LML-MAE: A Local-Aware Multi-Branch Lightweight Framework for Masked AutoEncoder

Ziqi Xi, XiaoLiang Gong
School of Computer Science and Technology, TongJi University
Email: {2432168, gxllshsh}@tongji.edu.cn

***Abstract*—Masked autoencoders (MAEs) have been widely adopted for self-supervised learning in point cloud understanding. However, existing methods often fail to sufficiently capture local structural information. While multi-scale frameworks enhance local geometry at different scales and cross-modal approaches leverage auxiliary data, both introduce significant computational overhead, and cross-modal data are often difficult to obtain. To address these limitations, we propose a Lightweight Multi-branch Local feature encoding Masked Autoencoder (LML-MAE) for point cloud representation learning. Our framework employs a multi-branch local feature extraction module based on depthwise separable convolutions to achieve expressive local encoding with reduced complexity. To prevent information leakage during masked reconstruction, a center position prediction module is introduced to enforce more effective pretraining objectives. Moreover, we design a local structural information embedding module that dynamically integrates point-wise neighborhood structural information into point features at the patch level, enabling more expressive local geometry modeling. Compared with existing multi-scale and cross-modal approaches, LML-MAE achieves lower computational cost and faster pretraining convergence. Extensive experiments on downstream tasks demonstrate that our method consistently outperforms the baseline Point-MAE model, achieving 5.67%, 5.47%, and 5.41% performance gains on three variants of the challenging ScanObjectNN dataset.**

***Index Terms*—3D Point cloud, Self supervised learning, MAE**

I. Introduction

Self-supervised learning (SSL) [20], [21] has achieved remarkable success in representation learning for images and text, particularly with masked autoencoder (MAE) paradigms. By reconstructing missing content from partial observations, MAE learns high-level representations without annotations. With the increasing importance of point cloud data in applications such as autonomous driving, 3D reconstruction, and robotics, extending SSL to point cloud understanding has attracted growing attention.

Point-MAE [7] pioneers the application of MAE to point clouds by masking point patches and reconstructing them using a Transformer-based encoder–decoder. Despite its effectiveness, it has two limitations. First, Transformers lack explicit inductive bias for capturing fine-grained local geometric structures [15], [16]. Second, directly using ground-truth patch centers in the decoder may cause positional information leakage, weakening representation learning [5].

To alleviate the lack of local geometric modeling, several subsequent works [1], [2], [6] adopt multi-scale feature learning frameworks to enhance local structure representation.

TABLE I
Comparisons with existing MAE-based methods in terms of pre-training efficiency and downstream performance.

Methods	Single/ Cross-Modal	Pre-trained Model Needed	# Params	GFLOPS	Time (h)	ScanObjectNN Performance
Point-MAE [7]	Single	✗	29.0 (baseline)	2.0 (baseline)	8.7 (baseline)	85.18 (baseline)
Point-M2AE [2]	Single	✗	15.3 (0.53×)	3.2 (1.6×)	19.1 (2.2×)	86.43 (↑1.25)
Point-FEMAE [6]	Single	✗	41.5 (1.43×)	4.4 (2.2×)	32.2 (3.7×)	90.22 (↑5.04)
ACT [8]	Cross	✓	135.5 (4.67×)	23.0 (13.5×)	34.8 (4.0×)	88.21 (↑3.03)
I2P-MAE [9]	Cross	✓	74.9 (2.58×)	14.6 (7.3×)	42.6 (4.9×)	90.11 (↑4.93)
ReCon [10]	Cross	✓	140.9 (4.85×)	18.2 (9.1×)	44.4 (5.1×)	90.63 (↑5.45)
LML-MAE	Single	✗	30.1 (1.04×)	2.6 (1.3×)	11.3 (1.3×)	90.59 (↑5.41)

Although effective, these methods typically introduce substantial computational overhead and significantly increase model complexity, which is undesirable for large-scale pre-training and limits their practical applicability.

Another line of research addresses the position leakage issue by delaying the use of masked tokens to the decoder, as in Point-MAE and its variants. However, directly providing the true center positions of masked patches to the decoder still allows the reconstruction task to be solved with limited reliance on encoder representations, thereby reducing the effectiveness of self-supervised pre-training. Introducing center position prediction [3] into the encoding process has been shown to mitigate this issue, but existing approaches remain constrained by insufficient local geometric modeling capability.

To address the above challenges, we propose **LML-MAE**, a lightweight masked autoencoder framework for point cloud self-supervised learning. LML-MAE is designed to jointly enhance local geometric representation, reduce computational overhead, and prevent positional information leakage during pre-training. Specifically, our contributions are summarized as follows:

- We introduce a local structural information embedding module that explicitly captures fine-grained geometric relationships within point cloud patches.
- We propose a lightweight multi-branch local feature encoding module that enables effective local feature learning with significantly reduced computational cost.
- We incorporate a center prediction module into the encoder to alleviate position leakage, encouraging the encoder to learn more discriminative semantic representations.

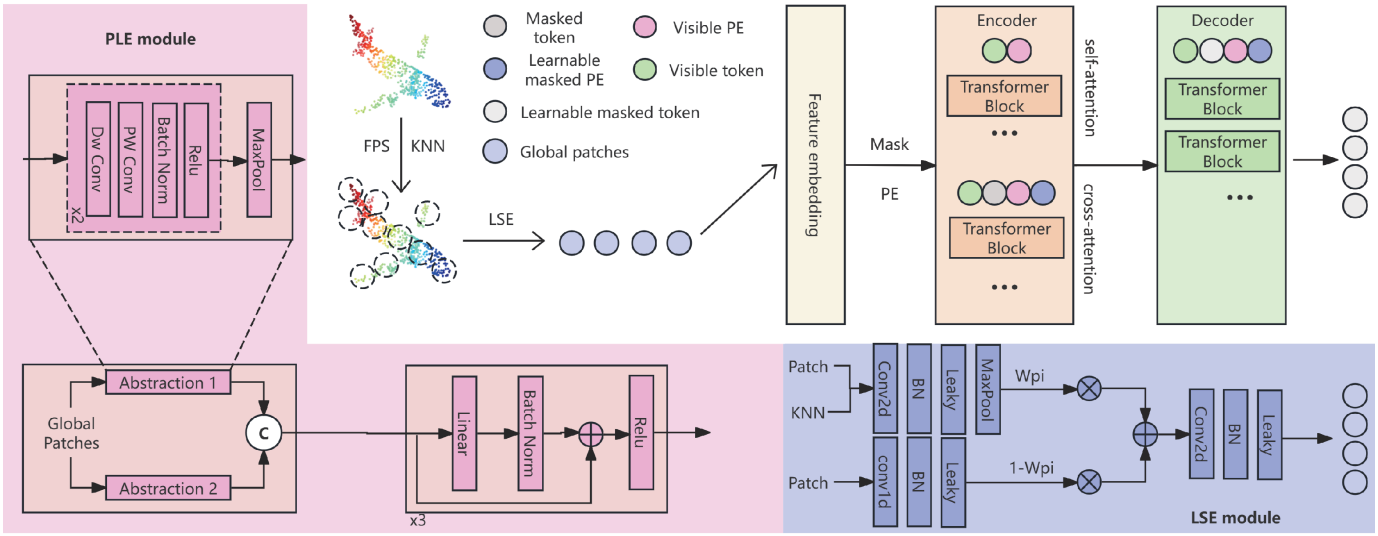


Fig. 1. Overview of the proposed LML-MAE framework. LML-MAE enhances masked point cloud pretraining by introducing a point-level local structure enhancement module (LSE) and a patch-level lightweight feature embedding module (PLE), with a center prediction module (PCM) applied as an auxiliary constraint to mitigate positional information leakage.

II. Method

A. Patch Generation and Feature Embedding

1) Patch Generation: We employ the Farthest Point Sampling (FPS) method to sample n central points from the point cloud $X \in \mathcal{R}^{N\times 3}$. Then, using k-nearest neighbors (KNN), we select k points near each center to form a patch P, constructing a neighborhood.

$$P = \mathrm{KNN}(\mathrm{FPS}(X), X), \quad P \in \mathrm{R}^{n\times k\times 3} \tag{1}$$

2) Local Structure Embedding: The Local Structure Embedding (LSE) module operates at the point level within each patch by explicitly modeling point-wise neighborhood structures before patch aggregation. Its goal is to generate more discriminative point features by capturing local structural information of each point within a patch while preserving global shape cues.

The LSE module consists of three stages: Neighborhood Feature Learning (NFL), Self Feature Learning (SFL), and Dynamic Feature Fusion (DFF).

In the NFL stage, edge features e_{ij} are constructed within each point's local region by calculating the differences between each point X_i and its j-th nearest neighbors. A Max-Pooling layer then aggregates these point-wise edge features into a single feature F_{e_i}.For each point $X_i \in \mathbb{R}^3$, we define its local neighborhood $\mathcal{N}(i)$ using k-nearest neighbors.The edge feature between point X_i and its j-th neighbor X_j is defined as

$$e_{ij} = \psi(X_j - X_i), \tag{2}$$

where $\psi(\cdot)$ denotes a shared MLP. The aggregated neighborhood feature for point X_i is obtained via max pooling:

$$\mathbf{F}_{e_i} = \max_{j\in\mathcal{N}(i)} e_{ij}. \tag{3}$$

During the SFL stage, X_i are processed into point-level features F_{p_i}, extracting global shape information. This global information reflects the geometric shape of the entire point cloud object, not just local structure. In parallel, each point X_i is mapped to a self feature through a shared transformation:

$$\mathbf{F}_{p_i} = \phi(X_i), \tag{4}$$

where $\phi(\cdot)$ denotes a shared MLP applied to each point.

During the DFF stage, an attention vector W_{p_i} is generated through a transformation network Φ and a sigmoid activation function to control the fusion degree of F_{e_i} and F_{p_i}. Finally, weighted fusion produces point features embedded with local structural information.

$$W_{p_i} = \sigma\big(\varphi(\mathbf{F}_{e_i} + \mathbf{F}_{p_i})\big), \tag{5}$$

$$\mathbf{F}'_{p_i} = W_{p_i} \odot \mathbf{F}_{p_i} + (1 - W_{p_i}) \odot \mathbf{F}_{e_i}, \tag{6}$$

3) Patch Feature Embedding: Unlike previous studies that employed lightweight PointNet [11] for feature extraction, we redesign a novel lightweight multi-branch feature embedding module (PLE) that aggregates structure-enhanced point features into compact patch tokens, providing efficient and expressive inputs for the Transformer encoder. By leveraging a multi-branch design with depthwise separable convolutions, the proposed PLE significantly reduces computational complexity while preserving the capacity to model neighborhood geometric relationships within each patch.

Mapping and Aggregation. Given point features $\{\mathbf{F}'_{p_i}\}_{i=1}^{k}$ within a patch, two lightweight convolutional layers are first applied to map each point into a higher-dimensional feature space:

$$\mathbf{H}_{p_i} = \phi(\mathbf{F}'_{p_i}), \tag{7}$$

where $\phi(\cdot)$ denotes a shared convolutional transformation with normalization and non-linear activation. To obtain a

TABLE II
CLASSIFICATION ACCURACY ON SCANOBJECTNN AND MODELNET40.

Methods	References	ScanObjectNN				ModelNet40		
		Input	OBJ_BG	OBJ_ONLY	PB_T50_RS	Input	w/o Vote	w/ Vote
Supervised Learning Only								
PointNet [11]	CVPR 2017	1k Points	73.3	79.2	68.0	1k Points	89.2	–
PointNet++ [12]	NeurIPS 2017	1k Points	82.3	84.3	77.9	1k Points	90.7	–
with Single-Modal Self-Supervised Learning								
Point-BERT [17]	CVPR 2022	2k Points	87.43	88.12	83.07	1k Points	92.7	93.2
Point-MAE [7]	ECCV 2022	2k Points	90.02	88.29	85.18	1k Points	93.2	93.8
Point-M2AE [2]	NeurIPS 2022	2k Points	91.22	88.28	86.43	1k Points	93.4	94.0
Point-FEMAE [6]	AAAI 2024	2k Points	95.18	93.29	90.22	1k Points	94.0	**94.5**
Point-UMAE [1]	ICASSP 2025	2k Points	91.57	88.98	86.95	1k Points	93.7	94.2
Point-DMAE [4]	CIKM 2025	2k Points	94.15	93.46	89.35	1k Points	93.4	–
LML-MAE	–	2k Points	**95.69**	**93.76**	**90.59**	1k Points	93.9	94.2
Improve (baseline: Point-MAE)	–	–	+5.67	+5.47	+5.41	–	+0.7	+0.4
with Cross-Modal Self-Supervised Learning								
ACT [8]	ICLR 2023	2k Points	93.29	91.91	88.21	1k Points	93.2	93.7
I2P-MAE [9]	IJCAI 2023	2k Points	94.84	91.86	86.07	1k Points	93.7	94.0
TAP [24]	ICCV 2023	2k Points	90.36	89.50	85.67	–	–	–
ReCon [10]	ICML 2023	2k Points	95.18	93.63	**90.63**	1k Points	**94.1**	**94.5**

permutation-invariant patch-level representation, max pooling is applied along the point dimension:

$$\mathbf{F}_{\text{patch}} = \text{MaxPool}\big(\{\mathbf{H}_{p_i}\}_{i=1}^{k}\big). \tag{8}$$

To reduce computational cost, standard convolutions are replaced with depthwise separable convolutions, that is DSConv [13], [14].

DSConv decomposes a standard convolution with kernel size $k\times k$ into a depthwise convolution followed by a pointwise 1×1 convolution. The parameter complexity is reduced from

$$k^2 C_{\text{in}} C_{\text{out}} \tag{9}$$

to

$$k^2 C_{\text{in}} + C_{\text{in}} C_{\text{out}}, \tag{10}$$

leading to significantly fewer parameters and multiply–accumulate operations while preserving representation capacity.

Unlike image convolutions that operate on dense grids, point cloud patches are sparse and unordered. The proposed DSConv-based multi-branch design allows efficient aggregation of local point features while avoiding over-parameterization, which is particularly beneficial under high masking ratios.

Multi-branch Fusion We experimented with mapping branches of varying dimensions and their combinations. Ultimately, two parallel branches capture geometric features under distinct receptive fields: one outputs low-dimensional features, the other high-dimensional features. After concatenation, these feature streams undergo residual MLP fusion, progressively mapping to the target dimension. The multi-branch feature embedding is defined as

$$\mathbf{F}_{ple} = \text{MLP}\big(\text{Concat}(\mathbf{F}_{b_1}, \mathbf{F}_{b_2})\big), \tag{11}$$

where $\mathbf{F}_{b_1}$ and $\mathbf{F}_{b_2}$ denote features extracted from two branches with different receptive fields.

4) Masking and Position Embedding: The global patch set is randomly divided into visible patches P_v and masked patches P_m according to a specified masking ratio m. Following MAE, we adopt learnable positional embeddings to encode patch positions, which are added to patch embeddings before being fed into the Transformer encoder.

Specifically, each patch is associated with a D-dimensional positional embedding corresponding to its spatial location, enabling the model to capture relative positional information among patches.

B. Encoder

A standard Transformer architecture [19] is adopted for the Encoder. Let T_v denote the visible patch tokens after random masking, T_m denote the mask patch tokens, and PE_v denote their corresponding position embeddings. The encoder applies stacked Transformer blocks to model contextual relationships among visible patches via self-attention. Specifically, visible tokens are augmented with their positional embeddings and processed through self-attention layers, producing the encoded visible representations E_v

$$E_v = \text{Transformer}(T_v + PE_v), \tag{12}$$

where each Transformer block consists of a multi-head self-attention module, a feed-forward network, and residual connections with pre-normalization.

During encoding, masked patches do not have access to their corresponding ground-truth positional embeddings. Instead, masked tokens T_m are combined with randomly initialized learnable positional embeddings PE, which serve as initial

positional priors for masked patches. These masked representations are used to construct queries, while both visible and masked tokens jointly serve as keys and values in the cross-attention operation, enabling masked patches to aggregate contextual information from the entire token set. Specifically, the query is constructed as

$$Q = T_m + PE_m^l, \tag{13}$$

and the key and value are constructed as:

$$K = V = \text{Concat}[T_v + PE_v, T_m + PE_m^l]. \tag{14}$$

The PCM predicts the center position embeddings for masked patches through cross-attention:

$$\tilde{PE}_m^l = \text{CrossAttn}(Q, K, V), \tag{15}$$

where $\tilde{PE}_m^l$ denotes the refined positional representations of masked patches inferred from both visible and masked contextual information.

C. Decoder

The decoder aligns with MAE series research, employing a standard Transformer architecture for point cloud reconstruction. It takes as input: encoded tokens of visible patch E_v, corresponding position embeddings PE_v of the visible patches, learnable masked tokens T_m^l, and predicted masked position embeddings $\tilde{PE}_m^l$. Subsequently, the decoder feeds the output masked tokens to the reconstruction head. The reconstruction head processes the decoded tokens via a linear projection layer to reconstruct the point cloud.

$$H_m = \text{D}\Big(\text{concat}[T_m^l, E_v],\ \text{concat}[\text{sg}(\tilde{PE}_m^l), PE_v]\Big) \tag{16}$$

$$R_m = \text{Reshape}\big(\text{Projection}(H_m)\big) \tag{17}$$

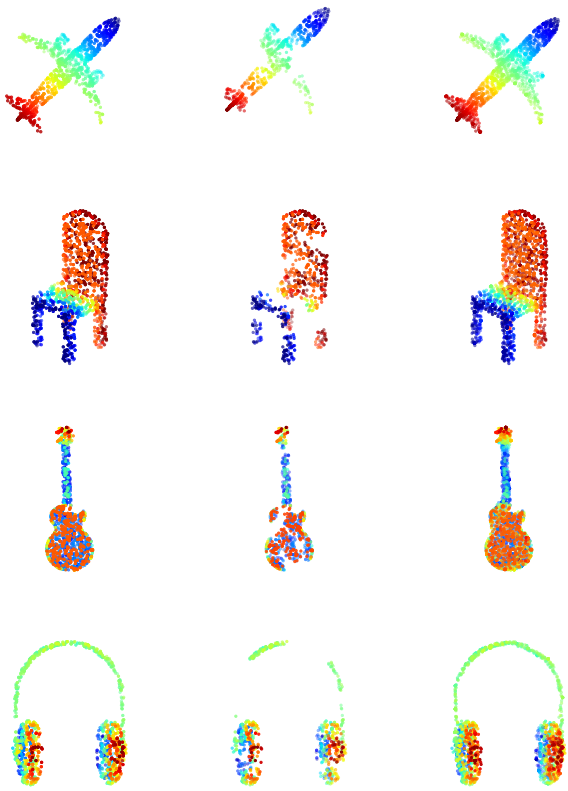
The loss function L comprises two components: the prediction loss for the center position embedding of masked patches L_{PC} and the reconstruction loss for the point coordinates of masked patches L_{Rec}. $L = \alpha \cdot L_{\text{PC}} + L_{\text{Rec}}$, α is the scaling factor.

The L_{PC} is computed using an L_2 loss function, measuring the difference between the predicted position encoding of the masked patch center and the true position encoding, where n denotes the number of patches, m denotes the masking ratio, and D denotes the hidden dimension of the Transformer.

$$L_{\text{PC}} = \frac{1}{mnD} \left\| \tilde{PE}_m^l - PE_m \right\|_2^2 \tag{18}$$

The L_{Rec} is computed using an L_2 Chamfer Distance loss function, measuring the difference between the reconstructed point coordinates of the masked patch and the true point coordinates.

$$L_{\text{Rec}} = \frac{1}{|R_m|} \sum_{a \in R_m} \min_{b \in P_m} \|a - b\|_2^2 + \frac{1}{|P_m|} \sum_{b \in P_m} \min_{a \in R_m} \|a - b\|_2^2 \tag{19}$$



(a) Input (b) visible points (c) Reconstruction

Fig. 2. Reconstruction results on ShapeNet. (a) Ground truth, (b) visible part of the ground truth, and (c) reconstructed point cloud.

III. Experiments

To evaluate the effectiveness of LML-MAE, we conduct experiments on multiple downstream tasks, including object classification, few-shot classification, and part segmentation, covering both global and local geometric understanding.

For classification, we use the ScanObjectNN and ModelNet40 datasets with standard evaluation metrics. For segmentation, we evaluate on ShapeNetPart following common protocols. All experiments adopt a unified training strategy to ensure fair comparisons.

A. Pre-training

All models are pre-trained on the ShapeNet dataset, which contains over 51,000 distinct 3D CAD models across 55 object categories. For each shape, 1024 points are sampled using Farthest Point Sampling (FPS) as the pre-training input. The sampled points are grouped into local patches using KNN, where each patch contains 32 points, resulting in 64 patches per point cloud.

The pre-training process follows the masked autoencoding paradigm, where a subset of patches is randomly masked and reconstructed. Figure 2 visualizes representative reconstruction results, demonstrating that the proposed method effectively captures both global structure and local geometric details. In

addition, Table I reports the pre-training efficiency comparison, showing that LML-MAE achieves favorable computational efficiency compared to existing MAE-based methods while maintaining strong representation learning capability.

B. Fine-Tuning for Downstream Tasks

1) Object Classification: We evaluate classification performance on ScanObjectNN and ModelNet40.

ScanObjectNN is a challenging real-world dataset consisting of approximately 15,000 point cloud samples collected from indoor scenes, featuring cluttered backgrounds, occlusions, and partial observations. Following standard practice, experiments are conducted on three commonly used variants: OBJ-BG, OBJ-ONLY, and PB-T50-RS, with each point cloud uniformly sampled to 2048 points.

As shown in Table II, LML-MAE consistently outperforms existing single-modal methods across all three ScanObjectNN variants. In particular, it obtains the highest accuracy on OBJ-BG and OBJ-ONLY among all compared methods, and delivers competitive performance on the most challenging PB-T50-RS variant, outperforming the baseline Point-MAE by 5.67%, 5.47%, and 5.41%, respectively.

For ModelNet40, which contains over 12,000 CAD models across 40 object categories, each sample is represented by 1024 points. We report overall classification accuracy with and without voting. As shown in Table II, LML-MAE consistently improves upon Point-MAE by 0.7% without voting and 0.4% with voting, demonstrating its strong generalization ability on clean synthetic data.

2) Few-shot learning: We further evaluate generalization under limited supervision on ModelNet40 using the standard n-way m-shot setting. Specifically, n categories are randomly sampled, and m labeled samples per category are provided as the support set. Each experimental setting is repeated 10 times with different random splits, and the mean accuracy and standard deviation are reported.

The results are summarized in Table III. LML-MAE consistently outperforms the baseline Point-MAE across all evaluated settings, achieving improvements of 1.2%, 0.9%, 1.35%, and 0.75% under 5-way and 10-way configurations. These results indicate that the proposed pre-training strategy enables the model to learn more transferable and discriminative representations, particularly in low-data regimes.

3) Part Segmentation: We further evaluate LML-MAE on the ShapeNetPart dataset for fine-grained geometric understanding. ShapeNetPart contains 16,881 shapes from 16 categories with part-level annotations. Following Point-MAE, we sample 2048 points per shape and evaluate segmentation performance using class-average IoU (Cls.mIoU) and instance-average IoU (Inst.mIoU).

As reported in Table IV, LML-MAE achieves a 0.6% improvement in Cls.mIoU over Point-MAE, while maintaining comparable Inst.mIoU performance. This demonstrates that our method enhances global representations while preserving fine-grained local geometry for dense prediction tasks.

TABLE III
FEW-SHOT CLASSIFICATION RESULTS ON MODELNET40 UNDER 5-WAY AND 10-WAY SETTINGS.

Methods	5-way		10-way	
	10-shot	20-shot	10-shot	20-shot
Supervised Learning Only				
PointNet [11]	52.0±3.8	57.8±4.9	46.6±4.3	35.2±4.8
OcCo [18]	90.6±2.8	92.5±1.9	82.9±1.3	86.5±2.2
with Single-Modal Self-Supervised Representation Learning				
Point-BERT [17]	94.6±3.1	96.3±2.7	91.0±5.4	92.7±5.1
Point-MAE [7]	96.3±2.5	97.8±1.7	92.6±4.1	95.0±3.0
Point-M2AE [2]	96.8±1.8	98.3±1.4	93.2±4.5	95.0±3.0
Point-FEMAE [6]	97.2±1.9	98.6±1.1	**94.0±3.3**	95.8±2.8
Point-UMAE [1]	97.1±1.9	98.6±0.7	92.6±4.0	95.1±3.0
Point-DMAE [4]	96.8±2.4	98.5±0.7	93.6±3.7	95.7±3.2
LML-MAE	**97.5±2.1**	98.7±1.9	93.9±3.7	95.8±3.2
Improve (baseline: Point-MAE)	+1.2	+0.9	+1.3	+0.8
with Cross-Modal Self-Supervised Representation Learning				
ACT [8]	96.8±2.3	98.0±1.4	93.3±4.0	95.6±2.8
I2P-MAE [9]	97.0±1.8	98.3±1.3	92.5±3.5	95.0±3.0
TAP [24]	97.4±1.9	**98.7±1.1**	93.2±4.6	95.7±2.8
ReCon [10]	97.3±1.9	98.7±1.2	93.3±3.9	**95.8±2.8**

TABLE IV
PART SEGMENTATION RESULTS (CLS.MIOU / INST.MIOU) ON SHAPENET-PART.

Methods	Cls.mIoU	Inst.mIoU
Supervised Learning Only		
PointNet [11]	80.4	83.7
PointNet++ [12]	81.9	85.1
Single-Modal Self-Supervised Representation Learning		
Transformer [19]	83.4	84.7
Point-BERT [17]	84.1	85.6
Point-MAE [7]	84.2	86.1
Point-FEMAE [6]	**84.9**	86.3
Point-UMAE [1]	84.6	86.1
Point-DMAE [4]	84.8	86.2
LML-MAE	84.8	86.1
Improve (baseline: Point-MAE)	+0.6	+0.0
Cross-Modal Self-Supervised Representation Learning		
ACT [8]	84.7	86.1
ReCon [10]	84.8	**86.4**

C. Ablation Studies

We conduct ablation studies on the PB-T50-RS variant of the ScanObjectNN to evaluate the proposed modules in LML-MAE. Compared with the baseline Point-MAE, our method introduces three key modules: Local Structural Information Embedding (LSE), Lightweight Multi-branch Local Feature Encoding based on Depthwise Separable Convolution (DSConv), and the Center Position Prediction Module (PCM).

As shown in Table V, each individual module improves performance over the baseline, and the performance gains are not merely additive when multiple modules are combined, suggesting that these components interact in a complementary manner. The best performance is achieved when all are jointly applied.

These improvements arise from their complementary roles. The DSConv-based PLE module enhances fine-grained local

TABLE V
ABLATION STUDY ON USING DEPTHWISE SEPARABLE CONVOLUTION, PCM AND LSE.

Using DSConv	PCM	LSE	PB T50 RS
✗	✗	✗	88.69
✗	✓	✗	89.31
✓	✗	✗	89.42
✗	✗	✓	89.04
✓	✗	✓	89.66
✗	✓	✓	89.59
✓	✓	✗	90.18
✓	✓	✓	90.59

feature extraction with minimal computational overhead. The LSE module explicitly injects neighborhood structural information at the patch level, enabling the encoder to better capture local geometric relationships. Meanwhile, the PCM introduces a structure-aware pretraining signal, which encourages the model to leverage global context while mitigating information leakage. Together, they operate across feature extraction, structural embedding, and positional reasoning, forming an efficient and unified framework.

IV. CONCLUSION

In this paper, we present LML-MAE, a lightweight masked autoencoder for point cloud representation learning. The framework enhances local geometric modeling by dynamically integrating patch-level neighborhood structural information into point features, while employing a multi-branch local feature encoding module built with efficient depthwise separable convolutions. A center position prediction strategy is further introduced to improve masked pretraining by mitigating information leakage. Extensive experimental results demonstrate that LML-MAE consistently achieves superior performance over the Point-MAE baseline across multiple downstream tasks, while significantly reducing computational overhead and accelerating the pretraining process. In future work, we plan to investigate efficiency-aware integration of complementary modalities, aiming to further improve robustness and representation quality without sacrificing the lightweight nature of the proposed framework.

REFERENCES

[1] H. Zeng, P. Zhang, F. Li, T. Ye, J. Wang, and X. Yang, "Point-UMAE: Unet-like Masked Autoencoders for Point Cloud Self-supervised Learning," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 2025, pp. 1–5.

[2] R. Zhang, Z. Guo, P. Gao, R. Fang, B. Zhao, D. Wang, Y. Qiao, and H. Li, "Point-m2ae: Multi-scale masked autoencoders for hierarchical point cloud pre-training," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 27061–27074, 2022.

[3] X. Zhang, S. Zhang, and J. Yan, "PCP-MAE: Learning to predict centers for point masked autoencoders," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 37, pp. 80303–80327, 2024.

[4] X. Jin, Z. Wang, W. Zheng, and F. Nie, "Point-DMAE: Point cloud self-supervised learning via density-directed masked autoencoders," in *Proc. ACM Int. Conf. on Information and Knowledge Management (CIKM)*, 2025, pp. 1231–1238.

[5] X. Xiao, R. Yao, Z. Tian, and S. Du, "Point-MaDi: Masked autoencoding with diffusion for point cloud pre-training," in *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[6] Y. Zha, H. Ji, J. Li, R. Li, T. Dai, B. Chen, Z. Wang, and S.-T. Xia, "Towards compact 3D representations via point feature enhancement masked autoencoders," in *Proc. AAAI Conf. on Artificial Intelligence (AAAI)*, vol. 38, no. 7, 2024, pp. 6962–6970.

[7] Y. Pang, W. Wang, F. E. H. Tay, W. Liu, Y. Tian, and L. Yuan, "Masked autoencoders for point cloud self-supervised learning," in *Computer Vision – ECCV 2022*, 2022, pp. 604–621.

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

[9] R. Zhang, L. Wang, Y. Qiao, P. Gao, and H. Li, "Learning 3D representations from 2D pre-trained models via image-to-point masked autoencoders," in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 21769–21780.

[10] Z. Qi, R. Dong, G. Fan, Z. Ge, X. Zhang, K. Ma, and L. Yi, "Contrast with reconstruct: Contrastive 3D representation learning guided by generative pretraining," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2023, pp. 28223–28243.

[11] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 652–660.

[12] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint* arXiv:1704.04861, 2017.

[14] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.

[15] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, and K. Xu, "Geometric transformer for fast and robust point cloud registration," in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 11143–11152.

[16] H. Yu, Z. Qin, J. Hou, M. Saleh, D. Li, B. Busam, and S. Ilic, "Rotation-invariant transformer for point cloud matching," in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 5384–5393.

[17] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-BERT: Pre-training 3D point cloud transformers with masked point modeling," in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 19313–19322.

[18] H. Wang, Q. Liu, X. Yue, J. Lasenby, and M. J. Kusner, "Unsupervised point cloud pre-training via occlusion completion," in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2021, pp. 9782–9792.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[20] J. Gui, T. Chen, J. Zhang, Q. Cao, Z. Sun, H. Luo, and D. Tao, "A survey on self-supervised learning: Algorithms, applications, and future trends," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 12, pp. 9052–9071, 2024.

[21] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer, "S4L: Self-supervised semi-supervised learning," in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2019, pp. 1476–1485.

[22] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 32, 2019.

[23] V. Rani, S. T. Nabi, M. Kumar, A. Mittal, and K. Kumar, "Self-supervised learning: A succinct review," *Archives of Computational Methods in Engineering*, vol. 30, no. 4, pp. 2761–2775, 2023.

[24] Z. Wang, X. Yu, Y. Rao, J. Zhou, and J. Lu, "Take-a-Photo: 3D-to-2D generative pre-training of point cloud models," in *Proc. IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, 2023, pp. 5640–5650.