Self-supervised Pre-training with Masked Shape Prediction for 3D Scene Understanding

Li Jiang[1] Zetong Yang[2] Shaoshuai Shi[1] Vladislav Golyanik[1] Dengxin Dai[1] Bernt Schiele[1]
[1]Max Planck Institute for Informatics, Saarland Informatics Campus [2]CUHK
{lijiang, sshi, golyanik, ddai, schiele}@mpi-inf.mpg.de tomztyang@gmail.com

Abstract

Masked signal modeling has greatly advanced self-supervised pre-training for language and 2D images. However, it is still not fully explored in 3D scene understanding. Thus, this paper introduces Masked Shape Prediction (MSP), a new framework to conduct masked signal modeling in 3D scenes. MSP uses the essential 3D semantic cue, i.e., geometric shape, as the prediction target for masked points. The context-enhanced shape target consisting of explicit shape context and implicit deep shape feature is proposed to facilitate exploiting contextual cues in shape prediction. Meanwhile, the pre-training architecture in MSP is carefully designed to alleviate the masked shape leakage from point coordinates. Experiments on multiple 3D understanding tasks on both indoor and outdoor datasets demonstrate the effectiveness of MSP in learning good feature representations to consistently boost downstream performance.

1. Introduction

Self-supervised pre-training has witnessed considerable progress in natural language processing (NLP) [4, 10, 42] and 2D computer vision [2, 15, 17, 18], the main idea of which is to define a pretext task to leverage unlabeled data to learn meaningful representations. With the development of transformer [11,30,59], masked signal modeling (MSM) has been proved to be an effective pretext task, attaining better results than other tasks like contrastive learning [6, 18]. An MSM architecture first partially masks out the input and then reconstructs the masked part given the remaining content, forcing the network to learn semantic knowledge for completing the missing part.

Compared to 2D images, the labeling of 3D real-scene data is more labor-intensive. Therefore, self-supervised pre-training is important in 3D scene understanding for its ability in boosting the performance with limited labeled data. Previous 3D scene-level pre-training methods mostly follow the contrastive pipeline [20, 21, 43, 64]. Though effective, MSM is less explored in 3D scene level. Some recent methods [36, 74] also explore MSM with point clouds but focus on single-object-level understanding. In contrast, we investigate MSM for more practical scene-level understanding that contains complicated contextual environments, and we propose a Masked Shape Prediction (MSP) framework to conduct pre-training on point cloud scenes.

There are several key problems when performing masked signal modeling in 3D scenes. The first is the design of the reconstruction target. In 2D images, pixel colors constitute the semantic contents, making appearance signals [17,62] good choices as targets. In 3D, the most essential semantic clue is geometric shape, which motivates us to explore shape information in target design. In 3D scene-level understanding with complex object distribution, broad contextual information is essential in achieving outstanding performance. Therefore, to promote the network to exploit contextual cues in shape prediction, we propose the context-enhanced shape target, which includes two components: *shape context* and *deep shape feature*. Shape context explicitly describes the 3D shape by discretizing the local space into multiple bins, which is robust to the uneven point distributions. Deep shape feature is extracted from point clouds with complete shapes by a deep network. As a learned shape descriptor, deep shape feature is able to adaptively integrate contextual information in a larger range, thanks to the large receptive field of the deep network. By combining shape context and deep shape feature as our context-enhanced shape target, the network is promoted to not only focus on explicit shape patterns, but also on contextual object relations in a larger scope.

Using the geometric shapes as reconstruction target, however, raises another problem. Shape information can be inferred from the point coordinates, yet masked signal modeling requires the coordinates of masked points to specify the target positions for reconstruction, which may reveal the masked shape and thus create a shortcut for network learning. In this paper, we discuss several MSP network designs to prevent the masked shape from being revealed by the masked point coordinates. The core idea is to either avoid the information interactions between masked points or restrict the interactions to sparsely sampled keypoints.

We follow [20, 64] to perform unsupervised pre-training on ScanNet v2 [9] indoor scene dataset, and then evaluate it via supervised fine-tuning in different downstream tasks. Our MSP extracts representative 3D features that are beneficial in indoor scene understanding tasks on multiple datasets [1, 9, 54], achieving excellent performance in both segmentation and detection and showing great ability in data-efficient learning. We also evaluate its transferring ability to outdoor scenes. Our core technical contributions are listed below:

- We propose a self-supervised pre-training method for 3D scene understanding, namely, Masked Shape Prediction (MSP), which consistently boosts the downstream performance.
- We present the context-enhanced shape target, combining the strengths of explicit shape context descriptor and implicit deep shape feature.
- We explore different MSP network architecture designs to promote feature learning and mitigate the masked shape leakage problem.

2. Related Work

3D Point Cloud Understanding. 3D point cloud understanding tasks have been widely explored in recent years, including detection [31, 37, 38, 51, 53, 73], segmentation [12, 16, 19, 22, 23, 27, 35, 56, 68, 70, 77] and classification [39, 58, 61, 63]. Two major data representations used in these methods are points and voxels. Point-based methods [40, 61] take raw points as input, which reserve the precise position information. Voxel-based methods apply sparse convolutions [8, 14] on voxelized 3D data, capable of processing large-scale point clouds efficiently. Recent methods propose transformer [59] backbones, but are limited to specific tasks, *e.g.*, Point Transformer [78] for segmentation and classification, and Voxel Transformer [33] for detection. [71] proposes an embedding-querying paradigm, enabling a general transformer-based backbone network on various tasks. We apply the EQ-Net in [71] as the feature extractor in our pre-training framework, which also serves as the backbone network in downstream tasks.

3D Shape Descriptor. Geometric shape is an important signal in 3D point cloud understanding. There are several 3D shape descriptors, such as shape context [3, 24], point feature histogram [47], and fast point feature histogram [46]. In this work, we adopt shape context to describe shape due to its intuitive formulation and robustness to noise.

Self-supervised Pre-training. Self-supervised pre-training has achieved great success in NLP [4, 10, 41, 42] and 2D computer vision [6, 15, 17, 18, 62, 79]. Recently, its effectiveness has also been verified in 3D domain [25, 26, 44, 48, 49, 60, 64, 72] with diverse pretext tasks. Among them, [20, 21, 43, 64, 75] attempt to extend the contrastive learning scheme to 3D with different designs in feature pair construction. OcCo [60] proposes an encoder-decoder framework to complete the occluded points. IAE [67] applies an autoencoder to reconstruct implicit representations of point clouds. [49] solves the jigsaw puzzles by reconstructing shapes with randomly arranged parts.

Masked Signal Modeling. Inspired by the success of masked signal modeling for self-supervised pre-training in NLP [10] and 2D [2, 17], Point-BERT [74] proposes the masked point modeling, using dVAE [45] tokens as prediction targets. Some recent works [29, 36] also investigate the masked point modeling by applying an autoencoder structure or a discriminative decoder. These methods perform pre-training on ShapeNet [5] and mainly put their attention on single-object-level understanding. This paper also follows the line of masked signal modeling but focuses on the 3D scene understanding.

3. Method

We start by giving an overview of our masked shape prediction (MSP) in Sec. 3.1. We then introduce our context-enhanced shape target in Sec. 3.2. Our exploration of the pre-training MSP network design is presented in Sec. 3.3, with a focus on mitigating masked shape leakage.

3.1. Overview of Masked Shape Prediction

Pretext Task Definition. For pre-training the network without labels, we follow the works in NLP [10] and 2D [2, 17, 62] to perform the masked signal modeling task, which masks out a portion of the input data and then reconstructs the features of the removed parts based on the remaining contents. Appearance signals like pixel colors [17, 66] are effective reconstruction targets in masked image modeling. However, unlike 2D vision in which appearance information contributes most to the semantic understanding, 3D vision with point clouds relies more on geometric shape for semantic reasoning. Hence, we define our 3D pre-training task as masked shape prediction (MSP), with geometric shapes as reconstruction targets. Specifically, we denote the masked and remaining points as $\mathbf{P}_m$ and $\mathbf{P}_r$, respectively. Our task is to reconstruct the shape features $\mathbf{F}_m$ of $\mathbf{P}_m$ given $\mathbf{P}_r$. The overall pipeline is shown in Fig. 1(a). We first extract point features from $\mathbf{P}_r$, and then design an MSP network to build the interaction between masked and unmasked points, which is finally utilized to predict the shape features for the masked parts.

Masking Strategy. Similar to 2D works [2, 17], we divide the 3D space into blocks of side length w. Since 3D data is sparse, we only consider non-empty blocks with interior points. We randomly sample some non-empty blocks with

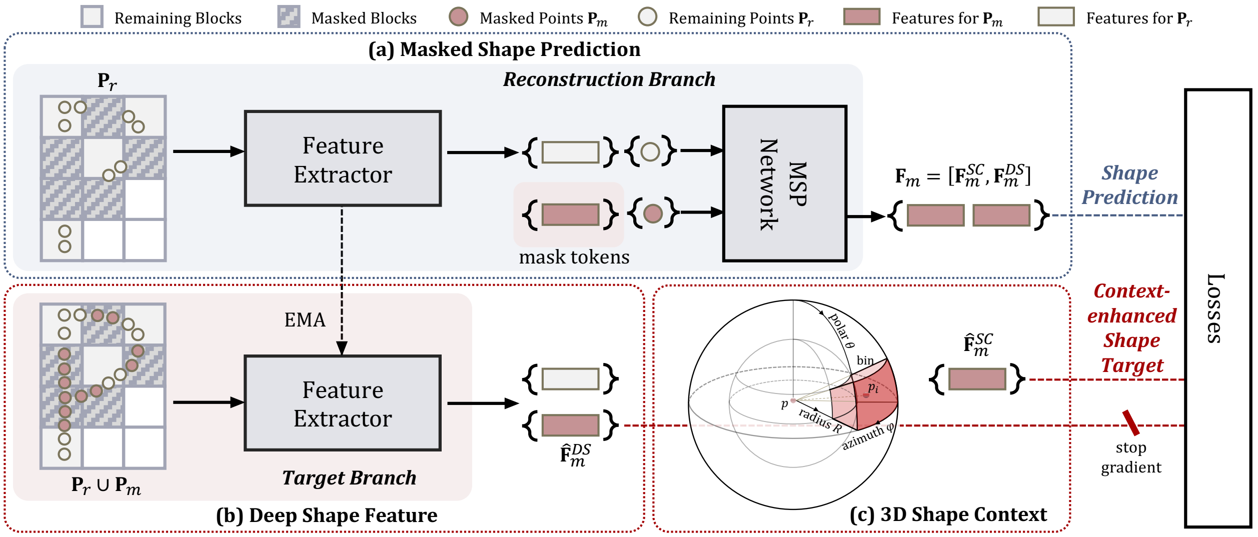

Remaining Blocks
Masked Blocks
Masked Points $\mathbf{P}_m$
Remaining Points $\mathbf{P}_r$
Features for $\mathbf{P}_m$
Features for $\mathbf{P}_r$
(a) Masked Shape Prediction
Reconstruction Branch
$\mathbf{P}_r$
Feature Extractor
mask tokens
MSP Network
$\mathbf{F}_m = [\mathbf{F}_m^{SC}, \mathbf{F}_m^{DS}]$
Shape Prediction
Losses
Context-enhanced Shape Target
stop gradient
EMA
Feature Extractor
Target Branch
$\mathbf{P}_r \cup \mathbf{P}_m$
$\hat{\mathbf{F}}_m^{DS}$
(b) Deep Shape Feature
polar θ
bin
p_i
p
radius R
azimuth φ
$\hat{\mathbf{F}}_m^{SC}$
(c) 3D Shape Context

Figure 1. **Illustration of the masked shape prediction (MSP) pipeline and the context-enhanced shape target.** The reconstruction branch takes the remaining points $\mathbf{P}_r$ as input and predicts shape features using our well-designed context-enhanced shape target as supervision, which has two components: the deep shape feature $\hat{\mathbf{F}}_m^{DS}$ and the 3D shape context feature $\hat{\mathbf{F}}_m^{SC}$. $\hat{\mathbf{F}}_m^{SC}$ explicitly encodes the local geometric shape around a center point. $\hat{\mathbf{F}}_m^{DS}$ is produced by the target branch with a complete point cloud as the input. The reconstruction and target branches share the same feature extractor architecture.

a ratio r and drop all points in them. By adjusting the block size w and masking ratio r, we can control the difficulty of masked shape prediction. Since the network is expected to attain semantic understanding of the scene by completing the missing part, properly setting the masking ratio and the block size is important, as discussed in Sec. 4.4.

3.2. Context-enhanced Shape Target

The central problem in MSP is how to represent the geometric shape of the masked parts to effectively guide the network pre-training. Since the contextual information is important in scene-level understanding, we propose the context-enhanced shape reconstruction target to promote the network to extract semantically rich information. The context-enhanced shape target consists of two elements: **shape context** which explicitly describes the local geometric shape around the masked points, and **deep shape feature** which adaptively encodes the surrounding contextual information and represents the shape implicitly.

Shape Context. Shape context is a traditional feature descriptor that well presents the local shape structure; it was first introduced for 2D shape matching [3] and extended to 3D in [24, 65]. For computing the shape context feature for a center point p, as shown in Fig. 1(c), we split the ball of radius R centered at p into several bins by partitioning it along the polar angle θ, azimuth angle φ, and radius R. We evenly divide polar and azimuth angles into n_θ and n_φ sectors, respectively, while for radius R, we partition it into n_r sectors in a spatially-increasing way, where the radius sectors for inner bins are smaller so that the inner shape are described in a more detailed manner. Specifically, for a neighboring point p_i in the ball with relative distance d_i, polar angle θ_i and azimuth angle φ_i to p, we calculate its bin index b_i as

$$\begin{aligned} b_i^\theta &= \left\lfloor \frac{\theta_i}{\pi} \cdot n_\theta \right\rfloor, \; b_i^\varphi = \left\lfloor \frac{\varphi_i}{2\pi} \cdot n_\varphi \right\rfloor, \\ b_i^r &= \left\lfloor \frac{\log(d_i + \xi) - \log(\xi)}{\log(R + \xi) - \log(\xi)} \cdot n_r \right\rfloor, \; b_i = (b_i^\theta, b_i^\varphi, b_i^r), \end{aligned} \tag{1}$$

where ξ is a hyperparameter to control the spatial variance of radius partition. In this way, we allocate each neighboring point of p to the bin it falls inside.

Different from the original shape context feature that describes the point counts in each bin, considering that the point cloud in a real scene is usually inhomogeneous, we set the bin value to one if any point exists in the bin and to zero if there is no point. The shape context feature of size $n_\theta \times n_\varphi \times n_r$ thus robustly describes the geometric shape around point p. We adopt a multi-scale setting for the sector numbers n_θ, n_φ and n_r. Two partitions, $\{2, 4, 3\}$ and $\{4, 8, 5\}$, are adopted jointly to represent the coarser shape and finer detail in the local ball. We denote the ground-truth shape context features of the masked points as $\hat{\mathbf{F}}_m^{SC}$.

Deep Shape Feature. Besides shape context, we adopt another shape representation to further enhance the contextual information for describing the shape. For this shape representation, inspired by BYOL [15], we adopt a two-branch structure, as shown in Fig. 1(b). The target branch takes the whole point cloud with full shape information as input, whose network parameters are updated as the exponential moving average (EMA) of the feature extractor weights in the reconstruction branch. The target branch accepts com-

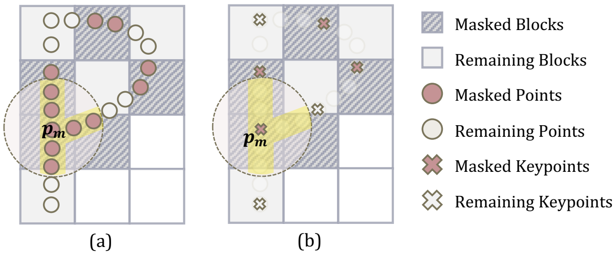


Figure 2. Illustration of the masked shape leakage under (a) original points and (b) subsampled points. The local shape around the masked point p_m is shown in yellow, which can be easily inferred from the nearby points of p_m in (a) and largely kept secret in (b).

plete shape input, thus is expected to produce features with a comprehensive understanding of the contextual shape. The masked parts of the produced features are then taken as the targets for the predicted masked shape features in the reconstruction branch. We denote the deep shape feature target as $\hat{\mathbf{F}}_m^{DS}$. Compared to explicit shape context descriptor, the deep shape feature extracted by the network has a larger receptive field and thus enables a relatively more global understanding of the contextual information in a 3D scene. Also, the feature extractor in the target branch are adaptively updated in the training process to extract more representative features for different shapes.

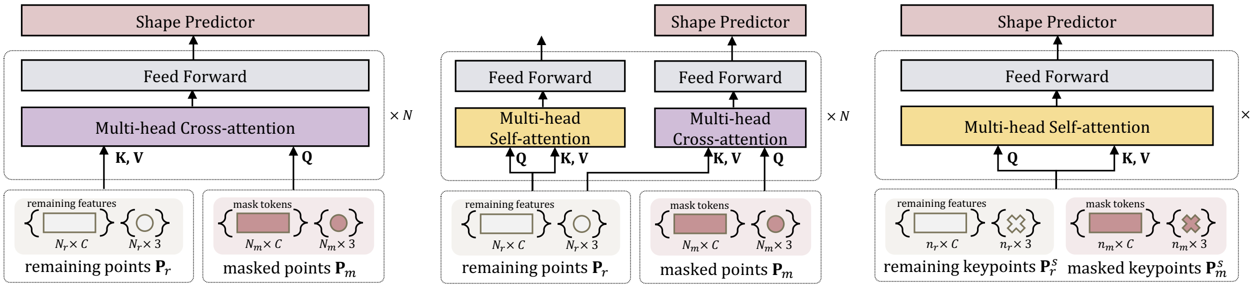
Loss Function. We combine the shape context and the deep shape feature as the context-enhanced shape target to supervise the shape prediction, achieving better performance than using them alone (see Table 6). Specifically, as shown in Fig. 1(a), the MSP network predicts the shape features $\mathbf{F}_m = [\mathbf{F}_m^{SC}, \mathbf{F}_m^{DS}]$. The shape context prediction $\mathbf{F}_m^{SC}$ is optimized towards $\hat{\mathbf{F}}_m^{SC}$ with the binary cross-entropy loss. For the deep shape feature, we adopt the cosine similarity loss to minimize the distance between predicted $\mathbf{F}_m^{DS}$ and target features $\hat{\mathbf{F}}_m^{DS}$. Additionally, we take color prediction as an auxiliary task and optimize it with mean squared error (MSE) loss, which can further slightly boost the performance, as shown in Table 6. The final loss is the sum of the above losses with equal loss weights of 1.0.

3.3. MSP Network: Discussion of the Masked Shape Leakage Problem

Masked Shape Leakage. In the pre-training methods with masked signal modeling [2, 10, 17, 62], the positions of the masked parts (*e.g.*, the pixel indices) are required to indicate the locations for feature prediction. In 3D situation, these positions are the masked point coordinates $\mathbf{P}_m$. However, the shape knowledge of masked parts is also contained in these point coordinates. Hence, when using shape targets, a potential problem is that these masked point coordinates may leak the target information. For example, in Fig. 2(a), p_m is a point in masked blocks, whose surrounding shape is to be predicted in MSP. However, the local geometric shape of p_m (shown in yellow) is implied by the positions of the nearby points around p_m (*i.e.*, points in the circle centered at p_m). So if the masked points around p_m (*i.e.*, red points in the circle centered at p_m) are known in the shape prediction process of p_m—which is a usual case in masked signal modeling that uses self-attention layers to build interaction among points—the masked shape around p_m, *i.e.*, the shape prediction target, may be revealed by these masked points' coordinates. We follow previous masked signal modeling works [2, 17, 74] to use the transformer structure—which shows great effectiveness in building connections between remaining and masked parts—in our MSP network design, while taking the masked shape leakage problem into consideration, as discussed in the following paragraphs.

MSP-CA & MSP-CA++. To mitigate the masked shape leakage, an intuitive strategy is to avoid information interaction between masked points. For this purpose, we propose the MSP-CA architecture (Fig. 3(a)) with cross-attentions to generate shape features for masked points based on the remaining point features. The cross-attentions in MSP-CA are QKV-based multi-head attention layers [59] with $\mathbf{P}_m$ as queries and $\mathbf{P}_r$ as keys. Since there is no information interaction between masked points in MSP-CA, the feature of each masked point is extracted independently. Hence, for a specific masked point, the masked parts of its local geometric structure will not be revealed by other masked points. To improve the feature interaction among points, as shown in Fig. 3(b), we further propose an advanced version of MSP-CA, *i.e.*, MSP-CA++, which applies self-attentions on $\mathbf{P}_r$ to enhance the connections among remaining parts and refine the remaining point features. Although avoiding masked shape leakage, the lack of communication between masked points in MSP-CA and MSP-CA++ may hinder the shape reasoning for masked points far from the remaining parts, since the masked points close to the remaining points can serve as bridges to propagate information from remaining points to distant masked points. Therefore, we propose another architecture, MSP-SA (Fig. 3(c)).

MSP-SA. In this architecture, we enable the interaction between masked points to avoid the aforementioned issue of distant masked points, but restrict the interaction to only sparsely sampled keypoints to alleviate masked shape leakage. Specifically, after using the feature extractor to generate features for remaining points $\mathbf{P}_r$, we randomly sample a subset of keypoints $\mathbf{P}^s$ from the whole point cloud $\mathbf{P}$, and denote the masked and remaining points in the subset as $\mathbf{P}_m^s$ and $\mathbf{P}_r^s$, respectively. As shown in Fig. 3(c), MSP-SA adopts a standard transformer encoder structure [59] with iterative self-attention and feed-forward layers to build connection among all the keypoints in $\mathbf{P}^s$. Note that the remaining points are also sparsely sampled as $\mathbf{P}_r^s$ to maintain balanced sparsity with $\mathbf{P}_m^s$, which facilitates the shape reasoning by shortening the information propagation path



(a) MSP-CA

(b) MSP-CA++

(c) MSP-SA

Figure 3. **MSP Network Architectures.** (a) **MSP-CA** and (b) **MSP-CA++** apply cross-attentions to query shape features for masked points from only the remaining point features, thus avoiding the masked shape leakage in shape prediction. MSP-CA++ further adopts self-attention layers to refine the remaining point features. (c) **MSP-SA** performs self-attentions on the entire scene to enhance the information propagation, but restricts the interaction to subsampled keypoints to mitigate shape target leakage. N_r, N_m, n_r, and n_m denote the point numbers of $\mathbf{P}_r$, $\mathbf{P}_m$, $\mathbf{P}_r^s$, and $\mathbf{P}_m^s$, respectively. C denotes the feature channel number. The attention and feed-forward layers are both followed by normalization layers and residual additions (as in [59]), which are ignored in the figure for clarity.

between two distant parts. As shown in Fig. 2(b), with information propagation between only sparsely sampled keypoints, the masked parts of p_m's surrounding shape (shown in yellow) will not be severely leaked by p_m's surrounding points as most of them are dropped out in the subsampling and kept unknown in p_m's local shape prediction process.

In general, MSP-CA, MSP-CA++, and MSP-SA are all able to learn semantically-meaningful latent features, yet MSP-SA with a preferable information propagation manner is a more effective architecture when the sampling number is properly set. Sec. 4.4 shows the experimental comparison of these MSP network designs.

Network Details. As the MSP network input, the features of masked points are initialized as learnable mask tokens as in 2D works [2, 17]. The shape predictor is a linear layer to produce final shape predictions. Considering the network efficiency in processing scene-level point clouds, we follow [71,78] to implement the attention layers in a local way, in which k nearest neighbor points are searched for each query for attention calculation. Similar to the autoencoder structure in MAE [17] for 2D vision, we take the feature extractor as an encoder for visible parts and the MSP network as a decoder for shape reconstruction. The MSP network is only used in pre-trainining and discarded in the downstream tasks. We adopt EQ-Net [71], a scene-level transformer-based network, as the feature extractor, which also serves as a unified and strong backbone in various downstream tasks.

4. Experiments

4.1. Pre-training Setups

Data Setups. We pre-train our model on ScanNet v2 [9], which contains 1613 indoor 3D scenes created from RGB-D sequences with 2.5M views, and we use the training split for pre-training. The data augmentations include point jittering, flipping, rotation, and elastic transformation [14].

Network Architecture. We use the Embedding-Querying Network (EQ-Net) in [71] as the feature extractor in pre-training and the backbone in downstream tasks. Unlike [71] which applies different embedding and querying architectures in different 3D tasks, we unify the network structure so that we can apply the pre-trained weights to different downstream tasks. Specifically, we take SparseConvNet [8, 14] as embedding network and the transformer-based Q-Net [71] as the querying network. The dimension of the output feature of EQ-Net is set to 576. For MSP network, we set the number of attention blocks to 6, and the number of heads to 12. The neighborhood number k in local attention is 32.

Implementation and Training Details. We set R and ξ in shape context to 0.15 and 0.3, respectively. For the deep shape feature, we set the target decay rate for EMA update to 0.999. We set the ratio r and block size w for masking to 60% and 0.3m, respectively. For MSP-SA, we randomly sample 10k keypoints. AdamW [32] is adopted as the optimizer with a weight decay of 0.1. We train for 600 epochs with a batch size of 8 with four GPUs.

4.2. Fine-tuning in Downstream Tasks

We evaluate the pre-trained representations on various downstream tasks in a supervised way, including semantic segmentation, indoor and outdoor object detection. The MSP network structure is MSP-SA by default.

Semantic Segmentation. We perform tests on two real-world datasets, ScanNet v2 and S3DIS [1]. ScanNet v2 contains 20 segmentation categories. S3DIS has 271 scenes in six areas with points annotated in 13 categories. We follow previous works [20, 64, 77] to test on Area 5. We adopt AdamW optimizer with a weight decay of 0.1. The batch size is set to 8. For ScanNet v2 and S3DIS, we train for 200 and 600 epochs, respectively. The results (mIoU(%)) are shown in Table 1. Great performance gains are attained with the pre-trained weights.

Indoor Object Detection. We adopt two datasets, ScanNet

Method	ScanNet val.		S3DIS Area 5	
	scratch	pre-trained	scratch	pre-trained
PointContrast [64]	72.2	74.1	68.2	70.3
CSC [20]	72.2	73.8	68.2	72.2
DepthContrast [75]	70.3	71.2	68.2	70.6
Ours	73.6	**75.6**	70.7	**73.0**

Table 1. Results (mIoU(%)) of semantic segmentation.

Method	P	ScanNet v2		SUN RGB-D	
		AP$_{50}$	**AP**$_{25}$	**AP**$_{50}$	**AP**$_{25}$
VoteNet [37]	×	33.5	58.6	32.9	57.7
H3DNet [76]	×	48.1	67.2	39.0	60.1
3DETR [34]	×	47.0	65.0	32.7	59.1
GroupFree$_{L6,O256}$ [31]	×	48.9	67.3	45.2	63.0
PointContrast [64] (VoteNet)	✓	38.0	58.5	34.8	57.5
CSC [20] (VoteNet)	✓	39.3	-	36.4	-
DepthContrast [75] (VoteNet)	✓	42.9	64.0	35.5	61.6
DepthContrast [75] (H3DNet)	✓	50.0	69.0	43.4	63.5
IAE [67] (VoteNet)	✓	39.8	61.5	36.0	60.4
RandomRooms [43] (VoteNet)	✓	36.2	61.3	35.4	59.2
RandomRooms [43] (H3DNet)	✓	51.5	68.6	43.1	61.6
STRL [21] (VoteNet)	✓	38.4	59.5	35.0	58.2
MaskPoint [29] (3DETR)	✓	42.1	64.2	-	-
Ours (VoteNet)	×	44.5	66.4	36.7	61.8
Ours (VoteNet)	✓	48.5	67.4	39.5	62.7
Ours (GroupFree$_{L6,O256}$)	×	51.1	70.1	45.4	64.2
Ours (GroupFree$_{L6,O256}$)	✓	**53.7**	**71.8**	**47.5**	**64.8**

Table 2. Results of indoor object detection. The methods in the brackets indicate the detection heads. "P" indicates "Pre-trained".

Method	P	Vehicle	Pedestrian	Cyclist	Avg.
Second [69]†	×	62.02	47.49	53.53	54.35
Ours (Second)	×	64.33	50.18	57.31	57.27
Ours (Second)	✓	65.12	50.87	59.08	**58.36**
CenterPoint [73]†	×	62.65	58.23	64.87	61.92
Ours (CenterPoint)	×	63.99	59.35	67.19	63.51
Ours (CenterPoint)	✓	64.11	60.00	68.66	**64.26**

Table 3. Results of outdoor object detection with 20% training data. "†" denotes the results reported in OpenPCDet [57]. "P" denotes "Pre-trained". Our models use EQ-Net as the backbone network. The metric is mean average precision weighted by heading (mAPH) at Level 2 [55].

v2 [9] and SUN RGB-D [54], for indoor object detection. ScanNet v2 contains 1613 scenes, which are split into 1201, 312, and 100 scenes for training, validation, and testing, respectively. It includes 18 object categories. SUN RGB-D contains 10335 single-view indoor scenes with bounding boxes in 10 categories, including 5285 scenes for training and 5050 scenes for testing. The optimizer used for both datasets is AdamW. We set the weight decay to 0.1, the learning rate to 0.001 with cosine decay, and the training epochs to 200. The batch sizes for ScanNet v2 and SUN RGB-D are 4 and 8, respectively. We conduct experiments based on two detection methods: VoteNet [37] and GroupFree [31]. GroupFree is one of the state-of-the-art methods for indoor object detection. As shown in Table 2, with EQ-Net as the backbone network, we get a high training-from-scratch baseline performance. Based on the strong baseline, our pre-trained weights still improve the mAP by a large margin. Among all the VoteNet-based models, our fine-tuning model gets the highest performance. Also, our GroupFree-based model with pre-trained weights as initialization attains top performance on both datasets.

Outdoor Object Detection. We evaluate the transferring ability of our method to outdoor scenes by conducting object detection experiments on the large-scale Waymo [55] dataset, which includes 798 training sequences with $\approx 158k$ LiDAR samples and 202 validation sequences with $\approx 40k$ LiDAR samples. We follow the settings in OpenPCDet [57] and use 20% of the training data for our experiments. The optimizer is AdamW with a weight decay of 0.01 and a learning rate of 0.003. We train for 30 epochs with a batch size of 4. We experiment with two outdoor detection methods, SECOND [69] and CenterPoint [73], both of which are utilized as fundamental and strong 3D region proposal networks in state-of-the-art outdoor 3D detectors [7,28,50,52]. Table 3 shows the results. Surprisingly, although the pre-training is on an indoor dataset, the learned representations still benefit the fine-tuning on outdoor scenes, which means that some unified intrinsic 3D shape information is learned and exploited in the pre-training process.

Data efficiency of pre-training. An important purpose of pre-training is to improve performance on tasks with limited data. We show that our pre-training is data-efficient by fine-tuning on ScanNet v2 in two settings: limited scenes and limited annotations per scene. For scene and annotation splits, we follow the configurations in CSC [20] and use the official data-efficient splits of ScanNet benchmark [9]. Our experiments are conducted on two tasks: semantic segmentation and object detection. **(i)** We show the semantic segmentation results in Table 4. When trained with {1%, 5%, 10%, 20%} scenes, our models initialized with pre-trained weights consistently outperform the ones with random initialization and attain better results than CSC. Also, with only limited {20, 50, 100, 200} points annotated per scene, our pre-training consistently improves mIoU. We find that in the limited annotation setting, our models with random initialization already achieve much better performance than CSC, as we use a stronger transformer-based backbone. It is noteworthy that when only few data or annotations are available, our pre-training can boost the model performance by a large margin (*e.g.*, +4.4 p.p. for 1% data and +3.3 p.p. for 20 points). **(ii)** The VoteNet [37]-based detection results are shown in Table 5. {10%, 20%, 40%, 80%} scenes are sampled for limited-scene object detection. We make much better baseline predictions than CSC with deficient training data. Our pre-training further brings performance gains based on the strong baselines. For limited-annotation de-

Data Pct.	CSC [20]			Ours		
	scratch	pre-trained	Δ	scratch	pre-trained	Δ
100%	72.2	73.8	+1.6	73.6	**75.6**	+2.0
1%	26.0	28.9	+2.9	25.8	**30.2**	+4.4
5%	47.8	49.8	+2.0	48.1	**50.3**	+2.2
10%	56.7	59.4	+2.7	57.6	**62.3**	+4.7
20%	62.9	64.6	+1.7	63.9	**66.0**	+2.1

(a) Limited scenes.

No. of Points	CSC [20]			Ours		
	scratch	pre-trained	Δ	scratch	pre-trained	Δ
all	72.2	73.8	+1.6	73.6	**75.6**	+2.0
20	53.6	53.8	+0.2	62.2	**65.5**	+3.3
50	60.7	62.9	+2.2	68.1	**70.3**	+2.2
100	65.7	66.9	+1.2	69.0	**71.5**	+2.5
200	68.2	69.0	+0.8	70.4	**72.0**	+1.6

(b) Limited annotated points per scene.

Table 4. Data-efficient semantic segmentation results on ScanNet validation set. The metric is mIoU(%).

Data Pct.	CSC [20] (VoteNet)			Ours (VoteNet)		
	scratch	pre-trained	Δ	scratch	pre-trained	Δ
100%	35.4	39.3	+3.9	44.5	**48.5**	+4.0
10%	0.3	8.6	+8.3	29.5	**32.8**	+3.3
20%	4.6	20.9	+16.3	34.7	**37.2**	+2.5
40%	22.0	29.2	+7.2	37.4	**41.4**	+4.0
80%	33.7	36.7	+3.0	43.0	**46.1**	+3.1

(a) Limited scenes.

No. of Boxes	CSC [20] (VoteNet)			Ours (VoteNet)		
	scratch	pre-trained	Δ	scratch	pre-trained	Δ
all	35.4	39.3	+3.9	44.5	**48.5**	+4.0
1	9.1	10.9	+1.8	16.5	**17.9**	+1.4
2	15.9	18.5	+2.6	23.8	**26.1**	+2.3
4	22.5	26.1	+3.6	30.6	**33.1**	+2.5
7	26.5	30.4	+3.9	34.0	**38.5**	+4.5

(b) Limited annotated boxes per scene.

Table 5. Data-efficient object detection results on ScanNet validation set. VoteNet [37] is the detection head used in the experiments. The metric is mAP@0.5(%).

tection, $\{1, 2, 4, 7\}$ bounding boxes are annotated for each scene. The results in Table 5b again indicate the effectiveness of our pre-training in label-efficient learning.

4.3. Study of the Reconstruction Targets

We compare the fine-tuning performance of different targets in Table 6. Besides our context-enhanced shape target, we also explore *point color* and *local point set* as the reconstruction target. Pixel color carries important semantic information that is helpful in recognizing or separating objects, as validated in 2D works [2, 66]. When it comes to 3D, learning to reconstruct point colors also benefits the semantic understanding, but the performance gain is limited.

Local point set is also used in Point-MAE [36], a recent work for single-object-level 3D masked signal modeling, as a shape reconstruction target. To apply the point set target in 3D scenes, for a masked point, we take the points in a local ball with a radius R centered at that point as its ground-truth local point set. We then set a fixed predicted point number K for each masked point to ensure a fixed channel number for shape features. So for each input masked point, the shape predictor produces a feature of size $K \times 3$, *i.e.*, K 3D coordinates relative to the center. In our implementation, we set R and K to 0.15m and 200, respectively. We minimize the Chamfer distance loss [13] to decrease the distance of the predicted and ground-truth point sets for each input masked point. Local point set is the most explicit representation of local 3D geometry, capable of describing the shape details with points in continuous space, which shows great results in 3D single-object-level pre-training, as shown by Point-MAE [36]. However, local point set is not as effective in 3D scenes as in objects (see Table 6). A possible reason is that for a real-scene point cloud, the scanning and 3D reconstruction inevitably introduce noisy points and inconsistent point densities. In ScanNet, the point numbers in local balls with radius 0.15m range from one to thousands. The local point sets for depicting similar shapes can be very different in point numbers and distributions, which makes local point set an inferior option as the shape target.

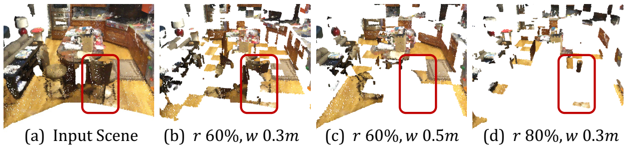

(a) Input Scene (b) r 60%, w 0.3m (c) r 60%, w 0.5m (d) r 80%, w 0.3m

Figure 4. Masked point clouds with different masking settings. In (b), the chair is partially masked out, leaving clues for completing the chair. In (c) and (d), the chair is totally masked out, making the chair reconstruction a hard problem given only visible parts.

In contrast, shape context is a more stable geometric descriptor, which structures the surrounding local space into ordered bins and thus is more robust to the point density and distribution variations. Besides, shape context models the space occupancy, explicitly taking the empty space into consideration, which is also essential in describing shape. Different from the hand-crafted shape context, deep shape features are adaptively learned to fit in with varying point distributions. Also, a deeper encoding of the shape and a more flexible aggregation of contextual information are enabled in this manner. Large downstream performance gains can be attained with only shape context or deep shape features as reconstruction targets. Our context-enhanced shape target combines the strengths of shape context and deep shape feature and achieves better performance. In addition, although the color alone does not bring large improvement, integrating color with our context-enhanced target further boost the fine-tuning performance.

4.4. Ablation Studies

We conduct ablations by fine-tuning on ScanNet semantic segmentation and report mIoU(%) on validation set.

Masking Ratio r and Block Size w. The effects of dif-

Targets				mIoU(%)
Color	Point Set	CEST (Ours)		
		SC	DSF	
from scratch				73.64
✓				73.98
	✓			73.71
		✓		75.05
			✓	74.92
		✓	✓	75.42
✓		✓	✓	**75.57**

Table 6. Effects of different reconstruction targets. The experiments are conducted on ScanNet semantic segmentation based on MSP-SA. CEST, SC, and DSF denote context-enhanced shape target, shape context and deep shape feature, respectively.

r	40%	50%	60%	70%	80%
mIoU	75.00	75.24	**75.57**	74.98	74.34

w (m)	0.2	0.3	0.4	0.5
mIoU	74.85	**75.57**	75.12	74.62

Table 7. Effects of different masking ratios r and block sizes w. The default settings (r 60%, w 0.3m) are marked in gray.

Model	*scratch*	MSP-CA	MSP-CA++	MSP-SA			
				5k	10k	20k	40k
mIoU(%)	73.64	74.85	75.05	75.16	**75.57**	75.41	74.49

Table 8. Analysis on MSP network architectures. MSP-SA is tested with {5k, 10k, 20k, 40k} keypoints.

ferent masking ratios and block sizes are shown in Table 7. The best fine-tuning performance is achieved with a masking ratio of 60% and a block size of 0.3m. An example of the masked point cloud in this case is shown in Fig. 4(b). Shapes in 3D scenes (*e.g.*, the chair) are largely masked out so that the masked shape can not be easily interpolated from the remaining parts, forcing the network to exploit semantic information for completing the shape. However, when the masking ratio or block size is too large, as shown in Fig. 4 (c) and (d), it is likely to mask out the entire objects in the scene, leaving deficient clues for masked object reasoning.

MSP Network Architectures & Keypoint Numbers. We introduce three MSP Network architectures, *i.e.*, MSP-CA, MSP-CA++, and MSP-SA, for mitigating the masked shape leakage. As shown in Table 8, MSP-CA learns meaningful representations, boosting the downstream performance with an improvement of 1.21%. With enhanced information interaction among remaining points, MSP-CA++ further improves the performance. By cutting off communication between masked points, masked shape leakage is entirely avoided in MSP-CA and MSP-CA++. However, the lack of interaction among masked points makes the shape prediction hard when a masked point is far from all the remaining parts, potentially hindering the semantic reasoning in MSP. MSP-SA addresses this concern and meanwhile alleviates the masked shape leakage by building information interaction only between subsampled sparse keypoints. Its performance is greatly affected by the keypoint sampling numbers. With a proper sampling number (*e.g.*, 10k from hundreds of thousands of points in a ScanNet scene), MSP-SA further improves the feature learning, achieving better fine-tuning performance than MSP-CA and MSP-CA++. When the keypoint number is small, fewer shape patterns are covered and learned in the pre-training, causing a performance drop. When sampling too many keypoints, the fine-tuning performance also drops, since the keypoint coordinates reveal too much geometric information.

Feature Extractor	*scratch*	MSP	*Improvement*
SparseConvNet	72.80	74.13	+1.33
EQ-Net	73.64	75.57	+1.93

Table 9. Combination of MSP and different feature extractors.

Method	*scratch*	PointContrast	CSC	MSP (Ours)
mIoU(%)	73.64	74.28	74.88	75.57

Table 10. Comparison of different pre-training methods with EQ-Net as the backbone (*i.e.*, the feature extractor).

Feature Extractor. We also try our pre-training task MSP on another popular feature extraction network, *i.e.,* SparseConvNet [8, 14]. To achieve this, we use SparseConvNet to extract voxel features and then map the voxels to points to get the features of the corresponding keypoints as the input to our MSP network. The fine-tuning results on ScanNet are shown in Table 9. With convolution-based SparseConvNet as the feature extractor, we can still observe improvement brought by MSP, but with the transformer-based EQ-Net, the feature learning ability of MSP is better exploited.

Comparison of Different Pre-training Methods with the Same Backbone. To better compare MSP with other scene-level pre-training methods, we keep the same backbone EQ-Net [71] and apply other methods, *i.e.*, PointContrast [64] and CSC [20], for pre-training. Specifically, we directly adopt the released codes of PointContrast and CSC for the contrastive loss and implement a data loader to augment each scene twice as the pre-training input. The training schemes for both pre-training and fine-tuning are kept the same as MSP. The fine-tuning results are shown in Table 10.

5. Conclusion

Our experiments show that the proposed MSP is a powerful self-supervised pre-training method for 3D scene understanding that significantly boosts the performance of downstream tasks. MSP learns representative features that generalise well, thanks to the combination of robust shape context and flexible deep shape feature as the context-enhanced shape target. Nevertheless, the effectiveness of MSP decreases when training for more epochs in downstream tasks. We conjecture that this is due to the relatively small size of our pre-training dataset. We take the extension of MSP to larger datasets as future work.

References

[1] Iro Armeni, Ozan Sener, Amir Roshan Zamir, Helen Jiang, Ioannis K. Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *CVPR*, 2016. 2, 5

[2] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2022. 1, 2, 4, 5, 7

[3] Serge Belongie, Jitendra Malik, and Jan Puzicha. Shape matching and object recognition using shape contexts. *TPAMI*, 2002. 2, 3

[4] 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. In *NeurIPS*, 2020. 1, 2

[5] 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] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 1, 2

[7] Xuesong Chen, Shaoshuai Shi, Benjin Zhu, Ka Chun Cheung, Hang Xu, and Hongsheng Li. Mppnet: Multi-frame feature intertwining with proxy points for 3d temporal object detection. In *ECCV*, 2022. 6

[8] Christopher B. Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019. 2, 5, 8

[9] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas A. Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017. 2, 5, 6

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019. 1, 2, 4

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1

[12] Francis Engelmann, Martin Bokeloh, Alireza Fathi, Bastian Leibe, and Matthias Nießner. 3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation. In *CVPR*, 2020. 2

[13] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017. 7

[14] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *CVPR*, 2018. 2, 5, 8

[15] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. In *NeurIPS*, 2020. 1, 2, 3

[16] Lei Han, Tian Zheng, Lan Xu, and Lu Fang. Occuseg: Occupancy-aware 3d instance segmentation. In *CVPR*, 2020. 2

[17] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 1, 2, 4, 5

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, June 2020. 1, 2

[19] Ji Hou, Angela Dai, and Matthias Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *CVPR*, 2019. 2

[20] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. In *CVPR*, 2021. 1, 2, 5, 6, 7, 8

[21] Siyuan Huang, Yichen Xie, Song-Chun Zhu, and Yixin Zhu. Spatio-temporal self-supervised representation learning for 3d point clouds. In *ICCV*, 2021. 1, 2, 6

[22] Li Jiang, Hengshuang Zhao, Shu Liu, Xiaoyong Shen, Chi-Wing Fu, and Jiaya Jia. Hierarchical point-edge interaction network for point cloud semantic segmentation. In *ICCV*, 2019. 2

[23] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *CVPR*, 2020. 2

[24] Marcel Körtgen, Gil-Joo Park, Marcin Novotni, and Reinhard Klein. 3d shape matching with 3d shape contexts. In *The 7th central European seminar on computer graphics*, 2003. 2, 3

[25] Jiaxin Li, Ben M. Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. *CVPR*, 2018. 2

[26] Lanxiao Li and Michael Heizmann. A closer look at invariances in self-supervised pre-training for 3d vision. In *ECCV*, 2022. 2

[27] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *NeurIPS*, 2018. 2

[28] Zhichao Li, Feng Wang, and Naiyan Wang. Lidar r-cnn: An efficient and universal 3d object detector. In *CVPR*, 2021. 6

[29] Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. In *ECCV*, 2022. 2, 6

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*, 2021. 1

[31] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3d object detection via transformers. *ICCV*, 2021. 2, 6

[32] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 5

[33] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *ICCV*, 2021. 2

[34] Ishan Misra, Rohit Girdhar, and Armand Joulin. An End-to-End Transformer Model for 3D Object Detection. In *ICCV*, 2021. 6

[35] Alexey Nekrasov, Jonas Schult, Or Litany, Bastian Leibe, and Francis Engelmann. Mix3d: Out-of-context data augmentation for 3d scenes. In *3DV*, 2021. 2

[36] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *ECCV*, 2022. 1, 2, 7

[37] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3d object detection in point clouds. In *ICCV*, 2019. 2, 6, 7

[38] Charles Ruizhongtai Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from RGB-D data. *CVPR*, 2018. 2

[39] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2

[40] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2

[41] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018. 2

[42] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. 2019. 1, 2

[43] Yongming Rao, Benlin Liu, Yi Wei, Jiwen Lu, Cho-Jui Hsieh, and Jie Zhou. Randomrooms: Unsupervised pre-training from synthetic shapes and randomized layouts for 3d object detection. In *ICCV*, 2021. 1, 2, 6

[44] Yongming Rao, Jiwen Lu, and Jie Zhou. Global-local bidirectional reasoning for unsupervised representation learning of 3d point clouds. In *CVPR*, 2020. 2

[45] Jason Tyler Rolfe. Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*, 2016. 2

[46] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, 2009. 2

[47] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, and Michael Beetz. Persistent point feature histograms for 3d point clouds. In *ICoIAS*, 2008. 2

[48] Aditya Sanghi. Info3d: Representation learning on 3d objects using mutual information maximization and contrastive learning. In *ECCV*, 2020. 2

[49] Jonathan Sauder and Bjarne Sievers. Self-supervised deep learning on point clouds by reconstructing space. In *NeurIPS*, 2019. 2

[50] Hualian Sheng, Sijia Cai, Yuan Liu, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, and Min-Jian Zhao. Improving 3d object detection with channel-wise transformer. In *ICCV*, 2021. 6

[51] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. PV-RCNN: point-voxel feature set abstraction for 3d object detection. In *CVPR*, 2020. 2

[52] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *IJCV*, 2022. 6

[53] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 2

[54] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *CVPR*, 2015. 2, 6

[55] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 6

[56] Lyne Tchapmi, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *3DV*, 2017. 2

[57] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection from point clouds. https://github.com/open-mmlab/OpenPCDet, 2020. 6

[58] 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 *ICCV*, 2019. 2

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2, 4, 5

[60] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J Kusner. Unsupervised point cloud pre-training via occlusion completion. In *ICCV*, 2021. 2

[61] 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 Trans. Graph.*, 2019. 2

[62] 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. 1, 2, 4

[63] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 2

[64] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *ECCV*, 2020. 1, 2, 5, 6, 8

[65] Saining Xie, Sainan Liu, Zeyu Chen, and Zhuowen Tu. Attentional shapecontextnet for point cloud recognition. In *CVPR*, 2018. 3

[66] 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. In *CVPR*, 2022. 2, 7

[67] Siming Yan, Zhenpei Yang, Haoxiang Li, Li Guan, Hao Kang, Gang Hua, and Qixing Huang. Implicit autoencoder for point cloud self-supervised representation learning. *arXiv preprint arXiv:2201.00785*, 2022. 2, 6

[68] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *CVPR*, 2020. 2

[69] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 2018. 6

[70] Bo Yang, Jianan Wang, Ronald Clark, Qingyong Hu, Sen Wang, Andrew Markham, and Niki Trigoni. Learning object bounding boxes for 3d instance segmentation on point clouds. In *NeurIPS*, 2019. 2

[71] Zetong Yang, Li Jiang, Yanan Sun, Bernt Schiele, and Jiaya Jia. A unified query-based paradigm for point cloud understanding. In *CVPR*, 2022. 2, 5, 8

[72] Junbo Yin, Dingfu Zhou, Liangjun Zhang, Jin Fang, Cheng-Zhong Xu, Jianbing Shen, and Wenguan Wang. Proposalcontrast: Unsupervised pre-training for lidar-based 3d object detection. In *ECCV*, 2022. 2

[73] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. In *CVPR*, 2021. 2, 6

[74] 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. 2022. 1, 2, 4

[75] Zaiwei Zhang, Rohit Girdhar, Armand Joulin, and Ishan Misra. Self-supervised pretraining of 3d features on any point-cloud. In *ICCV*, 2021. 2, 6

[76] Zaiwei Zhang, Bo Sun, Haitao Yang, and Qixing Huang. H3dnet: 3d object detection using hybrid geometric primitives. In *ECCV*, 2020. 6

[77] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *CVPR*, 2019. 2, 5

[78] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021. 2, 5

[79] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2022. 2