PointGAC: Geometric-Aware Codebook for Masked Point Cloud Modeling

Abiao Li[1], Chenlei Lv[2],[*], Yuming Fang[1], Yifan Zuo[1], Jian Zhang[3], Guofeng Mei[4]

[1]Jiangxi University of Finance and Economics   [2]Shenzhen University

[3]University of Technology Sydney   [4]Fondazione Bruno Kessler

Abstract

*Most masked point cloud modeling (MPM) methods follow a regression paradigm to reconstruct the coordinate or feature of masked regions. However, they tend to over-constrain the model to learn the details of the masked region, resulting in failure to capture generalized features. To address this limitation, we propose **PointGAC**, a novel clustering-based MPM method that aims to align the feature distribution of masked regions. Specially, it features an online codebook-guided teacher-student framework. Firstly, it presents a geometry-aware partitioning strategy to extract initial patches. Then, the teacher model updates a codebook via online k-means based on features extracted from the complete patches. This procedure facilitates codebook vectors to become cluster centers. Afterward, we assigns the unmasked features to their corresponding cluster centers, and the student model aligns the assignment for the reconstructed masked features. This strategy focuses on identifying the cluster centers to which the masked features belong, enabling the model to learn more generalized feature representations. Benefiting from a proposed codebook maintenance mechanism, codebook vectors are actively updated, which further increases the efficiency of semantic feature learning. Experiments validate the effectiveness of the proposed method on various downstream tasks. Code is available at* https://github.com/LAB123-tech/PointGAC.

1. Introduction

Self-supervised point cloud representation learning aims to extract robust and generalizable feature representations from large-scale unlabeled data [32, 33]. These learned representations can significantly improve various downstream applications, including object classification [30], semantic segmentation [21, 23, 24], point cloud completion [50], and other 3D understanding tasks [35, 38, 39]. Due to its ability to learn without relying on annotated data, self-supervised point cloud representation learning has attracted increas-
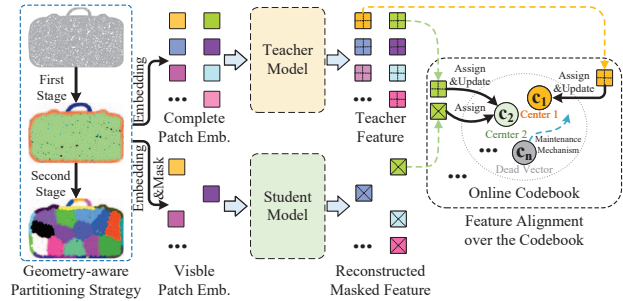
---

*Corresponding author.(chenleilv@mail.bnu.edu.cn)

Figure 1. Fundamental concept of **PointGAC**. Both the unmasked teacher features and the reconstructed masked features at corresponding positions are assigned to the same cluster center over the codebook. This clustering paradigm avoids forcing the student model to overfit the exact feature details of the teacher model.

ing research interest. This is achieved by designing pretext tasks that encourage the model to capture meaningful geometric features. Among these, masked point cloud modeling is a prominent approach in vision transformers.

Inspired by the success of MAE [16], masked point cloud modeling [7, 28, 34, 44, 56, 58] aims to train models capable of reconstructing the coordinates of masked regions using the visible portions of the point cloud. However, these coordinate-based reconstruction approaches focus on capturing surface details, limiting their effectiveness in learning semantic information and resulting in inferior performance in downstream tasks. To mitigate this limitation, recent works [10, 17, 25, 42, 43, 48, 53] adopt a teacher-student framework, where instead of reconstructing coordinates, the student model is trained to align with the masked features extracted by the teacher model. However, a notable drawback of these approaches is that they tend to over-constrain the student model, forcing it to regress the point-wise feature details of the teacher, which prevents it from learning more generalized features. Moreover, these methods often overlook geometric consistency when constructing local point cloud patches, making it challenging for the model to learn meaningful representations.

To address these issues, we propose **PointGAC**, a novel clustering-based approach that aims to learning the similarity of features in their distribution, as illustrated in Fig. 1. PointGAC begins by partitioning the input point cloud into

non-overlapping, geometrically homogeneous patches. After embedding the patches, the teacher model updates an online codebook by leveraging the features extracted from the complete patch embeddings. As a result, codebook vectors gradually evolve into cluster centers. The student model is tasked with reconstructing the masked features from the visible patch embedding. Finally, the reconstructed masked features and the unmasked teacher features at the corresponding locations are assigned to the same cluster center to achieve feature alignment. This clustering paradigm enables the student model to learn the cluster centers to which the masked point cloud patches belong. It avoids the need of point-wise feature alignment, facilitating the learning of more generalized feature representations in the student network. Moreover, we find that many vectors in the codebook remain unupdated and become dead vectors. Therefore, we design a codebook maintenance mechanism to revitalize these dead vectors. It helps to improve the efficiency of masked feature alignment over the codebook. Our method presents a general solution for point cloud-based semantic analysis. Experimental results demonstrate its efficacy across a variety of downstream tasks. The contribution can be summarized as follows:

- We propose a novel online codebook-guided teacher-student framework that follows a clustering-based paradigm to accomplish MPM tasks. This framework predicts the cluster center to which the masked region belongs, effectively avoiding the problem of over-constraining the model.
- We propose a geometry-aware partitioning strategy to divide the point cloud into geometrically homogeneous patches. This strategy ensures geometric consistency within each local point cloud patch, enhancing the quality and efficiency of feature learning.
- We design a codebook maintenance mechanism to increase the update frequency of the codebook vectors. It improves the efficiency of masked feature alignment.

2. Related Work

Masked point modeling. Inspired by the success of Mask Image Modeling (MIM), MPM has also gained increasing attention in point cloud research. The pioneering algorithm [52] introduces a trained tokenizer, allowing pretraining to be performed through masked token prediction. However, the tokenizer is trained offline, making it immutable during pretraining. Other works[18, 44, 54] explore multi-task learning to embed both local and global contexts for point cloud pretraining. Several studies [28, 48, 55] are dedicated to designing complex models, leading to significant advancements in the reconstruction of point cloud geometric structures. However, the model spends significant resources to accurately recover the position of each point, which may not be important for downstream tasks re-

lated to semantic understanding. Hence, certain algorithms [10, 17, 43, 48, 53] shift their focus toward reconstructing the features of masked regions.

Although existing methods have achieved impressive results, they are regression-based approaches for reconstructing the information of the masked regions. Such regression-based algorithms tend to over-constrain the model, leading it to focus excessively on reconstructing details while failing to learn more generalizable features. In this paper, we adopt a clustering-based method to identify which cluster center in the codebook the masked features belong to. This allows the model to avoid this problem.

Codebook for representation learning. The codebook is widely used as a feature representation method in computer vision. In generative models, [40, 46] leverages the codebook to map continuous features to discrete tokens, facilitating the diverse visual outputs from the textual descriptions. Inspired by this, subsequent studies [3, 11] represent the object as a series of discrete tokens to capture structural patterns. Several methods [1, 2, 5, 12] utilize a codebook to align representations of different augmented views in the clustering space. SwAV [5] adopts an optimal transport strategy [8] that assumes a uniform assignment of each batch data to all prototypes, while MSN [2] and PAWS [1] employ entropy-based regularization [19] to encourage balanced prototype usage. However, such an assumption may not hold in our setting due to the imbalanced and potentially diverse semantic distribution within each mini-batch. Instead, we adopt an online k-means algorithm to progressively update the codebook and introduce a maintenance mechanism to improve its utilization efficiency.

3. Method

Overview. Fig. 2 shows the architecture of the proposed PointGAC, adopting an online codebook-guided teacher-student framework. Given an input point cloud P with N points, we first partition it into L non-overlapping refined patches using a geometry-aware partitioning strategy. The central points are then processed through an MLP layer to obtain the positional embedding for each patch. Each patch undergoes the operation of normalization before being passed into a mini-PointNet for patch embedding. All patch and positional embeddings are summed to form input features for subsequent teacher-student framework.

In the teacher branch, input features are fed into an encoder to update codebook vectors via online k-means clustering. Encoded teacher features are matched with the codebook to generate teacher assignments, which serve as the target for the student model to learn. In the student branch, the task is to recover the masked features from partially visible features, obtaining the reconstructed features. Then, the reconstructed features are accomplished by matching with the codebook to generate student assignments. Then,
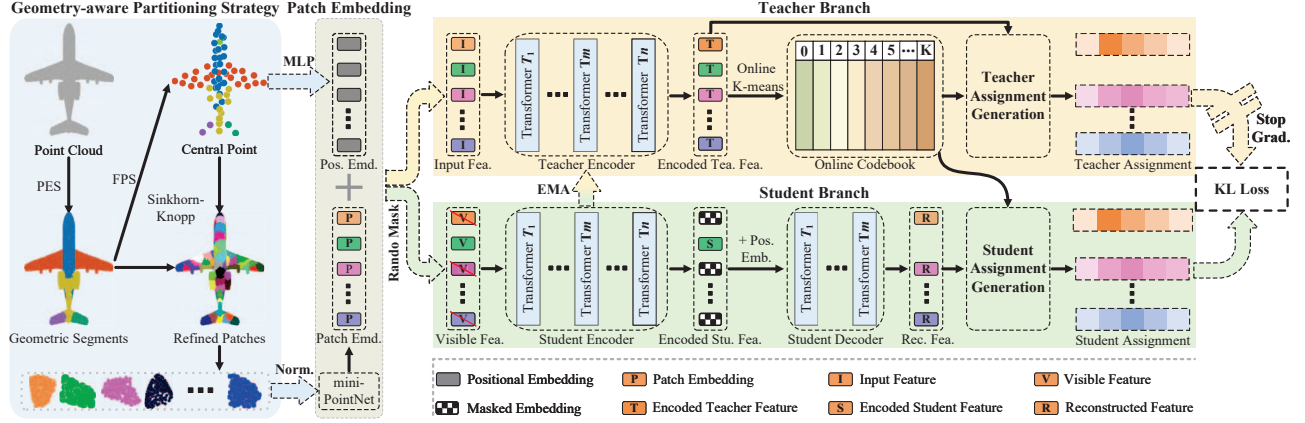
Figure 2. The framework of **PointGAC**. This framework utilizes a online codebook to achieve feature alignment. It aligns the reconstructed features on the student side with the encoded features on the teacher side at corresponding positions.

the teacher and student assignments are aligned using a KL divergence loss. During training, gradients are propagated only through the student branch, while the teacher encoder is updated using an Exponential Moving Average (EMA) of the student encoder's parameters. In the following, we will detail the key components of the proposed algorithm.

3.1. Geometry-aware Partitioning Strategy

In MIM tasks, due to the regular and structured nature of images, algorithms [4, 15, 57] typically divide images into non-overlapping grids. However, since point clouds are unordered and unstructured data, the vast majority of current algorithms rely on a combination of FPS (Farthest Point Sampling) and KNN (K-Nearest Neighbors) to construct overlapping point cloud patches. A major drawback of these methods is the leakage of information from visible patches to masked patches due to the information redundancy between patches, which reduces the practicality. To address this issue, Wang *et al.* [48] introduced an optimal transport-based approach to partition the point cloud into non-overlapping patches. However, it overlooks the geometric properties of the point cloud.

To overcome such limitations, we propose a geometry-aware partitioning strategy that divides a point cloud into non-overlapping and geometrically consistent patches, as shown in Fig. 2 (blue area). To group points with similar geometric features, we apply Potts Energy Segmentation (PES) [13, 22] for coarse partitioning, formulated as:

$$\arg \min_{g \in \mathbb{R}^{dg}} \sum_{i \in P} \|g_i - f_i\|^2 + \mu \sum_{(i,j) \in E_{\text{nn}}} w_{ij} [g_i - g_j \neq 0], \quad (1)$$

where f_i is the original geometric feature of point i, as defined in [22], g_i is the target geometric label, P represents the point cloud, E_{nn} is the set of edges that connect neighboring points, defining the local neighborhood structure, w_{ij} is the weight determining the influence of the pairwise interaction between points i and j. The Iverson bracket $[\cdot]$

equals 1 if the condition holds and 0 otherwise. The factor $\mu > 0$ regulates the coarseness of the resulting partition. This enables the generation of meaningful, non-overlapping geometric segments.

In MPM, a fixed number of patches is required, whereas PES produces a varying number of segments across different point clouds. To reconcile these differences, we use FPS to sample L central points (along with their geometric labels) from the geometric segments. Then, we assign each point to related central point, thereby forming L refined patches. This assignment satisfies two conditions: 1) All points within a patch belong to the same geometric segment; 2) Different patches belong to the same segment contain a similar number of points. We denote the assignment by a matrix Q, whose elements $Q_{ij} \in [0, 1]$.

To enforce the first condition, we define a mask matrix M with elements $M_{ij} \in \{0, 1\}$, where $M_{ij} = 1$ if the central point and its assigned points share the same geometric label, and 0 otherwise. We then apply M to Q via the Hadamard (element-wise) product $Q \circ M$ to guide the assignments. The second condition can be approximated as $\sum_i (Q \circ M)_{ij} / L = N/L$. Since FPS sampling tends to select more cenral points from segments containing a larger number of points, it naturally balances the partitioning. Consequently, there is no need to enforce explicit constraints at the level of each individual segment. To satisfy both constraints, we formulate the assignment process as a *masked optimal transport*, expressed as:

$$\min_Q \sum_{i=1}^{N} \sum_{j=1}^{L} M_{ij} Q_{ij} T_{ij}, \text{s.t.,} \sum_{i=1}^{N} \frac{(Q \circ M)_{ij}}{L} = \frac{N}{L}, \quad (2)$$

where T is the cost. Following [31], the solution to this problem is achieved based on Sinkhorn algorithm [8]. This process yields refined patches $\{P_1, \cdots, P_L\}$ that are geometrically aware, ensuring each patch consists exclusively of points from the same geometric segment.

Patch embedding layer. To extract features from the achieved patches, patch embedding layers are designed. Each refined patch is normalized by subtracting its center. The resulting patches are then processed by a mini-PointNet [36] to generate patch embeddings $E \in \mathbb{R}^{L \times D}$ (D is the dimension), which can be expressed as:

$$E = \text{PointNet}\left(\{P_1, \cdots, P_L\}\right), \tag{3}$$

Notably, patch sizes are not uniform. To enable parallel feature extraction across different patches, we construct a mask matrix $H \in \mathbb{R}^{N \times N}$ over the entire point cloud, where $H_{i,j} = 1$ indicates that points i and j belong to the same patch. Max pooling is then applied to regions where the mask matrix has a value of 1 in each row. Meanwhile, position embeddings $PE \in \mathbb{R}^{L \times D}$ are extracted from the patch center using an MLP layer. Finally, the patch embeddings E and the positional embeddings PE are summed to obtain F, which serves as input features for subsequent modules.

3.2. Teacher-Student Framework

To achieve feature alignment in a self-supervised manner, we employ a teacher-student framework, where the student branch reconstructs masked features based on the knowledge provided by the teacher branch.

Teacher branch. Given the input feature F, the encoder applies a self-attention mechanism to encode F. It is composed of n standard transformer blocks. Specially, the encoded teacher feature within the i-th transformer block of the encoder can be formulated as:

$$T_i = \text{Attn}\left(Q, K, V\right) = \text{SoftMax}\left(\frac{QK}{\sqrt{D}}\right)V, \tag{4}$$

where $Q = T_{i-1}W_Q$, $K = T_{i-1}W_K$, and $V = T_{i-1}W_V$, with W_Q, W_K, and W_V being learnable parameters. Furthermore, $T_0 = F$.

Subsequently, we utilize the encoded teacher features to update the codebook $C = [c_1, c_2, \ldots, c_K]$ where each vector $c_k \in \mathbb{R}^{1 \times D}$ represents a D-dimensional vector. The codebook is updated using an online k-means algorithm [46]. For each teacher feature, we identify the most similar codebook vector. We use $n_k^{(t)}$ to denote the number of features assigned to c_k at the t-th iteration in the pretraining. We also use $m_k^{(t)}$ to represent the element-wise sum of these features. To maintain a stable update processing, we utilize an exponential moving average:

$$N_k^{(t)} = \gamma N_k^{(t-1)} + (1 - \gamma)n_k^{(t)},$$
$$M_k^{(t)} = \gamma M_k^{(t-1)} + (1 - \gamma)m_k^{(t)}, \tag{5}$$

where $N_k^{(t)}$ is the accumulated count of the features assigned to c_k at the t-th iteration, with $N_k^{(1)} = 1$ as the initial value. $M_k^{(t)}$ is the accumulated element-wise sum of these

features and $M_k^{(1)}$ is initialized as the k-th codebook vector. Then, c_k is given by $c_k = M_k/N_k$. We set $\gamma = 0.99$ to ensure that the update processing remains stable, allowing the codebook vector to become the cluster center gradually.

Teacher assignment generation. To assign the encoded teacher features to cluster centers within the codebook, we compute the soft assignment of encoded teacher features across the codebook. Given the encoded teacher features $T = \{t_i\}_{i=1}^L, t_i \in \mathbb{R}^{1 \times D}$, the soft assignment $Q_t \in \mathbb{R}^{L \times K}$ is computed as the following formula:

$$Q_t = \text{SoftMax}\left(\frac{TC}{\tau_t}\right), \tag{6}$$

where τ_t is the temperature coefficient used to adjust the smoothness of Q_t. In our experiments, we adopt a cosine annealing strategy to adjust the value of τ_t. At the early stage, τ_t is relatively large, resulting in smoother distribution. As training progresses, we gradually reduce τ_t, which results in highly peaky distribution. Essentially, Q_t encodes the cluster center over the codebook to which the teacher feature belongs. We use these generated assignment vectors Q_t as the supervision signals for the student branch.

Student branch. The task of the student branch is to reconstruct the masked feature from the visible features. With a predefined masking ratio r, we apply random masking to the input feature F. The masked features are denoted as $S_m \in \mathbb{R}^{\lfloor Lr \rfloor \times D}$, and the visible features are denoted as $S_v \in \mathbb{R}^{\lceil L(1-r) \rceil \times D}$, where $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ denote the floor and ceiling functions, respectively. On the student side, apart from possessing an encoder with the same structure as the teacher model, it also includes a decoder that comprises m transformer modules. The student encoder extracts features from S_v, yielding the encoded student features S_E. Then, S_E is concatenated with the learnable mask embeddings $S_a \in \mathbb{R}^{\lfloor Lr \rfloor \times D}$ and added to the full set of positional information PE, serving as the input to the decoder. After being processed by the decoder, the reconstructed feature $S_R \in \mathbb{R}^{Lm \times D}$ is used to align the encoded teacher features.

Student assignment generation. The reconstructed features S_R and the teacher features T at the masked locations should be assigned to the same cluster center to achieve feature alignment. We compute the soft assignment $Q_s \in \mathbb{R}^{L \times K}$ of the features S_R over the codebook C using the following formula:

$$Q_s = \text{SoftMax}\left(\frac{S_D C}{\tau_s}\right), \tag{7}$$

where τ_t is the temperature parameter that controls the sharpness of the soft assignment. In the experiments, we set τ_s to a constant value that is larger than τ_t. It ensures the student assignment Q_s is smoother in distribution compared to the teacher assignment Q_T, helping the reconstructed feature to better align to the teacher feature.

3.3. Codebook Maintenance Mechanism

The codebook vectors are designed to serve as cluster centers for the patch features. However, many vectors remain unused and gradually become 'dead vectors' since online k-means are based on the nearest-neighbor lookup. To analyze this issue, we record the update frequency of all codebook vectors after training. The frequency matrix (1, K) was reshaped to (H, W) for visualization. As shown in Fig. 3(a), without any perturbation, many codebook vectors exhibit low update frequencies. For this issue, we experimentally introduced random perturbations to codebook vectors during training and observed it improved codebook utilization to some extent, as shown in Fig. 3(b).

Building on this insight, we propose a codebook maintenance mechanism, which applies varying degrees of meaningful perturbations to each codebook vector based on its update frequency. For each codebook vector c_k, we assign the most similar teacher feature t_k from the current batch of data to update it. The principle is that frequently updated codebook vectors receive minimal perturbations, whereas infrequently updated or dead vectors undergo larger perturbations. Specially, we count the number of times the codebook vector is updated, denoted as x. Then, we compute a weight that determines the magnitude of the perturbation to be added. The weights α_k are computed as:

$$\alpha_k = \frac{1}{1 + e^{\epsilon(x-\bar{x})}},$$
$$c_k \leftarrow c_k \cdot (1 - \alpha_k) + t_k \cdot \alpha_k, \tag{8}$$

where $\bar{x}$ is the average of the maximum and minimum update counts among all the codebook vectors. ϵ is a small value used to smoothly adjust the weights assigned to different codebook vectors. As illustrated in Fig. 3(c), the majority of the codebook vectors exhibit a high update frequency, indicating that they are being sufficiently updated at this stage. Experiments discuss the improvement.

4. Experiments

Pre-training dataset. We pre-trained the proposed model on ShapeNet55 [6], which consists of 52,472 unique 3D models covering 55 object categories. Before pre-training, we apply Potts energy segmentation on the point cloud to complete the geometric partitioning. Geometric labels are stored in the dataset to speed up pre-training.

Experimental details. We sample $N = 1024$ points with their corresponding geometric labels from the dataset using FPS and partition them into $L = 64$ patches. We employ the AdamW optimizer [20] with a weight decay of 0.04 for training. The learning rate follows a cosine decay schedule [29], starting with a 50-epoch linear warm-up phase and reaching a maximum learning rate of 0.001. τ_t is initialized at 0.07 and decay to 0.04 during training. τ_s remains fixed
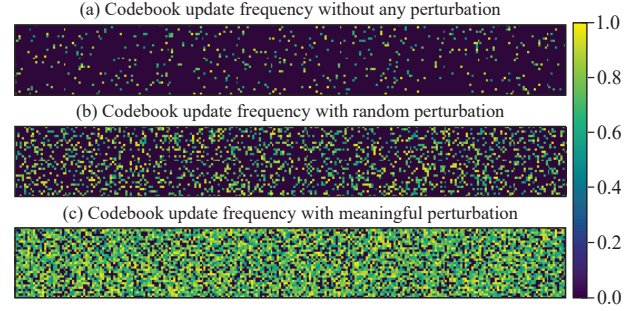

Figure 3. Visualization comparison of codebook update frequency under different perturbation modes. Each pixel in the figure represents the update frequency of a single vector.

at 0.1 throughout the entire training process. Following the practice in OBoW [12], the codebook size K is set to 8192.
Model architecture. The teacher and student encoders share the same structure, both composed of $n = 12$ transformer blocks. The student decoder consists of $m = 4$ standard transformer blocks. The hidden dimension of transformer blocks is $D = 384$.

4.1. Evaluations on Downstream Tasks

Shape classification. We evaluate our method by freezing the pre-trained student encoder and training a linear SVM on ModelNet40 for shape classification. We also report results with and without the voting trick after fine-tuning on ModelNet40. As shown in Tab. 1, compared to other teacher-student models following the regression paradigm, such as Point2Vec [53] and RIMAE [42], our method achieves the best performance in the Linear SVM evaluation. Since regressing the teacher features requires an accurate matching of each feature value, which makes the optimization process more challenging. By aligning the cluster centers to which the features belong, we only need to focus on the similarity of the feature distributions, thus simplifying the optimization process.

We further validate our model on the ScanObjectNN dataset, which contains around 15 000 real-world objects from 15 categories. We report results on three subsets: OBJ-BG (objects with background), OBJ-ONLY (objects only), and PB-T50-RS (objects with background and perturbations). To ensure fairness, each input point cloud is sampled to 2048 points. As shown in Tab. 1, PointGAC achieves competitive performance, particularly in challenging settings. Although the algorithm is pre-trained on the synthetic dataset, it generalizes well to real-world datasets and achieves the best performance. Also, on the OBJ_ONLY variant, our method shows significant performance improvement. Since the results of geometric partitioning remain unaffected by background noise. This property makes it advantageous for codebook vectors to serve as geometrically well-defined cluster centers.

Method	References	#Params(M)	ModelNet40				ScanObjectNN			
			#Points	Linear SVM	w/o Vote	w/ Vote	#Points	OBJ-BG	OBJ-ONLY	PB_T50_RS
Point-BERT[52]	CVPR 2022	22.1	1k points	87.4	92.7	93.2	1k points	87.4	88.1	83.1
PointMAE[34]	ECCV 2022	22.1	1k points	92.7	93.2	93.8	2k points	90.0	88.3	85.2
PointM2AE[55]	NeurIPS 2022	15.3	1k points	92.9	93.4	94.0	2k points	91.2	88.8	86.4
Point2Vec[53]	GCPR 2023	23.2	1k points	92.1	92.5	93.1	2k points	91.2	90.4	87.5
PointGPT[7]	NeurIPS 2023	19.5	1k points	-	-	94.0	2k points	91.6	90.0	86.9
MAE3D[18]	TMM 2023	23.4	1k points	-	93.4	-	2k points	88.4	87.7	86.2
PointGame[28]	TGRS 2023	-	1k points	90.0	93.1	93.5	2k points	89.0	89.0	85.8
MaskFeat3D[49]	ICLR 2024	-	1k points	91.1	93.3	93.7	2k points	91.7	90.0	87.7
RIMAE[42]	AAAI 2025	23.8	1k points	92.8	93.1	93.9	2k points	91.9	89.3	86.8
Ours	-	24.1	1k points	**93.5**	**94.3**	**94.6**	2k points	**92.6**	**91.4**	**89.4**

Table 1. Classification accuracy on ModelNet40 and ScanObjectNN. #Params indicates model size. For ModelNet40, overall accuracy (%) is reported with and without voting. For ScanObjectNN, overall accuracy (%) is shown for three variants.

Method	5-way		10-way	
	10-shot	20-shot	10-shot	20-shot
Point-BERT[52]	94.6±3.6	93.9±3.1	86.4±5.4	91.3±4.6
MaskPoint[27]	95.0±3.7	97.2±1.7	91.4±4.0	92.7±5.1
Point-MAE[34]	96.3±2.5	97.8±1.8	92.6±4.1	93.4±3.5
Point-M2AE[55]	96.8±1.8	98.3±1.4	92.3±4.5	95.0±3.0
PointGame[28]	96.7±2.7	97.9±1.5	92.9±3.7	94.7±2.6
MAE3D[18]	95.2±3.1	97.9±1.6	91.1±4.6	95.3±3.1
PointOTG[48]	97.2±2.3	98.7±1.2	93.2±3.4	95.6±2.6
Ours	**97.4±2.0**	**98.9±1.3**	**93.7±3.1**	**95.9±2.4**

Table 2. Few-shot classification results on ModelNet40. The mean accuracy and standard deviation are calculated in each setting.

Method	CD-S	CD-M	CD-H	CD-Avg	F1
Point-JEPA[41]	0.83	1.23	2.06	1.37	0.34
Pos-Bert[10]	0.91	1.27	2.25	1.48	0.32
3D-JEPA[17]	0.71	1.22	2.02	1.32	0.35
Point2Vec[53]	0.58	0.81	1.92	1.10	0.38
Point-MSD[43]	0.65	0.83	1.87	1.12	0.37
PM-MAE[26]	0.53	0.79	1.81	1.04	0.41
RIMAE[42]	**0.51**	0.72	1.74	0.99	0.46
Ours	0.52	**0.69**	**1.63**	**0.95**	**0.49**

Table 4. Results of point cloud completion on ShapeNet55 dataset. We use CD-S, CD-M, and CD-H to denote CD results in the simple, moderate, and hard settings, respectively.

Method	Semantic segmentation		Part segmentation	
	mAcc	mIoU	$mIoU_C$	$mIoU_I$
Transformer[47]	68.6	60.0	83.4	84.7
Point-MAE[34]	69.9	60.8	84.2	86.1
Point-BERT[52]	69.7	60.5	84.1	85.6
PCP-MAE[56]	70.4	61.1	84.9	86.1
Point-LGMask[44]	70.3	61.3	84.4	86.1
PointMPP[9]	69.8	61.4	84.7	86.1
RIMAE[42]	68.4	60.3	82.1	84.3
Ours	**70.9**	**61.8**	**85.1**	**86.7**

Table 3. Segmentation results on S3DIS and ShapeNetPart. For S3DIS, mean instance IoU (mIoU) and mean class accuracy (mAcc) are reported; for ShapeNetPart, category-level ($mIoU_C$) and instance-level ($mIoU_I$) mean IoU are provided.

Few-shot classification. We perform few-shot experiments on the ModelNet40 dataset using the m-way, n-shot setting, where $m \in \{5, 10\}$ denotes the number of randomly selected categories and $n \in \{10, 20\}$ indicating the number of samples per category. During testing, we randomly selected 20 unseen samples from each category. In experiments, we performed 10 independent trials for each setting. The experimental results are shown in Tab. 2. It can be observed that even with limited data, our method exhibits competitive performance compared to other methods, while showing minimal deviation in most cases.

Semantic segmentation. We test the generalization of our method on the large-scale S3DIS dataset for semantic segmentation evaluations. The S3DIS dataset consists of 271 scenes from 6 indoor regions, with each point labeled into 13 categories. Following the common practice, we use Area 5 for testing and the remaining areas for training. The experimental results are shown in Tab. 3. Compared to other methods, PointGAC achieves the best performance in terms of mIoU and mAcc metrics. Meanwhile, we provide a qualitative comparison of semantic segmentation results on the S3DIS dataset. As illustrated in Fig. 4, the predictions of our algorithm are closer to the ground truth and less incorrectly segmented result than PointMPP and PCP-MAE.

Part segmentation. We evaluate part segmentation performance of PointGAC on ShapeNetPart dataset, which consists of 16,881 samples across 16 categories. We partition the point cloud into 128 patches using our geometry-aware partitioning strategy. We use a trilinear interpolation algorithm [37] to up-sample the encoded features to all input points. The experimental results are shown in Tab. 3.

ceiling | floor | wall | beam | column | window | door | table | chair | sofa | bookcase | board | clutter

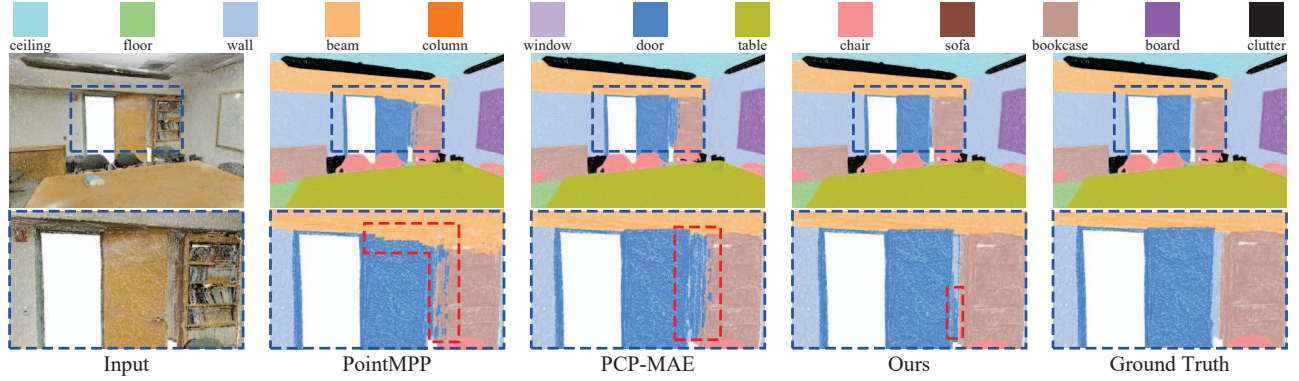Input   PointMPP   PCP-MAE   Ours   Ground Truth

Figure 4. Qualitative comparison on S3DIS semantic segmentation. The first column shows the original point cloud input, followed by columns 2-4, which display the segmentation results of PointMPP, PCP-MAE, and our method. The fifth column shows the ground truth. Regions with poor segmentation results are marked with red dashed boxes.

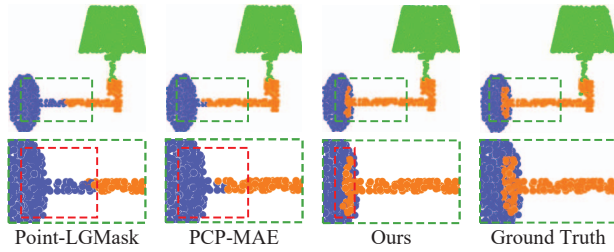Point-LGMask   PCP-MAE   Ours   Ground Truth

Figure 5. Qualitative comparison of different algorithms on ShapeNetPart, with differences highlighted in red boxes.

Our method achieves better results in terms of mIoU_C and mIoU_I. In our approach, the features are extracted from geometrically homogeneous patches, effectively capturing the distinctive properties of the local geometry. It is beneficial for recovering the features of the original input points. In Fig. 5, we visualize the segmentation results of different methods. PointGAC exhibits minimal segmentation errors.

Point cloud completion. Finally, we further validate the effectiveness of PointGAC on the ShapeNet55 dataset for the point cloud completion task. Following Pointr [51], we randomly remove 25% to 75% of points, creating varying levels of incompleteness. The remaining points are then sampled to 2048 points, which are used as the input. We use the original object containing 8192 points as the ground truth. During testing, we set the number of deleted points to 2048, 4096, or 6144. These correspond to three difficulty levels: simple, moderate, and hard. We use the ℓ_2 Chamfer Distance (CD) and F1-Score[45] as the evaluation metrics.

In Tab. 4, we compare PointGAC with other self-supervised algorithms whose pretext tasks focus on regressing the mask features. It can be seen that PointGAC achieves the lowest CD in both moderate and hard settings. In terms of F1-Score, PointGAC also achieves the best performance, which further indicates that the features extracted by our method are more robust to geometric reconstruction. We also provide a qualitative comparison of the point

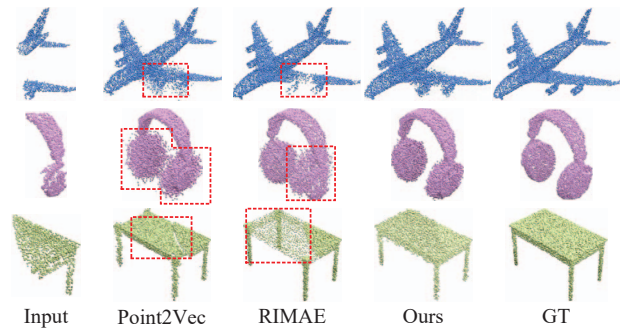
Input   Point2Vec   RIMAE   Ours   GT

Figure 6. Qualitative comparison of different algorithms on the point cloud completion task.

cloud completion results between the proposed algorithm and other methods in the hard setting. As shown in Fig. 6, it can be observed that almost all geometric details are lost for the input target. While the results reconstructed by other algorithms remain unsatisfactory, PointGAC achieves significantly higher fidelity in the reconstructed outputs.

4.2. Ablation Studies

Effects of different components. To further validate the effectiveness of the main module design, we conduct extensive experiments on the ModelNet40 dataset. We adopt the Point2Vec[53] method as our baseline model. It employs a teacher-student framework, where the student network learns to regress the feature output from the teacher network. The experimental results are shown in Tab. 5. It can be observed that the performance of the baseline algorithm is not ideal since regression is a challenging task. For Model B, directly constructing the codebook to perform the MPM task via online k-means did not lead to an improvement. Since the codebook at this stage contains many dead vectors. As observed in Model C, the codebook maintenance mechanism addresses this issue, enabling efficient feature alignment on the codebook for the masked point cloud modeling task. According to Model D, the perfor-

Models	Baseline	Codebook Update		GAP	−/+ Vote
		OKC	CMM		
A	✓				92.5 / 93.1
B	✓	✓			92.1 / 92.5
C	✓	✓	✓		93.7 / 94.2
D	✓			✓	92.8 / 93.4
E	✓	✓	✓	✓	**94.3 / 94.6**

Table 5. Ablation study of the main modules in the proposed algorithm on ModelNet40. OKC: Online K-means Clustering. CMM: Codebook Maintenance Mechanism. GAP: Geometry-aware Partition Strategy. −/+ Vote: without or with the voting strategy.

Group Strategy	Mask Ratio			w/o Vote	w/ Vote
	0.7	0.8	0.9		
KNN	✓			93.2	93.8
KNN		✓		93.7	94.2
KNN			✓	93.4	94.0
GAP	✓			93.9	94.4
GAP		✓		**94.3**	**94.6**
GAP			✓	93.6	94.2

Table 6. Evaluation of the geometric-aware partitioning strategy under different mask ratios.

mance is enhanced when the geometric-aware partitioning strategy replaces the traditional KNN in Model A. Furthermore, the performance reaches its peak when the geometry-aware partitioning strategy is applied to Model C, as demonstrated in Model E. At this point, the geometry represented by the codebook vectors becomes more distinct, enabling them to better serve as cluster centers.

Effect of different grouping strategies. We validate the impact of different grouping methods for constructing point cloud patches on ModelNet40. Specifically, we compare our method with conventional KNN algorithms under different masking ratios, as shown in Tab. 6. It can be seen that the performance of the KNN is not ideal under different masking ratios. It leads to information leakage among the obtained point cloud patches, which simplifies the pretext task and ultimately limits the model performance. Benefiting from geometric consistency, the proposed geometry-aware partitioning strategy improves accuracy compared to KNN. Also, appropriately increasing the masking rate can enhance the overall difficulty of the self-supervised pretext task, thus further improving the model performance.

Effect of different codebook construction formats. We present the impact of different codebook construction formats using the ModelNet40 dataset. Considering that codebook vectors should be as active as possible, we adopt a queue-based method for codebook construction, following the approach of MoCo [14]. Inspired by SwAV [5], we also compare with an alternative implementation using the Sinkhorn-Knopp algorithm [8]. As shown in Tab. 7, the

Codebook Format	Perturbed mode	w/o Vote	w/ Vote
Queue-based	None	93.7	94.1
Sinkhorn-based	None	93.1	93.4
Online k-means	None	92.7	93.2
Online k-means	Random	93.4	93.8
Online k-means	Meaningful	**94.3**	**94.6**

Table 7. Evaluation of the codebook construction format under different perturbed modes.

queue-based codebook ensures all vectors remain active by continuously updating with the latest batch data. However, its vectors do not serve as true cluster centers. Although the Sinkhorn-Knopp algorithm mitigates this by enforcing uniform assignments to K clusters, this assumption is often inconsistent with real-world data distributions, leading to suboptimal performance. The codebook constructed using online k-means can solve this problem. However, the performance does not improve if we simply replace it without further optimization. Because there are many dead vectors in the codebook that have not been updated. After adding meaningful perturbations to the vectors in the codebook as proposed in our design, the model achieves its best performance. It achieves much higher codebook utilization (see in Fig. 3(c)) which ensures the accuracy of the feature alignment over the codebook.

5. Conclusion

In this paper, we propose a novel online codebook-guided teacher-student framework that adopts a clustering-based paradigm to for the MPM task. Unlike traditional regression methods, our method focuses on identifying the cluster centers to which the masked features belong, enabling the model to learn more generalized features. Moreover, by incorporating a geometry-aware partitioning strategy, we ensure geometric consistency within local point cloud block, which enhances the quality of feature learning. To further improves the efficiency of masked feature alignment, we introduce a codebook maintenance mechanism that promotes the convergence of the codebook vectors towards the optimal cluster centroid. Extensive experiments on various downstream tasks validate the effectiveness of the proposed method. Our proposed pre-training framework provides a novel direction to advance point cloud processing.

6. Acknowledgement

References

[1] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Armand Joulin, Nicolas Ballas, and Michael Rabbat. Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples. In *ICCV*, pages 8443–8452, 2021. 2

[2] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Mike Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning. In *ECCV*, pages 456–473. Springer, 2022. 2

[3] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 2

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

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 33:9912–9924, 2020. 2, 8

[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. 5

[7] Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *NeurIPS*, 36, 2024. 1, 6

[8] Cuturi M Sinkhorn Distances. Lightspeed computation of optimal transport. *NeurIPS*, 26:2292–2300, 2013. 2, 3, 8

[9] Songlin Fan, Wei Gao, and Ge Li. Point-mpp: Point cloud self-supervised learning from masked position prediction. *IEEE TNNLS*, pages 1–13, 2024. 6

[10] Kexue Fu, Peng Gao, ShaoLei Liu, Linhao Qu, Longxiang Gao, and Manning Wang. Pos-bert: Point cloud one-stage bert pre-training. *ESWA*, 240:122563, 2024. 1, 2, 6

[11] Zigang Geng, Chunyu Wang, Yixuan Wei, Ze Liu, Houqiang Li, and Han Hu. Human pose as compositional tokens. In *CVPR*, pages 660–671, 2023. 2

[12] Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Pérez. Obow: Online bag-of-visual-words generation for self-supervised learning. In *CVPR*, pages 6830–6840, 2021. 2, 5

[13] Stéphane Guinard and Loic Landrieu. Weakly supervised segmentation-aided classification of urban scenes from 3d lidar point clouds. *ISPRS Archives*, 42:151–157, 2017. 3

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020. 8

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pages 16000–16009, 2022. 3

[16] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pages 16000–16009, 2022. 1

[17] Naiwen Hu, Haozhe Cheng, Yifan Xie, Shiqi Li, and Jihua Zhu. 3d-jepa: A joint embedding predictive architecture for 3d self-supervised representation learning. *arXiv preprint arXiv:2409.15803*, 2024. 1, 2, 6

[18] Jincen Jiang, Xuequan Lu, Lizhi Zhao, Richard Dazaley, and Meili Wang. Masked autoencoders in 3d point cloud representation learning. *IEEE TMM*, 2023. 2, 6

[19] Armand Joulin and Francis Bach. A convex relaxation for weakly supervised classifiers. *arXiv preprint arXiv:1206.6413*, 2012. 2

[20] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[21] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. In *CVPR*, pages 8500–8509, 2022. 1

[22] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *CVPR*, pages 4558–4567, 2018. 3

[23] Abiao Li, Chenlei Lv, Yuming Fang, and Yifan Zuo. Laptran: Transformer embedding graph laplacian for point cloud part segmentation. In *ICIP*, pages 3070–3074. IEEE, 2023. 1

[24] Abiao Li, Chenlei Lv, Guofeng Mei, Yifan Zuo, Jian Zhang, and Yuming Fang. Gstran: Joint geometric and semantic coherence for point cloud segmentation. In *ICPR*, pages 453–469. Springer, 2025. 1

[25] Jinlong Li, Cristiano Saltori, Fabio Poiesi, and Nicu Sebe. Cross-modal and uncertainty-aware agglomeration for open-vocabulary 3d scene understanding. In *CVPR*, pages 19390–19400, 2025. 1

[26] Chengxing Lin, Wenju Xu, Jian Zhu, Yongwei Nie, Ruichu Cai, and Xuemiao Xu. Patchmixing masked autoencoders for 3d point cloud self-supervised learning. *IEEE TCSVT*, 2024. 6

[27] Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. In *ECCV*, pages 657–675. Springer, 2022. 6

[28] Yun Liu, Xuefeng Yan, Zhiqi Li, Zhilei Chen, Zeyong Wei, and Mingqiang Wei. Pointgame: Geometrically and adaptively masked autoencoder on point clouds. *IEEE TGRS*, 61: 1–12, 2023. 1, 2, 6

[29] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 5

[30] Dening Lu, Qian Xie, Kyle Gao, Linlin Xu, and Jonathan Li. 3dctn: 3d convolution-transformer network for point cloud classification. *IEEE T-ITS*, 23(12):24854–24865, 2022. 1

[31] You-Wei Luo and Chuan-Xian Ren. Mot: Masked optimal transport for partial domain adaptation. In *CVPR*, pages 3531–3540. IEEE, 2023. 3

[32] Guofeng Mei, Cristiano Saltori, Fabio Poiesi, Jian Zhang, Elisa Ricci, Nicu Sebe, and Qiang Wu. Data augmentation-free unsupervised learning for 3d point cloud understanding. *arXiv preprint arXiv:2210.02798*, 2022. 1

[33] Guofeng Mei, Cristiano Saltori, Elisa Ricci, Nicu Sebe, Qiang Wu, Jian Zhang, and Fabio Poiesi. Unsupervised point cloud representation learning by clustering and neural rendering. *IJCV*, 132(8):3251–3269, 2024. 1

[34] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *ECCV*, pages 604–621. Springer, 2022. 1, 6

[35] Yu Pei, Xian Zhao, Hao Li, Jingyuan Ma, Jingwei Zhang, and Shiliang Pu. Clusterformer: Cluster-based transformer for 3d object detection in point clouds. In *ICCV*, pages 6664–6673, 2023. 1

[36] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 4

[37] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeruIPS*, 30, 2017. 6

[38] Wentao Qu, Yuantian Shao, Lingwu Meng, Xiaoshui Huang, and Liang Xiao. A conditional denoising diffusion probabilistic model for point cloud upsampling. In *CVPR*, pages 20786–20795, 2024. 1

[39] Wentao Qu, Jing Wang, YongShun Gong, Xiaoshui Huang, and Liang Xiao. An end-to-end robust point cloud semantic segmentation network with single-step conditional diffusion models. In *CVPR*, pages 27325–27335, 2025. 1

[40] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *ICML*, pages 8821–8831. Pmlr, 2021. 2

[41] Ayumu Saito, Prachi Kudeshia, and Jiju Poovvancheri. Point-jepa: A joint embedding predictive architecture for self-supervised learning on point cloud. *arXiv preprint arXiv:2404.16432*, 2024. 6

[42] Kunming Su, Qiuxia Wu, Panpan Cai, Xiaogang Zhu, Xuequan Lu, Zhiyong Wang, and Kun Hu. Ri-mae: Rotation-invariant masked autoencoders for self-supervised point cloud representation learning. *arXiv preprint arXiv:2409.00353*, 2024. 1, 5, 6

[43] Linzhi Su, Mengna Yang, Jie Liu, Xingxing Hao, Chenyang Zhang, and Xin Cao. Point-msd: Jointly mamba self-supervised self-distilling point cloud representation learning. In *ICSP*, pages 310–316. IEEE, 2024. 1, 2, 6

[44] Yuan Tang, Xianzhi Li, Jinfeng Xu, Qiao Yu, Long Hu, Yixue Hao, and Min Chen. Point-lgmask: Local and global contexts embedding for point cloud pre-training with multi-ratio masking. *IEEE TMM*, 2023. 1, 2, 6

[45] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *CVPR*, pages 3405–3414, 2019. 7

[46] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *NeurIPS*, 30, 2017. 2, 4

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017. 6

[48] Chuxin Wang, Yixin Zha, Jianfeng He, Wenfei Yang, and Tianzhu Zhang. Rethinking masked representation learning for 3d point cloud understanding. *IEEE TIP*, 2024. 1, 2, 3, 6

[49] Siming Yan, Yuqi Yang, Yuxiao Guo, Hao Pan, Peng-shuai Wang, Xin Tong, Yang Liu, and Qixing Huang. 3d feature prediction for masked-autoencoder-based point cloud pre-training. *arXiv preprint arXiv:2304.06911*, 2023. 6

[50] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *ICCV*, pages 12498–12507, 2021. 1

[51] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. Pointr: Diverse point cloud completion with geometry-aware transformers. In *ICCV*, pages 12498–12507, 2021. 7

[52] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-BERT: Pre-training 3d point cloud transformers with masked point modeling. In *CVPR*, pages 19313–19322, 2022. 2, 6

[53] Karim Abou Zeid, Jonas Schult, Alexander Hermans, and Bastian Leibe. Point2vec for self-supervised representation learning on point clouds. In *GCPR*, pages 131–146. Springer, 2023. 1, 2, 5, 6, 7

[54] Yaohua Zha, Huizhen Ji, Jinmin Li, Rongsheng Li, Tao Dai, Bin Chen, Zhi Wang, and Shu-Tao Xia. Towards compact 3d representations via point feature enhancement masked autoencoders. In *AAAI*, pages 6962–6970, 2024. 2

[55] Renrui Zhang, Ziyu Guo, Peng Gao, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, and Hongsheng Li. Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. *NeurIPS*, 35:27061–27074, 2022. 2, 6

[56] Xiangdong Zhang, Shaofeng Zhang, and Junchi Yan. Pcp-mae: Learning to predict centers for point masked autoencoders. *arXiv preprint arXiv:2408.08753*, 2024. 1, 6

[57] 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. 3

[58] Junsheng Zhou, Xin Wen, Baorui Ma, Yu-Shen Liu, Yue Gao, Yi Fang, and Zhizhong Han. 3d-oae: Occlusion autoencoders for self-supervised learning on point clouds. In *ICRA*, pages 15416–15423, 2024. 1