$$

where d is the hidden dimension of the network. To incorporate spatial information, we compute positional embeddings for both visible and masked patches by applying a shared-weight MLP with GELU activation to their respective center points:

$$PE^v = \text{MLP}(C^v), \quad PE^m = \text{MLP}(C^m). \tag{3}$$

The position embeddings PE^v and PE^m are added to every transformer block. For the encoder, these positional embeddings are fixed, derived directly from the input patch centers to provide stable spatial cues during the masking and diffusion process. In contrast, the decoder leverages predicted center positions estimated from the encoder's output. Note that we use the same MLP for the encoder and decoder in our autoencoder to compute positional embeddings.

3.3 Center diffusion process

After obtaining visible and masked point patches E^v, E^m. We adopt a 12-layer standard transformer encoder as the diffusion backbone, with each block containing a multi-head self-attention (MSA) layer and a feed-forward network (FFN). The encoder applies self-attention to E^v to extract conditioning features that guide the diffusion dynamics, and then performs cross-attention using E^m as the query to generate masked token representations conditioned on visible context.

Forward diffusion process. To model geometric corruption, we apply a forward diffusion process to the centers of visible and masked point patches. At each of the T time steps, Gaussian noise is incrementally added to C^v and C^m following a Markov chain:

$$q(C_t^v \,|\, C_{t-1}^v) = \mathcal{N}(C_t^v; \sqrt{1-\beta_t} C_{t-1}^v, \beta_t I), \tag{4}$$

where $t_c \in [1, 2, 3, ..., T]$ is the time step of center diffusion process and $\beta_t I$ is the variance of the noise at step t_c, which controls the amount of noise added at each step. Since all transition kernels of the diffusion process are Gaussian, samples from the intermediate distributions can be directly formulated in a single step by applying the reparameterization trick:

$$q(C_t^v \,|\, C_0^v) = \mathcal{N}(C_t^v; \sqrt{\bar{\alpha}_t} C_0^v, (1-\bar{\alpha}_t) I), \tag{5}$$

with $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$. Similarly, the forward process for masked centers can be sampled directly:

$$q(C_t^m \,|\, C_0^m) = \mathcal{N}(C_t^m; \sqrt{\bar{\alpha}_t} C_0^m, (1-\bar{\alpha}_t) I). \tag{6}$$

We use a linear variance schedule with β_t increasing from 0.0001 to 0.02 over 2000 steps. As time goes by, the centers gradually diffuse into a chaotic set of points. To ensure stability and consistency, we use the same variance schedule for both visible and masked centers.

Conditional reverse process. Providing strong conditioning information c is usually helpful to reduce the number of inference steps and improve the generation quality. For visible centers and corresponding tokens, we input the latent space E^v from the token layer and the corresponding noised position embedding PE_t^v to generate the conditional latent visible patches.

$$T^v = \text{Encoder}(E^v, PE^v), \quad T^m = \text{Encoder}(E^m, E^v, PE^m), \tag{7}$$

where T^v serves as the conditioning feature for visible centers, guiding the reverse transition, while T^m serves as the conditioning feature for masked centers, integrating visible and masked patch information to guide the reverse diffusion process. It is performed via self-attention and cross-attention within each transformer block, which includes both MSA and FFN layers. The self-attention and cross-attention for a single block can be formulated as:

$$Z^v = \text{SelfAttn}(Q^v, K^v, V^v), \quad Z^m = \text{CrossAttn}(Q^m, K^{m+v}, V^{m+v}), \tag{8}$$

where Q^v, K^v, V^v are the Query, Key, Value of the visible features, and Q^m, K^{m+v}, V^{m+v} are the masked query, concatenated key, and value, respectively. These attention outputs (Z^v, Z^m) are

passed through the FFN and normalization layers with shared parameters to produce the final encoder outputs T^v and T^m. The self-attention and cross-attention share the parameters of transformer encoder blocks to avoid increasing parameters.

Unlike conventional diffusion models predicting additive noise, our reverse process aims to recover the visible and masked centers C^v and C^m by gradually removing the noise under the condition of both the visible and masked representations. However, it is non-trivial to approximate $q(C^v_{t-1} \big| C^v_t, T^v_t)$ without knowing the entire diffusion process. Therefore, we train the Encoder module with transformers to learn $p_\theta(C^v_{t-1} \big| C^v_t, T^v_t)$ for approximating the conditional probabilities to infer the entire reverse diffusion process. The reverse transition for the visible centers can be formulated as:

$$C^x_{t-1} = \left(\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} C^x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} \hat{C}^x_0 \right) + \sigma_t \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I). \tag{9}$$

Here, $x \in \{v, m\}$ indicates visible or masked centers. $\hat{C}^x_0 = f_\theta(\cdot)$ denotes the predicted clean center, inferred by a shared-parameter encoder. For the visible case ($x = v$), the prediction is conditioned on C^v_t, E^v, PE^v_t, and t_c; for the masked case ($x = m$), additional inputs E^m and PE^m_t are included.

3.4 Patch diffusion process

Similar to the encoder's center diffusion process, the decoder in Point-MaDi leverages a diffusion-based approach to pre-train robust point cloud representations by denoising masked patches. However, it differs in its target and conditioning mechanism. While the encoder diffuses both visible and masked group centers (C^v, C^m), employing self-attention for visible centers and cross-attention for masked ones, the decoder exclusively denoises masked point cloud patches P^m. In the forward diffusion process, Gaussian noise is added to masked patches P^m_t over T time steps, formulated as $q(P^m_t | P^m_0) = \mathcal{N}(P^m_t; \sqrt{\bar{\alpha}_t} P^m_0, (1-\bar{\alpha}_t)I)$, where $\bar{\alpha}_t = \prod_{i=1}^{T} \alpha_i$, following the same linear variance schedule as in Sec. 3.3. The reverse process approximates the denoising transition $p_\theta(P^m_{t-1} \big| P^m_t, T^v_t, t) = \mathcal{N}(P^m_{t-1}; \mu_\theta(P^m_t, T^v_t, t), \sigma^2_t I)$, with the transformer decoder predicting clean patches. To this end, we train a diffusion model as the decoder, which conditions on the encoder's visible encoded features T^v concatenated with learnable mask tokens X^m, utilizing a transformer-based architecture with self-attention to process these integrated inputs. This design, coupled with specialized modules for token generation and reconstruction, enables the recovery of dense local geometry, complementing the encoder's sparse center denoising, with key components as follows.

Mask token layer. The mask token layer maps noisy masked patches P^m_t from the diffusion process to a latent representation $X^m_t \in \mathbb{R}^{r \times D}$, ensuring a constant number of output samples and aligning the masked and visible patches. It consists of a 1D convolutional layer processing input patches of shape $r \times k \times 3$. The model processes a point cloud with 1024 input points with a mask ratio $m = 0.6$, resulting in $\lfloor 1024 \cdot 0.6 \rfloor$ masked points. The output from the mask token layer is size $r \times d$.

Time embedding. To provide a unique embedding for each time step in the diffusion sequence, allowing the decoder transformer to learn the temporal relation and handle the time sequence. We construct a 384-dimensional frequency embedding TE_t followed by a two-layer MLP with dimensionality equal to the transformer's hidden size and SiLU activations.

Transformer decoder. The structure of the decoder contains fewer transformer blocks compared to the encoder. Empirically, the depth of the network affects its diffusion performance. Hence, we explore fine-tuning with a different number of layers. Unlike existing SSL methods such as Point-MAE [31] and ReCon [36], which rely on ground-truth positional embeddings for both visible and masked patches alongside visible tokens (C^v, C^m) and learnable mask tokens X^m, our decoder processes a concatenated input of encoded visible tokens and noisy mask tokens, paired with positional embeddings that combine ground-truth positions for visible patches PE^v and predicted positions $PE^{m,pred}$ for masked patches. This can be expressed as:

$$H^m = \text{Decoder}(T^v, X^m, PE^v, \text{SG}(PE^{m,pred}), TE), \tag{10}$$

where $H^m \in \mathbb{R}^{r \times d}$ denotes the decoder's output, SG is the stop-gradient operation. By using ground-truth PE^v, we leverage accurate spatial context for visible patches, while $PE^{m,pred}$, derived from the encoder's predicted centers C^m via stop-gradient, prevents leakage of ground-truth masked patch positions and encourages the decoder reliance on the encoder's learned representations. The stop-gradient further ensures that decoder gradients do not disrupt the encoder's center diffusion task, preserving the encoder's robust feature representations. Subsequently, H^m is passed through a

MLP layer for masked coordinate reconstruction, producing $\hat{P}^m \in \mathbb{R}^{r\times k\times 3}$. This hybrid approach enhances the robustness and generalization of patch reconstruction, complementing the encoder's sparse center denoising objective.

3.5 Training objective

Center diffusion loss. The encoder predicts clean group centers from noisy counterparts by leveraging latent features of visible and masked tokens in the forward diffusion process. It predicts visible group centers by feeding their latent features Z^v into a dedicated MLP, yielding $\hat{C}^v = \text{MLP}_v(Z^v)$. Similarly, it predicts masked group centers by processing Z^m through a separate MLP, giving $\hat{C}^m = \text{MLP}_m(Z^m)$, for visible and masked tokens, respectively. These two MLP networks do not share parameters, ensuring that each group (visible and masked) is processed independently. The loss function for this task is computed as follows:

$$\mathcal{L}_{\text{center}} = \frac{1}{g-r}\sum_{i=1}^{g-r}\left\|\hat{C}_i^v - C_i^v\right\|_2^2 + \frac{1}{r}\sum_{j=1}^{r}\left\|\hat{C}_j^m - C_j^m\right\|_2^2, \tag{11}$$

where $\hat{C}_i^v$, $\hat{C}_j^m$ and C_i^v, C_j^m are the predicted and ground-truth visible/masked centers, respectively.

Patch diffusion loss. Unlike the encoder, which predicts the center positions using MSE, the decoder reconstructs masked point cloud patches from noisy inputs. The L2 Chamfer Distance is adopted as the decoder loss function.

$$\mathcal{L}_{patch} = \frac{1}{|\hat{P}^m|}\sum_{\hat{p}\in\hat{P}^m}\min_{p\in P^m}\|\hat{p}-p\|_2^2 + \frac{1}{|P^m|}\sum_{p\in P^m}\min_{\hat{p}\in\hat{P}^m}\|p-\hat{p}\|_2^2. \tag{12}$$

Final loss. Therefore, we have a final loss with a weighting factor:

$$\mathcal{L} = \gamma\mathcal{L}_{center} + \mathcal{L}_{patch}, \tag{13}$$

where γ adjusts the relative importance of the center denoising task. We set it to 0.1 by default. Intuitively, the training process encourages the encoder to learn geometric features from the corrupted inputs and encourages the decoder to reconstruct the original point cloud. The diffusion process introduces structured perturbations to the data and promotes the encoder to capture both local geometric details and global context, thus enhancing its capacity beyond the original Point-MAE. After pre-training, we abandon the decoder and only keep the encoder for downstream tasks.

4 Experiments

4.1 Downstream tasks

Linear evaluation for real-world classification. We first fine-tune the proposed method on real-world scenes for 3D object classification. Rotation is applied for data augmentation during fine-tuning. We take the overall accuracy (OA) on ScanObjectNN [47] subsets as the evaluation metric and summarize experiment results as in Tab. 1. Our Point-MaDi achieves superior performance on all subsets, reaching 95.52%, 93.46%, and 89.52% accuracies, respectively. Compared to the previous Point-MAE [31], our diffusion-based Point-MaDi yields consistent improvements of 5.50%, 5.17%, and 4.34% on OBJ-BG, OBJ-ONLY, and PB-T50-RS, respectively. Furthermore, the performance is competitive with recent cross-modal methods (e.g., ReCon [36], I2P-MAE [67]), without requiring additional modalities or complex pre-training pipelines.

Linear evaluation for synthetic classification. We also conduct experiments on the classification of the synthetic ModelNet40 dataset [56]. Standard random scaling and translation are applied for data augmentation during training. While diffusion-based methods like PointDif may not consistently dominate on the relatively clean and less diverse ModelNet40 dataset, our Point-MaDi still achieves 93.6% accuracy, demonstrating strong generalization without relying on additional modalities or elaborate architectures.

Part semantic segmentation. We conduct part segmentation on the ShapeNetPart [60] dataset. Following previous research, we randomly sample 2,048 points from each input instance and adopt the same segmentation head for the fair comparison, which concatenates the global features with each local feature from the 4th, 8th, and 12th layers of the transformer block, and a shared MLP predicts a part label for each point. Both category mIoU and instance mIoU are computed and presented in

Table 1: Classification accuracy (%) on three variants of ScanObjectNN and ModelNet40. Parameters of inference models #P (M) are listed. We report ScanObjectNN results without voting. ModelNet40 results are shown without and with voting. Not all papers report Acc, indicated as "–" in our table.

Method	Reference	#P	ScanObjectNN			ModelNet40	
			OBJ-BG	OBJ-ONLY	PB-T50-RS	w/o Vote	w/ Vote
Supervised Learning Only							
PointNet [34]	CVPR 2017	3.5	73.3	79.2	68.0	89.2	–
PointNet++ [35]	NeurIPS 2017	1.5	82.3	84.3	77.9	90.7	–
DGCNN [51]	TOG 2019	1.8	82.8	86.2	78.1	92.9	–
SimpleView [11]	ICML 2021	–	–	–	80.5±0.3	93.9	–
PointMLP [25]	ICLR 2022	12.6	–	–	85.4±0.3	94.5	–
P2P-HorNet [52]	NeurIPS 2022	195.8	–	–	89.3	94.0	–
with Single-Modal Self-supervised Learning							
Point-BERT [62]	CVPR 2022	22.1	87.43	88.12	83.07	92.7	93.2
MaskPoint [22]	ECCV 2022	–	89.30	88.10	84.30	–	93.8
Point-MAE [31]	ECCV 2022	22.1	90.02	88.29	85.18	93.2	93.8
Point-M2AE [65]	NeurIPS 2022	15.3	91.22	88.81	86.43	93.4	94.0
PointGPT [5]	NeurIPS 2022	19.5	91.60	90.00	86.90	–	94.0
Point-CMAE [38]	ACCV 2024	22.1	93.46	91.05	88.75	93.6	–
PCP-MAE [68]	NeurIPS 2024	22.1	95.52	94.32	90.35	94.0	94.2
PointDif [70]	CVPR 2024	22.3	91.91	93.29	87.61	–	–
Point-MaDi (Ours)	–	22.1	**95.52**	93.46	89.52	93.6	94.1
Improve (over Point-MAE)	–	–	+5.50	+5.17	+4.34	+0.4	+0.3
with Cross-Modal Self-Supervised Learning							
ACT [9]	ICLR 2023	22.1	93.29	91.91	88.21	93.2	93.7
Joint-MAE [13]	IJCAI 2023	–	90.94	88.86	86.07	–	94.0
I2P-MAE [67]	CVPR 2023	15.3	94.14	91.57	90.11	93.7	94.1
ReCon [36]	ICML 2023	43.6	95.18	**93.63**	**90.63**	**94.1**	**94.5**

Table 2: Part segmentation on ShapeNetPart and semantic segmentation on S3DIS Area 5. The mean intersection over union (mIoU) for all classes (Cls.) and for all instances (Inst.) are reported for Part Segmentation. Mean accuracy (mAcc) and mIoU are reported for Semantic Segmentation.

Method	Reference	Part Seg.		Semantic Seg.	
		Cls. mIoU	Inst. mIoU	mAcc	mIoU
Supervised Learning Only					
PointNet [34]	CVPR 2017	80.4	83.7	49.0	41.1
DGCNN [51]	TOG 2019	82.3	85.2	–	–
PointMLP [25]	ICLR 2022	–	–	–	–
Self-Supervised Representation Learning					
Transformer [48]	NeurIPS 2017	83.4	84.7	68.6	60.0
Point-BERT [62]	CVPR 2022	84.1	85.6	–	–
MaskPoint [22]	ECCV 2022	84.4	86.0	70.1	61.0
Point-MAE [31]	ECCV 2022	84.2	86.1	69.9	60.8
PointGPT [5]	NeurIPS 2022	84.1	86.2	–	–
PointDif [70]	CVPR 2024	84.4	85.8	69.5	60.2
Point-MaDi (Ours)	–	**84.8**	**86.3**	**71.0**	**61.2**
Improve (over Point-MAE)	–	+0.6	+0.2	+1.1	+0.4

Tab. 2. Our Point-MaDi achieves state-of-the-art performance, with a category mIoU of 84.8% and an instance mIoU of 86.3%, improving over Point-MAE by 0.6% and 0.2%, respectively.

3D scene segmentation. We validate our model on the indoor S3DIS [2] dataset to demonstrate the ability of the models to comprehend contextual semantics and intricate local geometric relationships. Tab. 2 demonstrates the performance of our proposed method. Our Point-MaDi achieves superior performance on Area 5, with a mAcc of 71.0% and a mIoU of 61.2%, improving over Point-MAE by 0.2% and 0.4%, respectively. These results underscore the effectiveness of Point-MaDi's dual-diffusion pre-training in capturing complex scene semantics and fine-grained geometric details.

Table 3: Object detection results on ScanNet. We report average precision (%). "Pre Dataset" refers to the pre-training dataset. ScanNet-Medium is a subset of ScanNet.

Method	Reference	[P]	Pre Dataset	AP50
VoteNet [33]	ICCV 2019	×	–	33.5
STRL [16]	ICCV 2021	✓	ScanNet	38.4
PointContrast [57]	ECCV 2020	✓	ScanNet	38.0
3DETR [26]	ICCV 2021	×	–	37.9
Point-BERT [62]	CVPR 2022	✓	ScanNet-Medium	38.3
Mask-Point [22]	ECCV 2022	✓	ScanNet-Medium	42.1
Point-MAE [31]	ECCV 2022	✓	ShapeNet	42.8
TAP [53]	ICCV 2023	✓	ShapeNet	41.4
PointDif [70]	CVPR 2024	✓	ShapeNet	43.7
Point-MaDi (Ours)	–	✓	ShapeNet	**43.7**
Improve (over Point-MAE)	–	-	-	+0.9

Table 4: Classification accuracy (%) of decoder architectures on ScanObjectNN variants. The configurations differ in how attention is applied between visible and masked tokens.

Decoder Configuration	OBJ-BG	OBJ-ONLY	PB-T50-RS
Joint decoder	**95.52**	**93.46**	**89.52**
Cross decoder	94.66	92.60	88.69
Cross-self decoder	93.63	92.43	87.93

Table 5: Classification accuracy (%) of masking strategies on ScanObjectNN variants. "Random" is random masking, "Block" is block masking, "Rand & Block" combines both.

Masking Strategy	OBJ-BG	OBJ-ONLY	PB-T50-RS
Rand	93.98	92.77	88.45
Block	93.63	91.57	88.10
Rand & Block	**95.52**	**93.46**	**89.52**

3D object detection. To further demonstrate the scene understanding ability of the proposed method, we fine-tune our Point-MaDi on the more challenging indoor dataset ScanNetV2 [6]. Following MaskPoint, we utilize 3DETR [26] as the baseline and replace the encoder with our Point-MaDi backbone. For evaluation purposes, we measure the Average Precision (AP) of 3D bounding boxes with 0.5 thresholds for IoU. Tab. 3 presents the results. Our method, along with Point-MAE and TAP, is pre-trained on ShapeNet in a different domain compared to the ScanNet-Medium dataset. Our Point-MaDi, pretrained on ShapeNet, achieves a state-of-the-art AP50 of 43.7%, improving over Point-MAE by 0.9%. Despite the domain gap Point-MaDi outperforms methods pretrained on ScanNet-Medium, such as MaskPoint (42.1%) and Point-BERT (38.3%), highlighting the robustness of its dual-diffusion pre-training in capturing complex scene semantics and geometric structures.

4.2 Ablation studies

Decoder architecture. We discuss the effect of different decoder designs, exploring three configurations that vary in how attention modules are applied to visible latent tokens T^v and the noise tokens X^m. The joint decoder applies transformer blocks on the concatenated sequence of the visible latent and the noise tokens, enabling self-attention across all tokens to capture global interactions. The Cross decoder takes T^v as queries and X^m as keys and values in cross-attention, mapping noise tokens to reconstructed patches within visible context. The cross-self decoder combines cross-attention, where visible tokens T^v serve as context, with self-attention on noise tokens X^m, allowing noise tokens to interact before querying visible tokens. We conduct experiments with the same depth to investigate the quality of predicted representations. As shown in Tab. 4, the joint decoder achieves the best overall performance. We make the joint decoder the default option for pre-training.

Masking strategy. We assess the impact of different masking strategies on the classification tasks. We trained our model with a 60% mask ratio on three masking settings: Rand: randomly selecting masked and visible parts; Block: masking a large block that contains multiple continuous and consecutive patches; and Rand & Block: which denotes that we feed both masked inputs sequentially through the same network and employ a shared weight prediction head, ensuring no additional parameters are introduced during training. As illustrated in Tab. 5, the Rand & Block strategy achieves the best performance under the same masking ratio. It introduces more spatial diversity in corrupted regions, which encourages the model to learn more robust and generalized representations.

Effective of component. We conduct a comprehensive ablation study focusing on the components of our dual-diffusion framework in Tab. 6. To ensure a fair comparison, we maintained the core Point-MaDi framework. The baseline uses clean patch centers for both visible and masked patches in

Table 6: Classification accuracy (%) of different component configurations on three variants of ScanObjectNN.

Center (Vis)	Center (Mask)	Patch	Time Embedding	OBJ-BG	OBJ-ONLY	PB-T50-RS
-	-	-	-	93.97	92.60	88.83
✓	-	-	-	93.63	92.43	88.13
-	✓	-	-	94.32	92.08	88.79
✓	✓	-	-	94.32	92.94	89.17
-	-	✓	✓	94.66	93.11	89.17
✓	✓	✓	-	94.49	92.43	88.83
✓	✓	✓	✓	**95.52**	**93.46**	**89.52**

the encoder and employs learnable mask tokens in the decoder. The key variable is the activation or deactivation of the diffusion processes. The analysis reveals several key points: firstly, the baseline achieves a performance of 93.97%, 92.60%, 88.83% on OBJ-BG, OBJ-ONLY, PB-T50-RS. When only one set of centers is noised while the other remains clean, the performance compared to the baseline is either slightly degraded or shows mixed results, suggesting that providing clean positional information for only a subset of patches creates an inconsistent learning signal for the encoder; secondly, when both visible and masked centers are noised in the encoder, performance improves to 94.32%, 92.94%, 89.17%, demonstrating that our full center diffusion mechanism, which removes all ground-truth positional shortcuts and forces the encoder to predict all clean centers from noisy inputs, is more effective than partial noising or relying on clean centers. Furthermore, this benefit is amplified when combined with the patch diffusion task in the decoder: the configuration with full center diffusion and patch diffusion enabled achieves the best overall performance, outperforming the scenario where center diffusion is disabled but patch diffusion is active. Additionally, the analysis highlights the positive impact of incorporating time embeddings for patch diffusion processes.

5 Conclusions

In this work, we present a novel self-supervised pre-training framework for point cloud analysis that integrates a dual-diffusion paradigm into a masked autoencoding architecture. By jointly modeling the denoising of both patch centers and masked patches, our method mitigates the risk of geometric information leakage and encourages the learning of more robust semantic and geometric representations. The proposed framework leverages a center diffusion module in the encoder to eliminate reliance on positional embeddings, while a conditional patch diffusion module in the decoder facilitates fine-grained reconstruction guided by visible context. Through extensive experiments across multiple downstream tasks, our approach consistently demonstrates superior performance and generalization. These results validate our initial motivation to force the model to infer global spatial relationships, enhancing its ability to capture comprehensive 3D structural understanding.

6 Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant Nos. U24A20252, and 62327808, the Key Research and Development Program of Shaanxi Province of China under Grant Nos. 2024PT-ZCK-66 and 2024CY2-GJHX-48, Guangdong Major Project of Basic and Applied Basic Research under Grant No. 2023B0303000009, and the Fundamental Research Funds for the Central Universities under Grant No. xzy022024007.

References

[1] M. Afham, I. Dissanayake, D. Dissanayake, A. Dharmasiri, K. Thilakarathna, and R. Rodrigo. Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9902–9912, 2022.

[2] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1534–1543, 2016.

[3] M. Caron, N. Houlsby, and C. Schmid. Location-aware self-supervised transformers for semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 117–127, 2024.

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

[5] G. Chen, M. Wang, Y. Yang, K. Yu, L. Yuan, and Y. Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *Advances in Neural Information Processing Systems*, 36:29667–29679, 2023.

[6] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[8] P. Dhariwal and A. Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[9] R. Dong, Z. Qi, L. Zhang, J. Zhang, J. Sun, Z. Ge, L. Yi, and K. Ma. Autoencoders as cross-modal teachers: Can pretrained 2d image transformers help 3d representation learning? *arXiv preprint arXiv:2212.08320*, 2022.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] A. Goyal, H. Law, B. Liu, A. Newell, and J. Deng. Revisiting point cloud shape classification with a simple and effective baseline. In *International conference on machine learning*, pages 3809–3820. PMLR, 2021.

[12] Y. Guo, C. Yang, A. Rao, M. Agrawala, D. Lin, and B. Dai. Sparsectrl: Adding sparse controls to text-to-video diffusion models. In *European Conference on Computer Vision*, pages 330–348. Springer, 2024.

[13] Z. Guo, R. Zhang, L. Qiu, X. Li, and P.-A. Heng. Joint-mae: 2d-3d joint masked autoencoders for 3d point cloud pre-training. *arXiv preprint arXiv:2302.14007*, 2023.

[14] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[15] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[16] S. Huang, Y. Xie, S.-C. Zhu, and Y. Zhu. Spatio-temporal self-supervised representation learning for 3d point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6535–6545, 2021.

[17] G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 577–593. Springer, 2016.

[18] Y. Li, H. Liu, Q. Wu, F. Mu, J. Yang, J. Gao, C. Li, and Y. J. Lee. Gligen: Open-set grounded text-to-image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22511–22521, 2023.

[19] Y. Li, C. Madarasingha, and K. Thilakarathna. Diffpmae: Diffusion masked autoencoders for point cloud reconstruction. In *European Conference on Computer Vision*, pages 362–380. Springer, 2024.

[20] Z. Li, Z. Chen, A. Li, L. Fang, Q. Jiang, X. Liu, J. Jiang, B. Zhou, and H. Zhao. Simipu: Simple 2d image and 3d point cloud unsupervised pre-training for spatial-aware visual representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1500–1508, 2022.

[21] Z. Lin, Y. Gong, Y. Shen, T. Wu, Z. Fan, C. Lin, N. Duan, and W. Chen. Text generation with diffusion language models: A pre-training approach with continuous paragraph denoise. In *International Conference on Machine Learning*, pages 21051–21064. PMLR, 2023.

[22] H. Liu, M. Cai, and Y. J. Lee. Masked discrimination for self-supervised learning on point clouds. In *European Conference on Computer Vision*, pages 657–675. Springer, 2022.

[23] Y. Liu, C. Chen, Z. Wang, and L. Yi. Crossvideo: Self-supervised cross-modal contrastive learning for point cloud video understanding. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12436–12442. IEEE, 2024.

[24] S. Luo and W. Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2837–2845, 2021.

[25] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*, 2022.

[26] I. Misra, R. Girdhar, and A. Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2906–2917, 2021.

[27] S. Mo, E. Xie, R. Chu, L. Hong, M. Niessner, and Z. Li. Dit-3d: Exploring plain diffusion transformers for 3d shape generation. *Advances in neural information processing systems*, 36:67960–67971, 2023.

[28] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

[29] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*, pages 69–84. Springer, 2016.

[30] L. Nunes, R. Marcuzzi, B. Mersch, J. Behley, and C. Stachniss. Scaling diffusion models to real-world 3d lidar scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14770–14780, 2024.

[31] Y. Pang, W. Wang, F. E. Tay, W. Liu, Y. Tian, and L. Yuan. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pages 604–621. Springer, 2022.

[32] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[33] C. R. Qi, O. Litany, K. He, and L. J. Guibas. Deep hough voting for 3d object detection in point clouds. In *proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9277–9286, 2019.

[34] C. R. Qi, H. Su, K. Mo, and L. 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*, pages 652–660, 2017.

[35] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[36] Z. Qi, R. Dong, G. Fan, Z. Ge, X. Zhang, K. Ma, and L. Yi. Contrast with reconstruct: Contrastive 3d representation learning guided by generative pretraining. In *International Conference on Machine Learning*, pages 28223–28243. PMLR, 2023.

[37] Z. Qi, R. Dong, S. Zhang, H. Geng, C. Han, Z. Ge, L. Yi, and K. Ma. Shapellm: Universal 3d object understanding for embodied interaction. In *European Conference on Computer Vision*, pages 214–238. Springer, 2024.

[38] B. Ren, G. Mei, D. P. Paudel, W. Wang, Y. Li, M. Liu, R. Cucchiara, L. Van Gool, and N. Sebe. Bringing masked autoencoders explicit contrastive properties for point cloud self-supervised learning. In *Proceedings of the Asian Conference on Computer Vision*, pages 2034–2052, 2024.

[39] J. T. Rolfe. Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*, 2016.

[40] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[41] L. Ruan, Y. Ma, H. Yang, H. He, B. Liu, J. Fu, N. J. Yuan, Q. Jin, and B. Guo. Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10219–10228, 2023.

[42] S. Sameni, S. Jenni, and P. Favaro. Representation learning by detecting incorrect location embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9704–9713, 2023.

[43] Y. Song and S. Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

[44] Y. Song and S. Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020.

[45] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[46] B. Tran, B.-S. Hua, A. T. Tran, and M. Hoai. Self-supervised learning with multi-view rendering for 3d point cloud analysis. In *Proceedings of the Asian Conference on Computer Vision*, pages 3086–3103, 2022.

[47] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung. 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*, pages 1588–1597, 2019.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[49] H. Wang, J. Fan, Y. Wang, K. Song, T. Wang, and Z.-X. ZHANG. Droppos: Pre-training vision transformers by reconstructing dropped positions. *Advances in Neural Information Processing Systems*, 36:46134–46151, 2023.

[50] H. Wang, Q. Liu, X. Yue, J. Lasenby, and M. J. Kusner. Unsupervised point cloud pre-training via occlusion completion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9782–9792, 2021.

[51] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019.

[52] Z. Wang, X. Yu, Y. Rao, J. Zhou, and J. Lu. P2p: Tuning pre-trained image models for point cloud analysis with point-to-pixel prompting. *Advances in neural information processing systems*, 35:14388–14402, 2022.

[53] Z. Wang, X. Yu, Y. Rao, J. Zhou, and J. Lu. Take-a-photo: 3d-to-2d generative pre-training of point cloud models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5640–5650, 2023.

[54] C. Wei, L. Xie, X. Ren, Y. Xia, C. Su, J. Liu, Q. Tian, and A. L. Yuille. Iterative reorganization with weak spatial constraints: Solving arbitrary jigsaw puzzles for unsupervised representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1910–1919, 2019.

[55] S. Wu, Y. Lin, F. Zhang, Y. Zeng, J. Xu, P. Torr, X. Cao, and Y. Yao. Direct3d: Scalable image-to-3d generation via 3d latent diffusion transformer. *arXiv preprint arXiv:2405.14832*, 2024.

[56] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[57] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 574–591. Springer, 2020.

[58] J. Xu, X. Wang, W. Cheng, Y.-P. Cao, Y. Shan, X. Qie, and S. Gao. Dream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20908–20918, 2023.

[59] L. Xue, M. Gao, C. Xing, R. Martín-Martín, J. Wu, C. Xiong, R. Xu, J. C. Niebles, and S. Savarese. 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*, pages 1179–1189, 2023.

[60] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

[61] H.-T. Yu and M. Song. Mm-point: Multi-view information-enhanced multi-modal self-supervised 3d point cloud understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 6773–6781, 2024.

[62] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19313–19322, 2022.

[63] Y. Zeng, C. Jiang, J. Mao, J. Han, C. Ye, Q. Huang, D.-Y. Yeung, Z. Yang, X. Liang, and H. Xu. Clip2: Contrastive language-image-point pretraining from real-world point cloud data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15244–15253, 2023.

[64] L. Zhang, A. Rao, and M. Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3836–3847, 2023.

[65] R. Zhang, Z. Guo, P. Gao, R. Fang, B. Zhao, D. Wang, Y. Qiao, and H. Li. Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. *Advances in neural information processing systems*, 35:27061–27074, 2022.

[66] R. Zhang, Z. Guo, W. Zhang, K. Li, X. Miao, B. Cui, Y. Qiao, P. Gao, and H. Li. Pointclip: Point cloud understanding by clip. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8552–8562, 2022.

[67] R. Zhang, L. Wang, Y. Qiao, P. Gao, and H. Li. Learning 3d representations from 2d pre-trained models via image-to-point masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21769–21780, 2023.

[68] X. Zhang, S. Zhang, and J. Yan. Pcp-mae: Learning to predict centers for point masked autoencoders. *Advances in Neural Information Processing Systems*, 37:80303–80327, 2024.

[69] Z. Zhang, R. Girdhar, A. Joulin, and I. Misra. Self-supervised pretraining of 3d features on any point-cloud. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10252–10263, 2021.

[70] X. Zheng, X. Huang, G. Mei, Y. Hou, Z. Lyu, B. Dai, W. Ouyang, and Y. Gong. Point cloud pre-training with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22935–22945, 2024.

[71] Y. Zhou, B. Liu, Y. Zhu, X. Yang, C. Chen, and J. Xu. Shifted diffusion for text-to-image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10157–10166, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The Abstract and Introduction sections clearly and accurately summarize the paper's core contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Our paper includes a dedicated discussion of limitations and future directions in Appendix E.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This paper does not include theoretical results that require a full proof.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide comprehensive details to ensure the reproducibility of our experimental results. This includes clear descriptions of the model architecture, training settings, dataset splits, and data preprocessing steps. For each downstream task, we specify the exact hyperparameters and configurations used. All results are obtained using standard publicly available datasets, and the experimental setups are described in Appendix A.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have already open sourced our method. Experiments can be replicated using our open source PyTorch optimizer implementation together with existing open source code bases implementing each method.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have detailed all the training and evaluation settings before the main results in the experimental part.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have reported the experimental results averaged over multiple random seeds.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide information of GPU type and number of GPUs used for running our experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read the NeurIPS Code of Ethics and confirm that this research conforms to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: We foresee no potential societal impact of this work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper does not involve data or models with a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All assets used in this work have been properly credited, and the licenses and terms of use are explicitly mentioned and respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This research does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This research does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core methods of this research do not involve LLMs as important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Table 7: Training details for pre-training and downstream fine-tuning.

Config	ShapeNet	ScanObjectNN	ModelNet	ShapeNetPart	S3DIS
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Learning rate	5e-4	2e-5	1e-5	2e-4	2e-4
Weight decay	5e-2	5e-2	5e-2	5e-2	5e-2
Learning rate scheduler	cosine	cosine	cosine	cosine	cosine
Training epochs	300	300	300	300	60
Warmup epochs	10	10	10	10	10
Batch size	128	32	32	16	32
Drop path rate	0.1	0.1	0.1	0.1	0.1
Number of points	1024	2048	1024	2048	2048
Number of point patches	64	128	64	128	128
Diffusion step	2000	–	–	–	–
Beta start	1e-4	–	–	–	–
Beta end	1e-2	–	–	–	–
Augmentation	Scale&Trans+Rotation	Rotation	Scale&Trans	–	–
GPU device	RTX 3090	RTX 3090	RTX 3090	RTX 3090	RTX 3090

Appendix

A Experimental Settings Details

Pre-training details. We sample each input 1,024 points and divide them into 64 groups, each containing 32 points. We apply scale and translation operations, followed by rotation for data augmentation. The model is pre-trained with a batch size of 128 for 300 total epochs. Following Point-BERT [62], we set the hidden dimension of each encoder block to 384, the number of heads to 6, and the FFN expansion ratio to 4. The depth of the transformer decoder is set to 4. During pre-training, we adopt the AdamW optimizer with a weight decay of 0.05 and an initial learning rate of 5×10^{-4} with the cosine decay. All experiments are conducted on a single GeForce RTX 3090. To ensure a fair comparison, we employed identical experimental settings to the default fine-tuning. More details are provided in Tab. 7.

Dataset details. The ShapeNet [4] is used as our pre-training dataset; it covers over 50,000 unique 3D models from 55 common categories. ScanObjectNN [47] contains 15K unique 3D point cloud objects spanning 15 diverse categories, scanned from indoor scenes obtained by scanning, often characterized by cluttered backgrounds and occlusions. We evaluate our Point-MaDi on three variants: OBJ-BG, OBJ-ONLY, and PB-T50-RS, each with increasing complexity. ModelNet40 [56] includes 12,311 clean 3D CAD objects with 40 different categories; these objects are split into 9,843 samples in the official training set and 2,468 in the test set. ShapeNetPart [60] dataset contains 14,007 and 2,874 samples with 16 object categories and 50 semantic parts for training and validation. S3DIS [2] consists of 3D scan data from 271 rooms across 6 different indoor spaces, which are annotated into 13 classes. We evaluate our model on Area 5, while the other areas are used for fine-tuning our model.

B Model Efficiency Comparison

We also have conducted additional experiments comparing the pre-training efficiency of our method, Point-MaDi, against both single-modal and cross-modal masked autoencoding approaches, including the most relevant recent work PointDif. As shown in Tab. 8, we report four key metrics: the number of parameters, GFLOPs, pre-training time (hours), and downstream classification performance on ScanObjectNN and ModelNet40. Point-MaDi achieves a similar or faster pre-training time compared to PointDif (14.4h vs. 14.8h), while avoiding complex multi-stage training. Compared to other MAE-based methods, Point-MaDi introduces only a slight increase in parameters and pre-training cost, while maintaining high efficiency.

Table 8: Comparison of existing single-modal and cross-modal MAE methods in terms of pre-training efficiency and representation capability on standard SSL benchmarks.

Method	Reference	Single/Cross Modal	Pre-training efficiency			Performance	
			# Params	GFLOPS	Time (h)	ScanObjectNN	ModelNet40
Point-MAE [31]	ECCV 2022	Single	29.0	**2.3**	13.1	85.18	93.8
Point-M2AE [65]	NeurIPS 2022	Single	**15.3**	3.7	29.1	86.43	94.0
PointDif [70]	CVPR 2024	Single	25.5	-	14.8	87.61	-
ACT [9]	ICLR 2023	Cross	135.5	31.0	52.8	88.21	93.7
I2P-MAE [67]	CVPR 2023	Cross	74.9	16.8	64.4	90.11	94.1
ReCon [36]	ICML 2023	Cross	140.9	20.9	28.3	**90.63**	**94.5**
Point-MaDi	–	Single	29.5	2.9	**14.4**	89.52	94.1

Table 9: Few-shot classification results on ModelNet40. We perform ten separate trials for each experimental setting and the mean accuracy (%) and standard deviation are reported.

Method	5-way		10-way	
	10-shot	20-shot	10-shot	20-shot
Supervised Learning Only				
PointNet [34]	52.0±3.8	57.8±4.9	46.6±4.3	35.2±4.8
DGCNN [51]	31.6±2.8	40.8±4.6	19.9±2.1	16.9±1.5
OcCo [50]	90.6±2.8	92.5±1.9	82.9±1.3	86.5±2.2
with Single-Modal Self-Supervised Representation Learning				
Point-BERT [62]	94.6±3.1	96.3±2.7	91.0±5.4	92.7±5.1
MaskPoint [22]	95.0±3.7	97.2±1.7	91.4±4.0	93.4±3.5
Point-MAE [31]	96.3±2.5	97.8±1.8	92.6±4.1	95.0±3.0
Point-M2AE [65]	96.8±1.8	98.3±1.4	92.3±4.5	95.0±3.0
PointGPT [5]	96.8±2.0	98.6±1.1	92.6±4.6	95.2±3.4
Point-MaDi (Ours)	97.2±1.9	**99.0±0.9**	**93.5±4.3**	95.7±2.3
Improve (over Point-MAE)	+0.9	+1.2	+0.9	+0.7
with Cross-Modal Self-Supervised Representation Learning				
ACT [9]	96.8±2.3	98.0±1.4	93.3±4.0	95.6±2.8
Joint-MAE [13]	96.7±2.2	97.9±1.9	92.6±3.7	95.1±2.6
I2P-MAE [67]	97.0±1.8	98.3±1.3	92.6±5.0	95.5±3.0
TAP [53]	**97.3±1.8**	97.8±1.9	93.1±2.6	**95.8±1.0**
ReCon [36]	97.3±1.9	98.9±1.2	93.3±3.9	95.8±3.0

C Additional Experimental Results

C.1 Additional downstream tasks

3D object few-shot classification. We conduct few-shot classification experiments on the ModelNet40 dataset to evaluate our model's ability to generalize to new categories with limited labeled data. In the N-way K-shot setting, N classes are randomly selected, and K instances per class are used for training. We evaluate configurations with $N \in \{5, 10\}$ and $K \in \{10, 20\}$. For each class, K samples are utilized for fine-tuning the model, while 20 unseen samples per class are reserved for testing. We perform 10 independent trials for each configuration following previous works [50, 62]. The mean accuracy (%) and standard deviation across these trials are reported, as shown in Tab. 9. The results demonstrate that our method achieves superior performance compared to recent state-of-the-art approaches across various settings.

C.2 Additional ablation studies

Mask ratio. We evaluate the effect of varying the mask ratio to assess its impact on downstream performance. As shown in Tab. 10, the optimum mask ratio of our Point-MaDi is 60%. In our framework, Point-MaDi learns features from the ground truth of masked patches and uses visible patches and predicted centers as guidance to reconstruct masked patches only. Hence, a lower mask ratio provides excessive visible context, which simplifies center denoising and weakens the

Table 10: Performance of our model on different masking strategies. The accuracies (%) are reported on three variants of ScanObjectNN.

Mask Ratio	OBJ-BG	OBJ-ONLY	PB-T50-RS
40%	93.98	91.91	88.03
50%	93.63	92.43	88.51
60%	**95.52**	**93.46**	**89.52**
65%	94.66	92.25	88.31
70%	94.32	92.43	88.24
75%	93.12	92.43	88.72
80%	92.77	92.08	88.51

Table 11: Effect of different loss functions for $\mathcal{L}_{center}$ and $\mathcal{L}_{patch}$. The accuracies (%) are reported on three variants of ScanObjectNN.

Loss Function	OBJ-BG	OBJ-ONLY	PB-T50-RS
MSE, MSE	94.49	92.43	87.89
CDL2, CDL2	91.91	90.71	86.47
MSE, CDL2	**95.52**	**93.46**	**89.52**
Smooth L1, CDL2	94.84	92.25	88.83

Table 12: The number of decoder depth. The accuracies (%) are reported on three variants of ScanObjectNN.

Decoder Depth	# P (M)	OBJ-BG	OBJ-ONLY	PB-T50-RS
1	**24.36**	94.49	91.57	87.75
4	29.68	**95.52**	**93.46**	**89.52**
8	36.78	93.98	90.88	87.54
12	43.87	93.12	92.60	88.58

difficulty of learning meaningful spatial correlations, while higher ratios leave too few visible patches, hindering the encoder's ability to infer accurate centers.

Loss function. We explore different loss function settings to evaluate the impact on the pre-training objective by varying the loss functions for $\mathcal{L}_{center}$ and $\mathcal{L}_{patch}$. We test four combinations: (1) MSE for both $\mathcal{L}_{center}$ and $\mathcal{L}_{patch}$, (2) L2 Chamfer Distance for both, (3) MSE for $\mathcal{L}_{center}$ and CDL2 for $\mathcal{L}_{patch}$, and (4) Smooth L1 for $\mathcal{L}_{center}$ and CDL2 for $\mathcal{L}_{patch}$. The results are shown in Tab. 11. The combination of MSE for $\mathcal{L}_{center}$ and CDL2 achieves the best performance across all benchmarks. This suggests that MSE effectively aligns sparse group centers to ground truth by penalizing coordinate errors directly. Using CDL2 for both yields the lowest performance. Smooth L1 for $\mathcal{L}_{center}$ mitigates outlier effects but provides less precise center predictions than MSE. Consequently, we adopt the MSE loss for center prediction and CDL2 for patch reconstruction.

Effect of decoder depth. We investigate the impact of decoder depth in Point-MaDi to assess whether increasing the number of transformer layers enhances the quality of patch reconstruction. The results are presented in Tab. 12. The performance improves as the depth increases from 1 to 4, indicating that a moderate number of layers helps the decoder better process the predicted representations. Nevertheless, Point-MaDi does not benefit from a larger depth, as excessive layers (8–12) lead to overfitting, focusing on overly specific patch details.

Loss weighting. The final loss involves a weighted combination of the center denoising loss and the patch reconstruction loss. We evaluate different weighting values to examine their influence on performance. As shown in Tab. 13, setting γ yields the best overall results across all datasets. We also observe a consistent decline in performance as γ increases, indicating that excessive emphasis on patch reconstruction may distract the model from learning robust and generalizable structural representations through center denoising.

Diffusion timestep. To measure the impact of times t_p on the patch diffusion process, we pre-train the model with different values of $t_p \in \{20, 100, 200, 400, 1000, 2000\}$, keeping other hyperparameters fixed. The results are shown in Tab. 14. We observe that performance generally improves with

Table 13: Performance with different γ values. The accuracies (%) are reported on three variants of ScanObjectNN.

γ	OBJ-BG	OBJ-ONLY	PB-T50-RS
0.1	**95.52**	**93.46**	**89.52**
0.2	93.98	92.60	88.10
0.4	93.98	91.91	88.72
0.6	94.66	92.08	88.17
0.8	94.15	91.91	88.45

Table 14: The impact of pre-training with different timestep t_p. The accuracies (%) are reported on three variants of ScanObjectNN.

t_p	OBJ-BG	OBJ-ONLY	PB-T50-RS
20	94.49	92.25	88.17
100	94.66	93.29	88.65
200	94.84	92.77	88.72
400	93.98	92.77	88.65
1000	93.98	92.94	88.38
2000	**95.52**	**93.46**	**89.52**

Table 15: The impact of timestep schedules for center t_c and patch t_p diffusion. The accuracies (%) are reported on three variants of ScanObjectNN.

t_c	t_p	OBJ-BG	OBJ-ONLY	PB-T50-RS
200	2000	94.15	92.60	88.90
1000	2000	94.15	92.60	88.72
2000	2000	**95.52**	**93.46**	**89.52**
200	1000	93.80	92.94	88.97
2000	200	94.66	91.91	88.72

increasing t_p, peaking at $t_p = 2000$. This trend suggests that a larger timestep introduces a broader noise range, forcing the encoder to learn robust center predictions across diverse noise levels.

Timestep schedule. In our default setting, center diffusion and patch diffusion are sampled simultaneously using the same number of time steps $t_c = t_p = 2000$ with a shared linear variance schedule (β_t from 0.0001 to 0.02). We conducted ablation experiments by varying t_c and t_p independently. Results are shown in Tab. 15. We observe that desynchronizing the diffusion schedules leads to performance degradation in all cases. Interestingly, the model is more sensitive to changes in the decoder-side patch diffusion t_p compared to variations in the encoder-side center diffusion t_c. When t_p is reduced while keeping t_c fixed (e.g., $t_p = 200$), the performance drops more significantly, likely due to insufficient corruption during training, which weakens the decoder's ability to reconstruct complex local geometry. In contrast, reducing t_c while keeping t_p fixed leads to a smaller, though still noticeable, performance drop.

Time embedding of the encoder. In our framework, we intentionally omit time-step embeddings in the transformer encoder during the center diffusion process. The motivation behind this decision is to align the encoder's architecture with downstream tasks, where no diffusion steps or time conditioning are present. By avoiding the introduction of time embeddings during pre-training, we ensure that the encoder learns time-agnostic, task-agnostic features that generalize better across downstream domains. To validate this design decision, we conducted an ablation study comparing two variants in Tab. 16: 1) With time embedding added to the encoder; 2) Without time embedding, as used in our default implementation. The results on the ScanObjectNN benchmark are summarized below. Incorporating time embeddings led to consistent performance degradation across all three variants. We hypothesize that the time embeddings may introduce unnecessary conditioning noise or reduce the encoder's ability to generalize to downstream inputs, which are always clean and have no associated time-step semantics.

Table 16: The effect of time embedding in the encoder. The accuracies (%) are reported on three variants of ScanObjectNN.

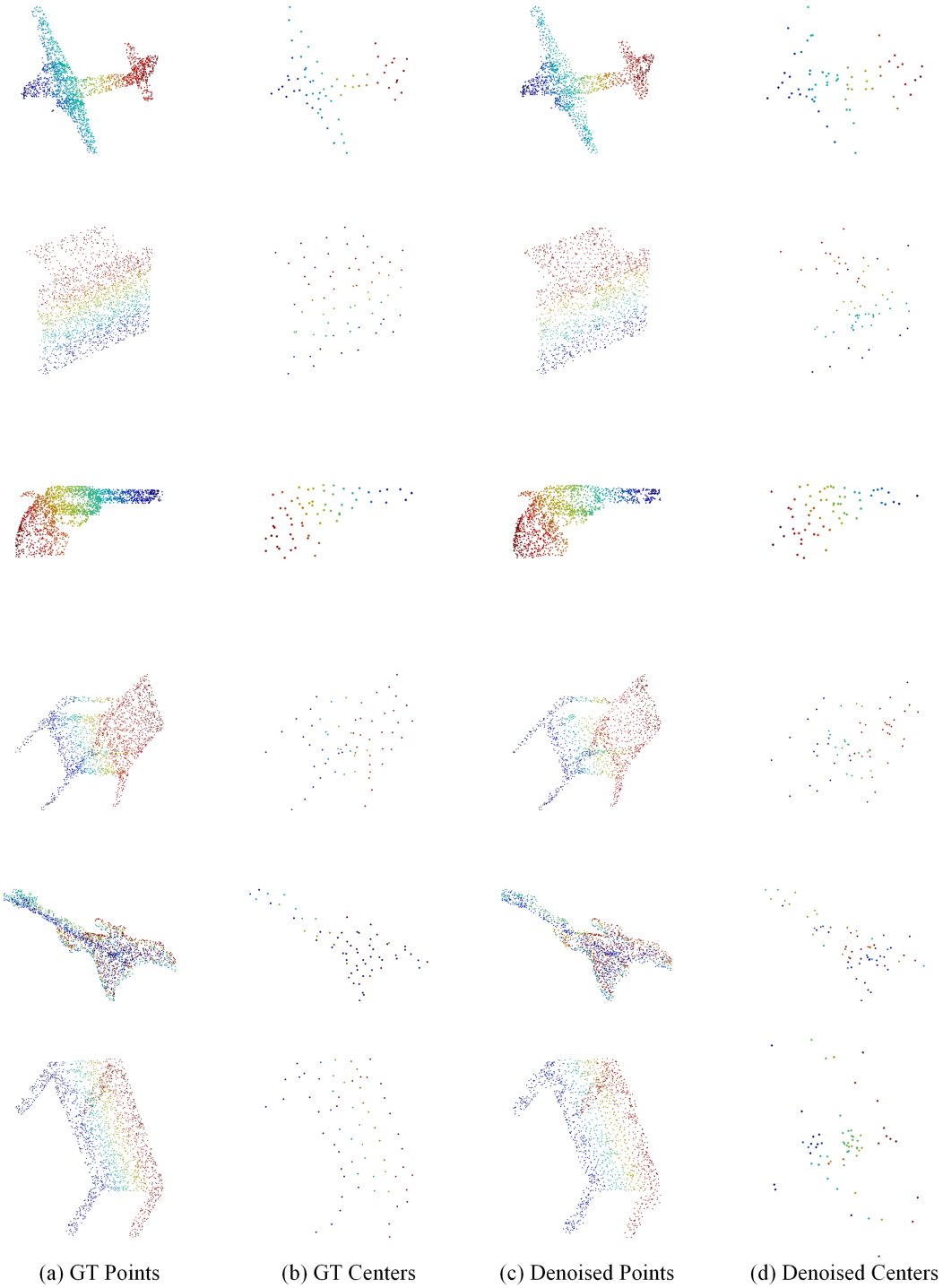
Time Embedding	OBJ-BG	OBJ-ONLY	PB-T50-RS
×	94.49	92.59	88.83
✓	**95.52**	**93.46**	**89.52**

D Additional Visualization Results

We present additional visualizations on the ShapeNet [4] test split to evaluate the denoising performance of our Point-MaDi model. The encoder independently predicts visible and masked center points from 3D input coordinates, which are concatenated to form the denoised centers. Concurrently, the decoder reconstructs the full point cloud from encoded features. As illustrated in Figures 3–5, the denoised centers closely align with the GT centers across diverse categories, demonstrating the encoder's ability to capture global structure effectively. Meanwhile, the decoder reconstructs both global shape and local geometry in the denoised points. These results highlight the model's strengths in global representation and reconstruction, with ongoing challenges in detailed local recovery.

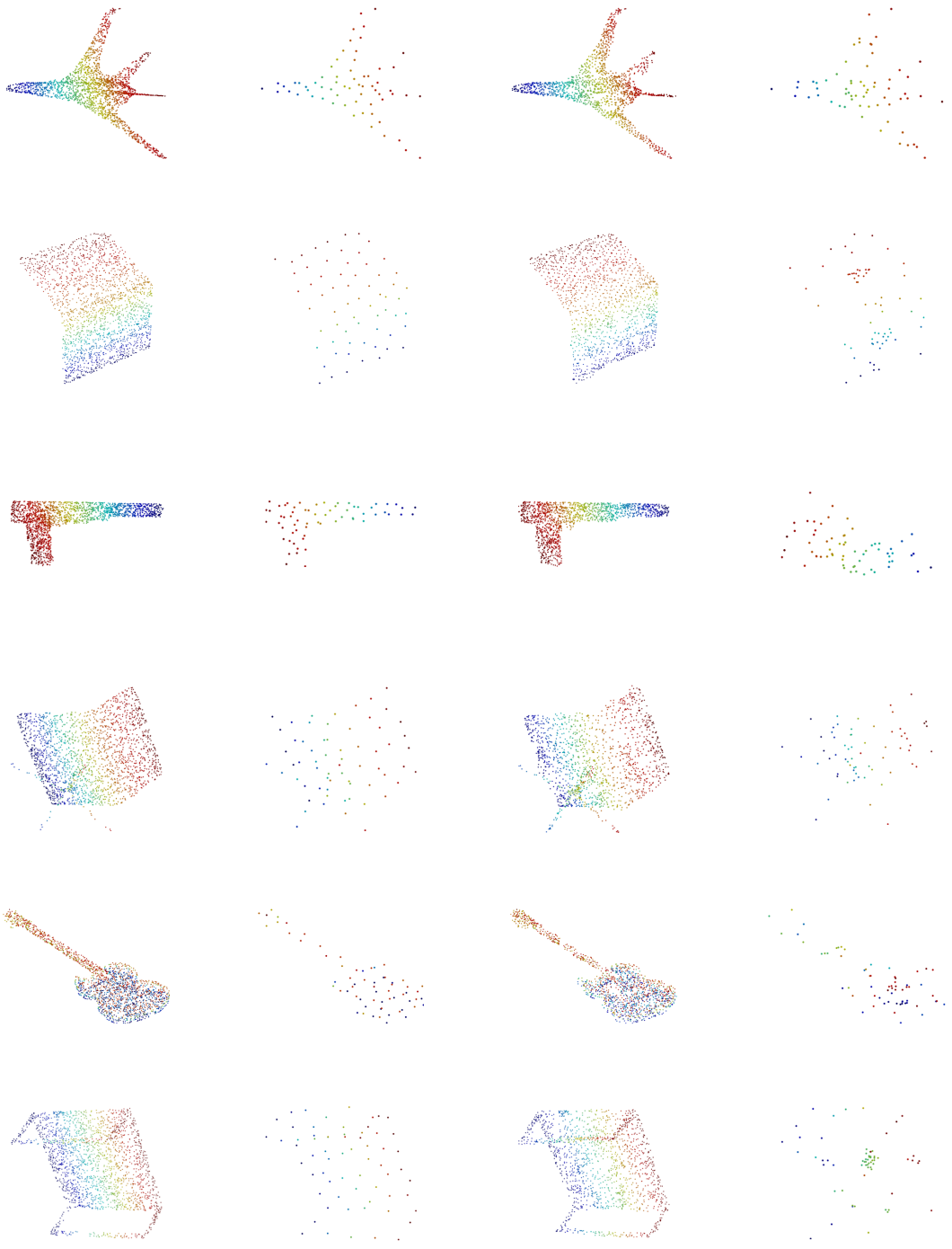
E Limitations

This work primarily focuses on learning robust geometric and semantic representations from point clouds in a purely self-supervised and unimodal setting. While the proposed dual-diffusion framework achieves consistent improvements across various downstream tasks, several directions remain open for future exploration. One natural extension is to incorporate multi-modal information, such as images or language, to further enrich the learned representations and enhance scene understanding. Additionally, while our encoder effectively predicts denoised centers that capture the global shape of point clouds, it struggles to preserve fine-grained local details, limiting the precision of local geometry in the predicted centers.



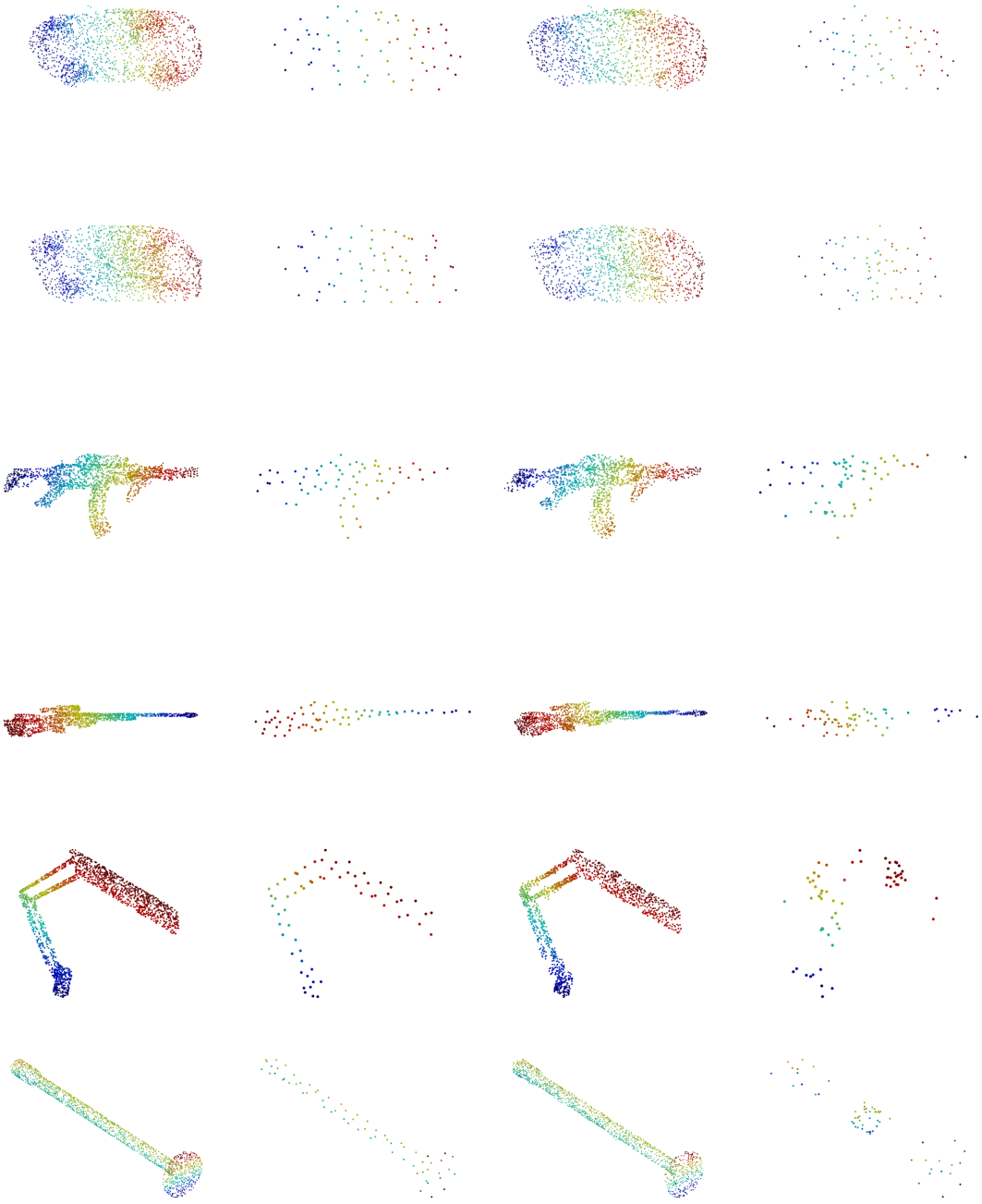
(a) GT Points (b) GT Centers (c) Denoised Points (d) Denoised Centers

Figure 3: Visualization of point cloud denoising by Point-MaDi. (a) GT Points: original point cloud on ShapeNet test split. (b) GT Centers: FPS-sampled centers. (c) Denoised Points: decoder-reconstructed point cloud. (d) Denoised Centers: encoder-predicted center points.



(a) GT Points (b) GT Centers (c) Denoised Points (d) Denoised Centers

Figure 4: Visualization of point cloud denoising by Point-MaDi. (a) GT Points: original point cloud on ShapeNet test split. (b) GT Centers: FPS-sampled centers. (c) Denoised Points: decoder-reconstructed point cloud. (d) Denoised Centers: encoder-predicted center points.



(a) GT Points (b) GT Centers (c) Denoised Points (d) Denoised Centers

Figure 5: Visualization of point cloud denoising by Point-MaDi. (a) GT Points: original point cloud on ShapeNet test split. (b) GT Centers: FPS-sampled centers. (c) Denoised Points: decoder-reconstructed point cloud. (d) Denoised Centers: encoder-predicted center points.