Point-M2AE: Multi-scale Masked Autoencoders for Hierarchical Point Cloud Pre-training

Anonymous Author(s)
Affiliation
Address
email

Abstract

Masked Autoencoders (MAE) have shown great potentials in self-supervised pre-training for language and 2D image transformers. However, it still remains an open question on how to exploit masked autoencoding for learning 3D representations of irregular point clouds. In this paper, we propose **Point-M2AE**, a strong **M**ulti-scale **M**AE pre-training framework for hierarchical self-supervised learning of 3D point clouds. Unlike the standard transformer in MAE, we modify the encoder and decoder into pyramid architectures to progressively model spatial geometries and capture both fine-grained and high-level semantics of 3D shapes. For the encoder that downsamples point tokens by stages, we design a multi-scale masking strategy to generate consistent visible regions across scales, and adopt a local spatial self-attention mechanism to focus on neighboring patterns. By multi-scale token propagation, the lightweight decoder gradually upsamples point tokens with complementary skip connections from the encoder, which further promotes the reconstruction from a global-to-local perspective. Extensive experiments demonstrate the *state-of-the-art* performance of Point-M2AE for 3D representation learning. With a frozen encoder after pre-training, Point-M2AE achieves **92.9%** accuracy for linear SVM on ModelNet40, even surpassing some fully trained methods. By fine-tuning on downstream tasks, Point-M2AE achieves **86.43%** accuracy on ScanObjectNN,**+3.36%** to the second-best, and largely benefits the few-shot classification, part segmentation and 3D object detection with the hierarchical pre-training scheme.

1 Introduction

Learning to represent from unlabeled data without annotations, known as self-supervised learning, has attained great success in natural language processing [10, 31, 32, 5], computer vision [19, 7, 8, 18] and multi-modality learning [30, 49, 21]. By pre-training on the large-scale raw data, the networks are endowed with robust representation abilities and can significantly benefit downstream tasks with fine-tuning. Motivated by masked language modeling [31, 10], MAE [18] and some other methods [45, 52, 3] adopt asymmetric encoder-decoder transformers [13] to apply masked autoencoding for self-supervised learning on 2D images. They represent the input image as multiple local patches, and randomly mask them with a high ratio to build the pretext task for reconstruction. Specifically, the encoder aims at capturing high-level latent representations from limited visible patches, and the lightweight decoder is forced to reconstruct the RGB values of masked patches on top. Despite its superiority on grid-based 2D images, we ask the question: can MAE-style masked autoencoding be adapted to irregular point clouds as a powerful 3D representation learner?

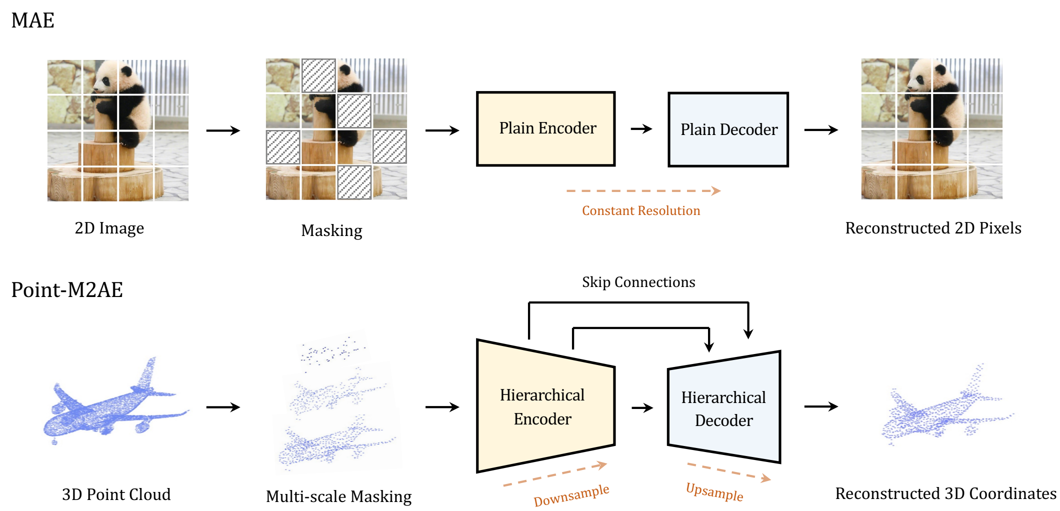

Figure 1: **Comparison of MAE (Top) and our Point-M2AE (Bottom).** MAE for 2D image pre-training adopts standard transformer of the plain encoder and decoder, while Point-M2AE introduces a hierarchical transformer with skip connections for multi-scale point cloud pre-training.

To tackle this challenge, we propose **M**ulti-scale **M**asked autoencoders for learning the hierarchical representations of point clouds via self-supervised pre-training, termed as Point-M2AE. We represent a point cloud as a set of point tokens depicting different spatial local regions, and inherit MAE's pipeline to first encode visible point tokens and then reconstruct the masked 3D coordinates. Different from 2D images, masked autoencoding for 3D point clouds has three characteristics to be considered. Firstly, it is critical to understand the relations between local parts and the overall 3D shapes, which have strong geometric and semantic dependence. As examples, the network can recognize an airplane starting from its wing, or segment the wing's part from the airplane's global feature. Therefore, we regard the standard transformer with the plain encoder and decoder is sub-optimal for capturing such local-global spatial relations in 3D, which directly downsamples the input into a low-resolution representation as shown in Figure 1 (Top). We modify both the encoder and decoder into multi-stage hierarchies for progressively encoding multi-scale features of point clouds, constructing an asymmetric U-Net [34] like architecture in Figure 1 (Bottom). In detail, the shallower stages of the encoder contain a larger number of point tokens to focus on local patterns, while the deeper stages merge spatially adjacent tokens to acquire global understanding. Secondly, as Point-M2AE encodes multi-scale point clouds unlike the single-scale 2D images, the unmasked visible regions are required to be block-wise within one scale and consistent across scales, which are respectively for reserving more complete local geometries and ensuring coherent feature learning for the network. For this, we introduce a multi-scale masking strategy, which generates random masks at the final scale with a high ratio (e.g., 80%), and back-projects the unmasked positions to all preceding scales. Thirdly, to better capture the fine-grained 3D geometries, we adopt a local spatial self-attention mechanism with increasing attention scopes for point tokens at different stages in the encoder, which refocus each token within neighboring detailed structures. Also, we utilize skip connections to complement the decoder with fine-grained information from the corresponding stages of the encoder.

By the multi-scale pre-training, Point-M2AE can encode point clouds from local-to-global hierarchies and then reconstructs the masked coordinates from global-to-local perspectives, which learns powerful 3D representations and performs superior transfer ability. After self-supervised pre-training on ShapeNet [6], Point-M2AE achieves 92.9% classification accuracy for linear SVM on ModelNet40 [43] with the frozen encoder, which surpasses the runner-up CrossPoint [2] by +1.2% and even outperforms some fully supervised methods. By fine-tuning on various downstream tasks, Point-M2AE achieves 86.43% (+3.36%) accuracy on ScanObjectNN [37] and 94.0% (+0.8%) accuracy on ModelNet40 [43] for shape classification, 86.51% (+0.91%) instance mIoU on ShapeNet-Part [47] for part segmentation, and 95.0% (+2.7%) accuracy on 10-way 20-shot ModelNet40 for few-shot classification. Our multi-scale masked autoencoding also benefits the 3D object detection on ScanNetV2 [9] by +1.3% AP_{25} and +1.3% AP_{50}, which provides the detection backbone with a hierarchical understanding of the point clouds. We summarize the contributions of our paper as follows:

1. We propose Point-M2AE, a strong masked autoencoding framework, which conducts hierarchical point cloud encoding and reconstruction for better learning multi-scale spatial geometries of 3D shapes.
2. We introduce a U-Net like transformer architecture for MAE-style pre-training on point clouds, and adopt a multi-scale masking strategy to generate consistent visible regions across scales.
3. Point-M2AE achieves *state-of-the-art* performance for transfer learning on various downstream tasks, which indicates our approach to be a powerful representation learner for 3D point clouds.

2 Related Work

Pre-training by Masked Modeling. Compared to contrastive learning methods [19, 7, 8] that learn from inter-sample relations, self-supervised pre-training by masked autoencoding builds the pretext tasks to predict the masked parts of the input signals. The series of GPT [31, 32, 5] and BERT [11] apply masked modeling to natural language processing and achieve extraordinary performance boost on downstream tasks with fine-tuning. Inspired by this, BEiT [4] proposes to match image patches with discrete tokens via dVAE [33] and pre-train a standard vision transformer [13, 48] by masked image modeling. On top of that, MAE [18] directly reconstructs the raw pixel values of masked tokens and performs great efficiency with a high mask ratio. The follow-up works further improve the performance of MAE by momentum encoder [52], contrastive learning [3], and modified reconstruction targets [41]. For self-supervised pre-training on 3D point clouds, the masked autoencoding has not been widely adopted. Similar to BEiT, Point-BERT [48] utilizes dVAE to map 3D patches to tokens for masked point modeling, but heavily relies on constrastive learning [19], complicated data augmentation, and the costly two-stage pre-training. In contrast, our Point-M2AE is a pure masked autoencoding method of one-stage pre-training, and follows MAE to reconstruct the input signals without dVAE mapping. Different from previous MAE methods adopting standard plain transformer, we propose a hierarchical transformer architecture along with the multi-scale masking strategy to better learn a strong and generic representation for 3D point clouds.

Self-supervised Learning for Point Clouds. 3D representation learning without annotations has been widely studied in recent years. Mainstream methods mainly build the pretext tasks to reconstruct the transformed input point cloud based on the encoded latent vectors, such as rotation [27], deformation [1], rearranged parts [35] and occlusion [39]. From another perspective, PointContrast [44] utilizes contrastive learning between features of the same points from different views to learn discriminative 3D representations. DepthContrast [50] further extends the contrast for depth maps of different augmentations. CrossPoint [2] conducts cross-modality contrastive learning between point clouds and their corresponding rendering images to acquire rich self-supervised signals. Point-BERT [48] first introduces BERT-style pre-training for 3D point clouds with a standard transformer network and performs competitively on various downstream tasks. In this paper, we propose an MAE-style [18] pre-training framework, Point-M2AE, which reconstructs the highly masked 3D coordinates of the input point cloud for self-supervised learning. Point-M2AE with a hierarchical architecture achieves *state-of-the-art* downstream performance by learning the multi-scale representation of point clouds.

3 Method

The overall pipeline of Point-M2AE is shown in Figure 2, where we encode and reconstruct the point cloud by a hierarchical network architecture. In Section 3.1, We first introduce the masking strategy of Point-M2AE with multi-scale representations of point clouds. Then in Section 3.2 and Section 3.3, we present the details of our encoder and decoder with multi-stage hierarchies.

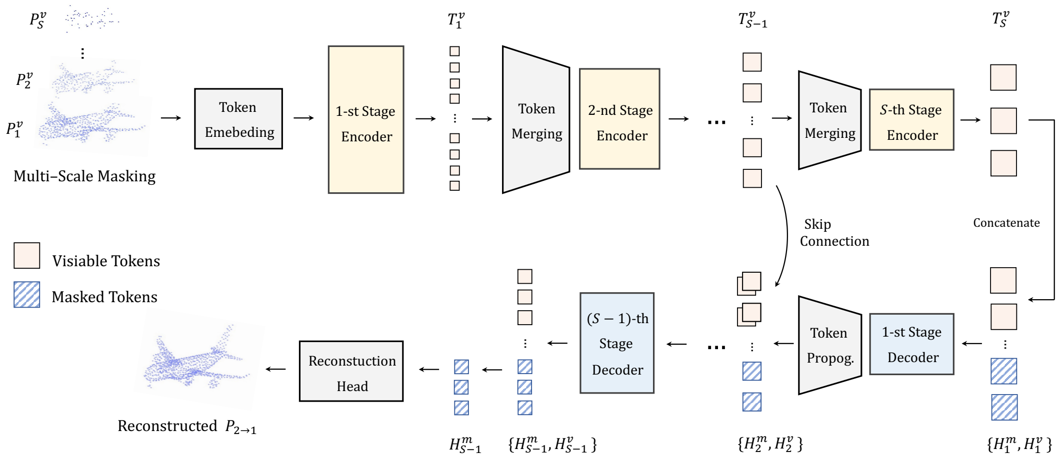

Figure 2: **Overall pipeline of Point-M2AE.** After the multi-scale masking, we embed point tokens at the 1-st scale and feed the visible ones into a hierarchical encoder-decoder transformer, which captures both high-level semantics and fine-grained patterns of the point cloud during pre-training.

3.1 Multi-scale Masking

To build a U-Net [34] like masked autoencoder for hierarchical learning, we encode the point cloud by S scales with different number of points at each scale, and correspondingly modify the standard plain encoder into the S-stage architecture. Following MAE, we embed the point cloud into discrete point tokens and randomly mask them for reconstruction. Importantly, for irregular-distributed points in the multi-scale architecture, the unmasked visible spatial regions are required to be consistent not only within one scale, but also across different scales. This is because the block-wise parts of 3D shapes tend to preserve more complete fine-grained geometries, and the unmasked positions are better to be shared across all scales for coherent feature learning of the encoder. Therefore, as shown in Figure 3, we first construct the S-scale coordinate representations of the input point cloud and back-project the random masks from the final S-th scale to the earlier scales to avoid fragmented visible parts.

S-scale Representations. We denote the input point cloud as $P \in \mathbb{R}^{N\times 3}$ and regard it as the 0-th scale. For the i-th scale, $1 \le i \le S$, we utilize Furthest Point Sampling (FPS) to downsample the points from the $(i-1)$-th scale, which produces seed points $P_i \in \mathbb{R}^{N_i\times 3}$ for scale i of N_i points. Then, we adopt k Nearest-Neighbour (k-NN) to aggregate the neighboring k points for each seed point and obtain the neighbor indices $I_i \in \mathbb{R}^{N_i\times k}$. By successively downsampling and grouping, we acquire the S-scale representations $\{P_i, I_i\}_{i=1}^{S}$ of the input point cloud, where the number of points N_i gradually decreases and the inclusion relations between scales are recorded in I_i.

Back-projecting Visible Positions. For seed points P_S at the final S-th scale, we randomly mask them with a large proportion (e.g., 80%) and denote the remaining visible points as $P_S^v \in \mathbb{R}^{N_S^v\times 3}$ of N_S points. We then back-project the unmasked positions P_S^v to ensure the consistent visible regions across scales. For the i-th scale, $1 \le i < S$, we retrieve all the k nearest neighbors of P_{i+1}^v from the indices I_{i+1} to serve as the visible positions P_i^v, and mask the others. By recursively back-projecting, we obtain the visible and masked positions of all S scales, denoted as $\{P_i^v, P_i^m\}_{i=1}^{S}$, where $P_i^v \in \mathbb{R}^{N_i^v\times 3}$, $P_i^m \in \mathbb{R}^{N_i^m\times 3}$ and $N_i = N_i^v + N_i^m$.

3.2 Hierarchical Encoder

Based on the multi-scale masking, we embed the initial tokens of visible points P_1^v for the 1-st scale and them into the hierarchical encoder with S stages. Every stage is equipped with K stacked encoder blocks, and each block contains a local spatial self-attention layer and a Feed Forward Network (FFN) of MLP layers. Between every two consecutive stages, we introduce spatial token merging modules to aggregate adjacent visible tokens and enlarge receptive fields for downsampling the point clouds.

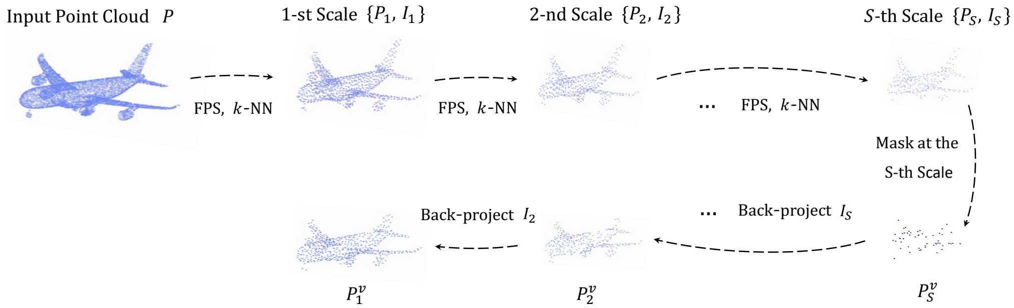

Figure 3: **Multi-scale masking strategy.** To obtain a consistent visible regions across scales, we first represent the input point cloud by multi-scale coordinates and generate the random mask at the highest one. Then, we back-project the unmasked visible positions to all earlier scales.

Token Embedding and Merging. Indexed by I_1, we utilize a mini-PointNet [28] to extract and fuse the features of every seed point from $P_1^v \in \mathbb{R}^{N_1^v \times 3}$ with its k nearest neighbors. After that, we obtain the initial point tokens $T_1^v \in \mathbb{R}^{N_1^v \times C_1}$ for the 1-st stage of the encoder, which embeds N_1^e local patterns of the 3D shape. Between the $(i-1)$-th and i-th stages, $1 < i \leq S$, we merge $T_{i-1}^v \in \mathbb{R}^{N_{i-1} \times C_{i-1}}$ to acquire the downsampled point tokens for the i-th stage. We utilize MLP layers and a max pooling to integrate every k tokens nearest to P_i^v indexed by I_i, which outputs $T_i^v \in \mathbb{R}^{N_i \times C_i}$. Due to our multi-scale masking, the merged T_i^v corresponds to the same visible parts of T_{i-1}^v, which enables the consistent feature encoding across different scales. For larger i of deeper stages, we set higher feature dimension C_i to encode spatial geometries with richer semantics.

Local Spatial Self-Attention. For smaller i of shallower stages, we expect each token to mainly focus on finer-grained information and not to be disturbed by long-range signals. Thus, we modify the original self-attention layer by a local spatial constraint that only neighboring tokens within a ball query [29] would be available for attention calculation. As the point tokens are downsampled by stages, we set increasing radii $\{r_i\}_{i=1}^S$ of multi-scale ball queries for gradually expanding the attention scopes, which fulfills the local-to-global feature aggregation scheme.

3.3 Hierarchical Decoder

Via the hierarchical encoder, we obtain the encoded visible tokens $\{T_i^v\}_{i=1}^S$ of all scales. Starting from the highest S-th scale, we assign a shared learnable mask token to all the masked positions P_S^m, and concatenate them with the visible tokens T_S^v. We denote them as $\{H_1^v, H_1^m\}$ with coordinates $\{P_S^v, P_S^m\}$, which serve as the input of the hierarchical decoder. We design the decoder to be lightweight with $S-1$ stages and only one decoder block for each stage, which enforces the encoder to embed more semantics of the point clouds. Each decoder block consists of a vanilla self-attention layer and an FFN. We do not apply the local constraint to the attention in the decoder, since a global understanding between visible and mask tokens is crucial to the reconstruction.

Point Token Upsampling. We upsample the point tokens between stages to progressively recover the fine-grained geometries of 3D shapes before reconstruction. We regulate that the j-th stage of the decoder corresponds to the $(S+1-j)$-th stage of the encoder, both of which contain point tokens of the same $(S+1-j)$-th scale with the feature dimension C_{S+1-j}. Between the $(j-1)$-th and j-th stage, $1 < j \leq S-1$, we upsample the tokens $\{H_{j-1}^v, H_{j-1}^m\}$ from the coordinates $\{P_{S+2-j}^v, P_{S+2-j}^m\}$ into $\{P_{S+1-j}^v, P_{S+1-j}^m\}$ via the token propagation module. Specifically, we obtain the k nearest neighbors of each point token in $\{H_{j-1}^v, H_{j-1}^m\}$ indexed by I_{S+2-j}, and recover their neighbors' features by weighted interpolation referring to PointNet++ [29], which generates the tokens $\{H_j^v, H_j^m\}$ of the j-th stage.

Skip Connections. To further complement the fine-grained geometries, we channel-wisely concatenate the visible tokens $H_j^v \in \mathbb{R}^{N_{S+1-j} \times C_{S+1-j}}$ of the decoder with $T_{S+1-j}^v \in \mathbb{R}^{N_{S+1-j} \times C_{S+1-j}}$ from the corresponding $(S+1-j)$-th stage of the encoder via skip connections, and adopt a linear projection layer to fuse their features. For the mask tokens H_j^m, we keep them unchanged, since the encoder only contains visible tokens without the masked ones.

Table 1: **Linear evaluation on ModelNet40 [43] by SVM.** We report different self-supervised learning methods and underline the second-best one.

Method	Acc. (%)
3D-GAN [42]	83.3
Latent-GAN [38]	85.7
SO-Net [22]	87.3
FoldingNet [46]	88.4
MAP-VAE [17]	88.4
VIP-GAN [16]	90.2
DGCNN + Jiasaw [36]	90.6
DGCNN + OcCo [39]	90.7
DGCNN + CrossPoint [2]	<u>91.2</u>
Transformer + OcCo [48]	89.6
Point-BERT [48]	87.4
Point-M2AE	**92.9**
Improvement	+1.7

Table 2: **Shape classification on ModelNet40 [43].** '#points' and 'Acc.' denote the number of points for training and the overall accuracy. [S] represents fine-tuning after self-supervised pre-training.

Method	#points	Acc. (%)
PointNet [28]	1k	89.2
PointNet++ [29]	1k	90.5
PointCNN [23]	1k	92.2
[S] SO-Net [22]	5k	92.5
DGCNN [40]	1k	92.9
PCT [15]	1k	93.2
Point Transformer [51]	-	93.7
Transformer [48]	1k	91.4
[S] Transformer + OcCo [48]	1k	92.1
[S] Point-BERT [48]	1k	93.2
[S] Point-BERT	4k	93.4
[S] Point-BERT	8k	93.8
[S] Point-M2AE	**1k**	**94.0**

Point Reconstruction. After $S-1$ stages of the decoder, we acquire $\{H^v_{S-1}, H^m_{S-1}\}$ with coordinates $\{P^v_2, P^m_2\}$ and reconstruct the masked values from the mask tokens H^m_{S-1}. Other than predicting values at the 0-th scale of the input point cloud P, we reconstruct the coordinates of P^m_1, namely, recovering the masked positions of the 1-st scale $P^m_1 \in \mathbb{R}^{N^m_1 \times 3}$ from the 2-nd scale $P^m_2 \in \mathbb{R}^{N^m_2 \times 3}$. This is because $\{P^v_1, P^m_1\}$ of the 1-st scale could well represent the overall 3D shape and simultaneously preserve enough local patterns, which already constructs a comparatively challenging pretext task for pre-training. If we further upsample $\{H^v_{S-1}, H^m_{S-1}\}$ into $\{H^v_S, H^m_S\}$ and reconstruct the masked raw points from P^m_1, the extra spatial noises and computational overhead would adversely influence our performance and efficiency. Therefore, for every token in $H^m_{S-1} \in \mathbb{R}^{N^m_2 \times C_2}$, we reconstruct its k nearest neighbors recorded in I_2 by a reconstruction head of one linear projection layer and compute the loss by l_2 Chamfer Distance [14], formulated as,

$$\widehat{P}^m_{2\to1} = \mathrm{Linear}(H^m_{S-1}), \quad \text{where } \widehat{P}^m_{2\to1} \in \mathbb{R}^{N^m_2 \times k \times 3}, \tag{1}$$

$$\mathcal{L}_{CD} = \mathrm{CharmferDistance}(P^m_{2\to1}, \widehat{P}^m_{2\to1}), \tag{2}$$

where $\widehat{P}^m_{2\to1}$ and $P^m_{2\to1}$ denote the predicted and ground-truth reconstruction coordinates from the 2-nd scale to the 1-st scale. We only utilize $\mathcal{L}_{CD}$ for supervision without contrastive loss to conduct a pure masked autoencoding for self-supervised pre-training.

4 Experiments

In Section 4.1 and Section 4.2, we introduce the pre-training experiments of Point-M2AE and report the fine-tuning performance on various downstream tasks. We also conduct ablation studies in Section 4.3 to validate the effectiveness of our approach.

4.1 Self-supervised Pre-training

Settings. We pre-train our Point-M2AE on ShapeNet [6] dataset, which contains 57,448 synthetic 3D shapes of 55 categories. We set the stage number S as 3, and construct a 3-stage encoder and a 2-stage decoder for hierarchical learning. We adopt 5 blocks in each encoder stage, but only 1 block per stage for the lightweight decoder. For the 3-scale point cloud, we set the point numbers, token dimensions, and radii of the local spatial attention layers respectively as {512, 256, 64}, {96, 192, 384} and {0.32, 0.64, 1.28}. We also set different k for the k-NN at different scales, which are {16, 8, 8}. We mask the highest scale of point clouds with a high ratio of 80% and set 6 heads for all the attention modules. The detailed training settings are in Appendix.

Table 3: **Shape classification on ScanObjectNN [37].** We report the accuracy (%) on the three splits of ScanObjectNN. [S] represents fine-tuning after self-supervised pre-training.

Method	OBJ-BG	OBJ-ONLY	PB-T50-RS
PointNet [28]	73.3	79.2	68.0
PointNet++ [29]	82.3	84.3	77.9
DGCNN [40]	82.8	86.2	78.1
PointCNN [23]	86.1	85.5	78.5
Transformer [48]	79.86	80.55	77.24
[S] Transformer + OcCo [48]	84.85	85.54	78.79
[S] Point-BERT [48]	87.43	88.12	83.07
[S] Point-M2AE	**91.22**	**88.81**	**86.43**
Improvement	+3.79	+0.69	+3.36

Linear SVM. After pre-training on ShapeNet, we test the 3D representation capability of Point-M2AE via linear evaluation on ModelNet40 [43]. We sample 1,024 points from each 3D shape of ModelNet40 and utilize our frozen encoder to extract their features. On top of that, we train a linear SVM and report the classification accuracy in Table 1. As shown, Point-M2AE achieves the best performance among all existing self-supervised methods for point clouds, and surpasses the second-best CrossPoint [2] by +1.7%. Point-M2AE also exceeds Point-BERT [48] by +5.5%, which is a masked point modeling method with a MoCo loss [19] but adopts a standard transformer and conducts single-scale learning. It is worth noting that even if we freeze all our parameters, Point-M2AE with 92.9% accuracy still outperforms many fully trained methods on ModelNet40, e.g., 90.5% by PointNet++ [29], 92.8% by DensePoint [24], etc. The experiments fully demonstrate the superior 3D representation capacity of our Point-M2AE.

4.2 Downstream Tasks

For fine-tuning on downstream tasks, we discard the hierarchical decoder in pre-training and append different heads onto the hierarchical encoder for different tasks.

Shape Classification. We fine-tune Point-M2AE on two shape classification datasets: the widely adopted ModelNet40 [43] and the challenging ScanObjectNN [37]. We follow Point-BERT to use the voting strategy [25] for fair comparison on ModelNet40, which tests the model for several times with different point cloud augmentation and ensembles the predictions. To handle the noisy spatial structures, we increase k of k-NN into {32, 16, 16} for ScanObjectNN to encode local patterns with larger receptive fields. As reported in Table 2, Point-M2AE achieves 94.0% accuracy on ModelNet40 with 1024 points per sample, which surpasses Point-BERT fine-tuned with 1024 points by +0.8% and 8192 points by +0.2%. For ScanObjectNN in Table 3, our Point-M2AE outperforms the second-best Point-BERT by a significant margin, +3.79%, +0.69% and +3.36%, respectively for the three splits, indicating our great advantages under complex circumstances by multi-scale encoding. As ScanObjectNN of real-world scenes has a large semantic gap with the pre-trained synthetic ShapeNet, Point-M2AE also exerts strong transfer ability to understand point clouds of another domain.

Part Segmentation. We evaluate Point-M2AE for part segmentation on ShapeNetPart [47], which predicts per-point part labels and requires detailed understanding for local patterns. We adopt an extremely simple segmentation head to validate the effectiveness of our pre-training for well capturing both high-level semantics and fine-grained details. By the hierarchical encoder, we obtain 3-scale point tokens of {512, 256, 64} points, and perform feature propagation in PointNet++ [29] to independently upsample the tokens into 2048 points of the input point cloud. Then, we concatenate the upsampled 3-scale features for each point and predict the part label by stacked linear projection layers. As reported in Table 4.2, Point-M2AE achieves the best 86.51% instance mIoU with the simple segmentation head, surpassing the second-best Point-BERT by +0.91%. Note that Point-BERT [48] and other methods [28, 29, 40] adopt hierarchical segmentation heads to progressively upsample the

Table 4: **Few-shot classification on ModelNet40 [43]**. We report the average accuracy (%) and standard deviation (%) of 10 independent experiments.

Method	5-way		10-way	
	10-shot	20-shot	10-shot	20-shot
DGCNN [40]	91.8 ± 3.7	93.4 ± 3.2	86.3 ± 6.2	90.9 ± 5.1
[S] DGCNN + OcCo [39]	91.9 ± 3.3	93.9 ± 3.1	86.4 ± 5.4	91.3 ± 4.6
Transformer [48]	87.8 ± 5.2	93.3 ± 4.3	84.6 ± 5.5	89.4 ± 6.3
[S] Transformer + OcCo [48]	94.0 ± 3.6	95.9 ± 2.3	89.4 ± 5.1	92.4 ± 4.6
[S] Point-BERT [48]	94.6 ± 3.1	96.3 ± 2.7	91.0 ± 5.4	92.7 ± 5.1
[S] Point-M2AE	**96.8 ± 1.8**	**98.3 ± 1.4**	**92.3 ± 4.5**	**95.0 ± 3.0**
Improvement	+2.2	+2.0	+1.3	+2.3

Table 5: **Part segmentation on ShapeNetPart [47]**. 'mIoU_C' (%) and 'mIoU_I' (%) denote the mean IoU across all part categories and all instances in the dataset, respectively.

Method	mIoU_C	mIoU_I
PointNet [28]	80.39	83.70
PointNet++ [29]	81.85	85.10
DGCNN [40]	82.33	85.20
Transformer [48]	83.42	85.10
[S] Transformer + OcCo [48]	83.42	85.10
[S] Point-BERT [48]	84.11	85.60
[S] Point-M2AE	**84.86**	**86.51**
Improvement	+0.75	+0.91

Table 6: **3D object detection on ScanNetV2 [9].** We report the performance (%) of self-supervised learning methods based on VoteNet [12] and 3DETR-m [26].

Method	AP_{25}	AP_{50}
VoteNet [12]	58.6	33.5
[S] STRL [20]	59.5	38.4
[S] PointContrast [44]	59.2	38.0
[S] DepthContrast [50]	61.3	–
3DETR [26]	62.1	37.9
3DETR-m [26]	65.0	47.0
[S] Point-M2AE	**66.3**	**48.3**
Improvement	+1.3	+1.3

point features from intermediate layers, while our head contains no hierarchical structure and only relies on the pre-trained encoder to capture the multi-scale information of point clouds. The results fully demonstrate the significance of Point-M2AE's multi-scale pre-training to segmentation tasks.

Few-shot Classification. We conduct experiments for few-shot classification on ModelNet40 [43] to evaluate the performance of Point-M2AE with limited fine-tuning data. As reported in Table 4.2, Point-M2AE achieves the best performance for all four settings, and surpasses Point-BERT by +2.2%, +2.0%, +1.3%, and +2.7%, respectively. Our approach also shows smaller deviations than other transformer-based methods, which indicates Point-M2AE has learned to produce more universal 3D representations for well adapting to downstream tasks under low-data regimes.

3D Object Detection To further evaluate our hierarchical pre-training on 3D object detection, we apply Point-M2AE to serving as the feature backbone on the indoor ScanNetV2 [9] dataset. We select 3DETR-m [26] as our baseline, which consists of a 3-block encoder and a transformer decoder. Considering the quite different dataset statistics, e.g., 2k input points for ShapeNet [6] and 50k input points for ScanNetV2, we adopt the same encoder architecture with that of 3DETR-m, and keep our hierarchical decoder with skip connections unchanged for self-supervised pre-training on ScanNetV2. More details of models and training are in Appendix. As reported in Table 4.2, compared to training from scratch, our hierarchical pre-training boosts the performance of 3DETR-m by +1.34% AP_{25} and +1.29% AP_{50}. The experiments demonstrate the effectiveness of Point-M2AE to learn multi-scale point cloud encoding for object detection and its potential to benefit a wider range of 3D applications.

4.3 Ablation Study

We conduct ablation study by modifying one of the components at a time to test their effectiveness and explore the best masking strategy for self-supervised pre-training. We report the classification accuracy on ModelNet40 [43] by linear SVM to evaluate the pre-trained representations. For

Table 7: **Effectiveness of Hierarchical Modules.** 'H' represents the encoder and decoder with multi-stage hierarchies. 'Skip C.' and 'Local SA' denote the skip connections and local spatial attention layers, respectively.

Encoder	Decoder	Skip C.	Local SA	Acc. (%)
H	H	✓	✓	**92.9**
-	-	✓	✓	90.7
-	H	✓	✓	91.5
H	-	✓	✓	92.2
H	H	-	✓	92.1
H	H	✓	-	92.3

Table 8: **Different Masking Strategy.** 'MS Mask' and 'Ratio' denote the multi-scale masking and the mask ratio.

MS Mask	Ratio	Acc. (%)
✓	0.8	**92.9**
-	0.8	88.4
✓	0.5	92.1
✓	0.6	92.3
✓	0.7	92.7
✓	0.9	92.5

downstream tasks, we compare the performance between fine-tuning and training from scratch to validate the significance of our hierarchical pre-training.

Hierarchical Modules. As reported in Table 7, on top of our final solution of Point-M2AE in the first row, we respectively experiment with removing the hierarchical encoder, hierarchical decoder, skip connections, and local spatial self-attention layers from our framework. Specifically, we replace our encoder and decoder with 1-stage plain architectures similar to MAE, which contains 15 and 2 blocks of vanilla self-attention layers, respectively. We observe the absence of multi-stage structures either in encoder or decoder would hurt the performance, and the hierarchical encoder plays a better role than the decoder. Also, the skip connections and local spatial attention can well benefit the network by providing complementary information and local inductive bias.

Masking Strategy. In Table 8, we report Point-M2AE with different mask settings. Without the multi-scale masking, we randomly generate masks at each scale, which leads to fragmented visible regions for all scales. With this strategy, the network would 'peek' different parts of the point cloud at different stages, which disturbs the representation learning and harms the performance by 4.5% accuracy. For different mask ratios, we find the 80% ratio performs the best to build a properly challenging pretext task for self-supervised pre-training.

With and without Pre-training. We report Point-M2AE on downstream tasks with and without the pre-training in Table 9. For 'w/o', we randomly initialize our network and adopt the same training settings with fine-tuning. As shown, the hierarchical pre-training can largely boost the performance on four datasets respectively by +1.5%, +2.5%, +3.8%, and +1.1%, indicating the significance of our pre-training scheme.

Table 9: **With and without pre-training.** 'ModelNet40-FS' denotes the few-shot classification on 10-way 20-shot ModelNet40 [43].

Dataset	w/o (%)	w (%)
ModelNet40 [43]	92.5	94.0
ScanObjectNN [37]	83.9	86.4
ModelNet40-FS [43]	91.2	95.0
ShapeNetPart [47]	85.4	86.5

5 Conclusion

We propose Point-M2AE, a multi-scale masked autoencoder for self-supervised pre-training on 3D point clouds. With a hierarchical architecture, Point-M2AE learns to produce powerful 3D representations by encoding multi-scale point clouds and reconstructing the masked coordinates from a global-to-local upsampling scheme. Extensive experiments have shown the *state-of-the-art* performance of Point-M2AE on downstream tasks and our superiority to be a strong 3D representation learner. **Limitations.** Although we have experimented Point-M2AE on various 3D tasks, its performance on open-world 3D object detection and scene segmentation has yet not been discussed. Our future work will focus on this direction to apply Point-M2AE for wider 3D applications. **Societal Impact.** We do not foresee negative social impact from the proposed work.

References

[1] Idan Achituve, Haggai Maron, and Gal Chechik. Self-supervised learning for domain adaptation on point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 123–133, 2021. 3

[2] Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. *arXiv preprint arXiv:2203.00680*, 2022. 2, 3, 6, 7

[3] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv preprint arXiv:2202.03555*, 2022. 1, 3

[4] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 3

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1, 3

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2, 6, 8

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1, 3

[8] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 1, 3

[9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias 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. 3, 8

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 3

[12] Zhipeng Ding, Xu Han, and Marc Niethammer. Votenet: A deep learning label fusion method for multi-atlas segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 202–210. Springer, 2019. 8

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1, 3

[14] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 6

[15] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7(2):187–199, 2021. 6

[16] Zhizhong Han, Mingyang Shang, Yu-Shen Liu, and Matthias Zwicker. View inter-prediction gan: Unsupervised representation learning for 3d shapes by learning global shape memories to support local view predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8376–8384, 2019. 6

[17] Zhizhong Han, Xiyang Wang, Yu-Shen Liu, and Matthias Zwicker. Multi-angle point cloud-vae: Unsupervised feature learning for 3d point clouds from multiple angles by joint self-reconstruction and half-to-half prediction. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10441–10450. IEEE, 2019. 6

[18] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021. 1, 3
[19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 1, 3, 7
[20] Siyuan Huang, Yichen Xie, Song-Chun Zhu, and Yixin 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. 8
[21] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021. 1
[22] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9397–9406, 2018. 6
[23] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31:820–830, 2018. 6, 7
[24] Yongcheng Liu, Bin Fan, Gaofeng Meng, Jiwen Lu, Shiming Xiang, and Chunhong Pan. Densepoint: Learning densely contextual representation for efficient point cloud processing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5239–5248, 2019. 7
[25] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019. 7
[26] Ishan Misra, Rohit Girdhar, and Armand Joulin. An end-to-end transformer model for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2906–2917, October 2021. 8
[27] Omid Poursaeed, Tianxing Jiang, Han Qiao, Nayun Xu, and Vladimir G Kim. Self-supervised learning of point clouds via orientation estimation. In *2020 International Conference on 3D Vision (3DV)*, pages 1018–1028. IEEE, 2020. 3
[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 5, 6, 7, 8
[29] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 5, 6, 7, 8
[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1
[31] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018. 1, 3
[32] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 1, 3
[33] Jason Tyler Rolfe. Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*, 2016. 3
[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2, 4
[35] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *Advances in Neural Information Processing Systems*, 32, 2019. 3
[36] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. *Advances in Neural Information Processing Systems*, 32, 2019. 6
[37] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*,

pages 1588–1597, 2019. 2, 7, 9
[38] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized representations of point clouds with graph-convolutional generative adversarial networks. *IEEE Transactions on Multimedia*, 23:402–414, 2020. 6
[39] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt 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. 3, 6, 8
[40] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 6, 7, 8
[41] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. *arXiv preprint arXiv:2112.09133*, 2021. 3
[42] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29, 2016. 6
[43] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 2, 6, 7, 8, 9
[44] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *European conference on computer vision*, pages 574–591. Springer, 2020. 3, 8
[45] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. *arXiv preprint arXiv:2111.09886*, 2021. 1
[46] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 206–215, 2018. 6
[47] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016. 2, 7, 8, 9
[48] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. *arXiv preprint arXiv:2111.14819*, 2021. 3, 6, 7, 8
[49] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. *arXiv preprint arXiv:2112.02413*, 2021. 1
[50] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan 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. 3, 8
[51] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021. 6
[52] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021. 1, 3

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]