Point Ladder Tuning: Parameter-Efficient Hierarchical Adaptation for 3D Point Cloud Understanding

Junlin Chang[1,2], Longhao Zou[2], and Rui Li[2†]

[1] Beihang University, Beijing
[2] Pengcheng Laboratory, Shenzhen
changjunlin@buaa.edu.cn {zoulh,lir01}@pcl.ac.cn

Abstract. Fine-tuning pre-trained point-cloud backbones typically updates all parameters, resulting in substantial computation and memory overhead. More importantly, modern point backbones rely on aggressive tokenization and downsampling, which yields compact global tokens but irreversibly discards fine-grained local geometry, an inherent bottleneck for parameter-efficient adaptation. Consequently, existing PEFT methods that operate only on these coarsened tokens can modulate global semantics but struggle to recover the missing multi-scale locality. We present Point Ladder Tuning (PLT), a locality-aware PEFT framework that performs hierarchical, instance-conditioned adaptation while keeping the backbone frozen. PLT forms a lightweight closed loop: (i) a Hierarchical Ladder Network (HLN) constructs a multi-resolution local feature pyramid directly from raw points; (ii) a Local–Global Fusion (LGF) aligns and fuses local pyramids with intermediate backbone semantics; and (iii) a Dynamic Prompt Generator produces instance-aware multi-scale prompts to modulate the frozen backbone effectively. For dense prediction, we further introduce a lightweight segmentation head that progressively upsamples fused features and leverages backbone priors to refine fine structures. Extensive experiments on classification and dense prediction show that PLT consistently surpasses prior PEFT baselines with minimal tunable parameters. PLT achieves state-of-the-art performance using only 2.71% trainable parameters for classification and 7.69% for dense prediction, and scales favorably to larger backbones, requiring merely 0.36% parameters on PointGPT-L.

Keywords: 3D Point Cloud Understanding · Parameter-Efficient Fine-Tuning

1 Introduction

The rapid advancement of 3D sensing technologies, such as Light Detection and Ranging (LiDAR), depth cameras (structured light/time-of-flight), photogrammetry, and stereo vision, has led to the widespread availability of point cloud

† Corresponding author.

Table 1: Locality preservation after tokenization. Cov@2r measures raw-point coverage by FPS token centers within 2r-radius neighborhoods, where r denotes the average nearest-neighbor distance among raw points. Boundary edge collapse is the fraction of cross-label raw KNN edges collapsed into the same token region. Token-Adj. R@k measures whether raw KNN edges remain connected within a token-center KNN graph, where k is the number of nearest token neighbors.

Dataset	Cov@2r (%)	Boundary Edge Coll. (%)	Token-Adj. R@1 (%)	Token-Adj. R@8 (%)
ScanObjectNN	22.64	N/A	58.96	99.31
ShapeNetPart	28.12	34.93	57.71	98.66
S3DIS	21.26	36.72	59.06	99.67

data, driving the development of learning-based 3D imaging and point cloud understanding methods. Applications span a variety of domains, including autonomous driving [9,28,39], virtual/augmented reality [5,16], and robotics [8,11, 33]. Compared to grid-structured images, point clouds are sparse and unordered, exhibit non-uniform sampling and varying densities, and lack explicit geometric connectivity. These properties demand representations that simultaneously preserve fine-grained local geometry and maintain coherent global context, making naive extensions of 2D architectures insufficient in practice.

To address these challenges, the community has developed point-specific architectures [20, 25–27, 35, 36] and increasingly relies on self-supervised pretraining [1, 4, 10, 13, 17, 24, 41, 44] to learn transferable 3D features. However, deploying such pre-trained backbones typically requires full fine-tuning, i.e., updating all parameters for every downstream task. This paradigm is often impractical due to: (i) overfitting and catastrophic forgetting, (ii) storage overhead from maintaining task-specific model copies, and (iii) substantial compute and memory cost.

Parameter-Efficient Fine-Tuning (PEFT) mitigates these issues by freezing pre-trained weights and optimizing only a small set of additional parameters, such as adapters [18], prompt tuning [19], and ladder-style tuning [29]. While highly effective in NLP and vision, directly applying these techniques to point-cloud backbones often yields sub-optimal results. As shown in Tab. 1,modern point cloud pretrained backbones rely on aggressive tokenization and downsampling of raw points (e.g., 2048 $\rightarrow$ 128), which preserve global semantics but inevitably loss fine-grained, multi-scale local geometry. Since most PEFT methods modulate only these coarsened tokens, they can adapt global semantics but have limited capacity to recover the missing locality that is critical for recognition and dense prediction. A natural idea is to introduce a local branch that extracts geometric details from raw points. However, local cues alone are typically insufficient for complex 3D understanding because they lack high-level semantic context. Without effectively integrating local geometry with the backbone's global priors, the additional branch brings limited gains.

To this end, we propose **Point Ladder Tuning (PLT)**, a novel locality-aware PEFT framework specifically designed for point cloud learning. Specially,

We decouple the adaptation process into two tightly coupled pathways. First, a lightweight **Hierarchical Ladder Network (HLN)** constructs a multi-resolution pyramid of geometric features directly from raw points, preserving spatial granularity across scales. In parallel, a dynamic-prompt-based global adaptation pathway converts the constructed multi-scale cues into instance-aware prompts and injects them back into the frozen backbone, enabling task-aware refinement without full retraining. These two pathways are coupled through a **Local–Global Fusion (LGF)** module: LGF selectively fuses the fine-grained local cues extracted by HLN with intermediate pretrained representations from the backbone, and then feeds back the resulting multi-scale signals to modulate self-attention and global feature learning inside the frozen model.

In summary, this work has the following contributions:

- We propose Point Ladder Tuning (PLT), a dual-pathway collaborative PEFT framework that adapts a frozen pretrained backbone through a construct-fuse-feedback loop.
- We design a lightweight Hierarchical Ladder Network (HLN) and a Local-Global Fusion (LGF) module to construct multi-resolution local geometric features, selectively integrate them with intermediate pretrained semantics, and feed the fused cues back into the backbone for more effective adaptation.
- We conduct extensive experiments on multiple point-cloud classification and segmentation benchmarks, demonstrating that PLT achieves competitive performance with only a small number of trainable parameters.

2 Related Work

2.1 Self-Supervised Learning on Point Cloud

Recent self-supervised pre-training has advanced point-cloud understanding by leveraging large unlabeled 3D data before task-specific fine-tuning. Current approaches mainly fall into three categories: contrastive learning, reconstruction-based learning, and hybrid methods combining both. In contrastive learning, models like PointContrast [38] and CrossPoint [1] learn view-invariant semantic features by contrasting augmented views of the same cloud. Reconstruction-based methods, such as PointBERT [41] and PointMAE [24], draw inspiration from NLP and vision models by employing masked prediction and autoencoding frameworks. PointGPT [7] scales this paradigm into a larger-scale generative foundation model, demonstrating strong generalization to diverse downstream tasks. To address the scarcity of labeled 3D data, ACT [14] leverages a cross-modal teacher-student framework that transfers semantic priors from a stronger teacher to the 3D learner.

Traditionally, these 3D pre-trained models are fine-tuned for downstream tasks through full parameter updates. However, full fine-tuning can be inefficient, often resulting in the degradation of valuable knowledge gained during pre-training and increasing the risk of catastrophic forgetting. Therefore, this paper investigates more efficient and effective strategies for transferring 3D pre-trained models to downstream tasks while preserving the benefits of pre-training.

2.2 PEFT for Point Cloud

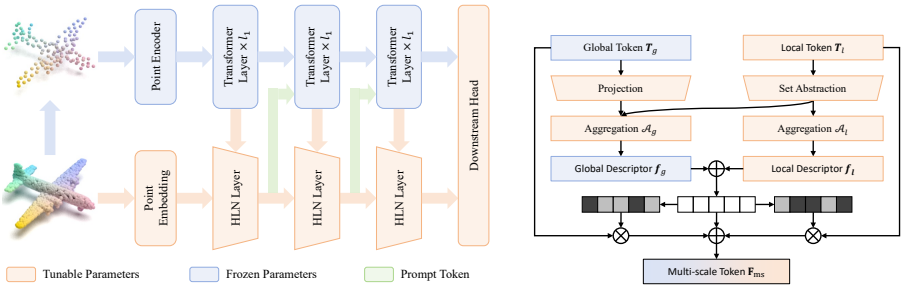
Fine-tuning pre-trained models often demands substantial computational and storage resources. To address these limitations, PEFT techniques have been developed, particularly within natural language processing (NLP) and computer vision. PEFT methods aim to transfer knowledge from pre-trained models to downstream tasks with minimal parameter updates.

In recent years, PEFT methods tailored for 3D point cloud understanding have shown potential in balancing performance and efficiency. IDPT [42] pioneers PEFT for point clouds by employing DGCNN [35] to generate instance-level prompts, effectively adapting conventional prompt-based tuning paradigms to geometric data. Point-PEFT [31] applies a domain-specific memory bank and geometry-aware adapters to encode local geometric priors, while DAPT [46] utilizes dynamic adapters that scale tokens according to task-specific importance and integrate internal prompts for instance-adaptive feature modulation. PPT [45] injects trainable positional prompt tokens at sampled centers and refines them through lightweight inter-layer adapters to enhance 3D spatial awareness. PMA [43] leverages Mamba-based sequence modeling to fuse complementary semantics across modalities, facilitating more comprehensive point cloud reasoning. PointLoRA [34] combines low-rank adaptation (LoRA) with multi-scale token selection, achieving efficient fine-tuning with minimal trainable parameters. Despite these advances, most PEFT methods freeze the backbone network, which restricts fine-grained local feature learning and hinders local-global context integration capabilities crucial for dense prediction tasks [3]. To address this, PointGST [22] incorporates frequency-domain spectral bases derived from raw point clouds into intermediate layers, enriching multi-scale global and local context. STAG [15], GEM [30], and GAPrompt [2] are closer geometry-aware methods. STAG [15] adapts coarsened tokens through a side network, GEM [30] refines token-level positional/context cues with Spatial/Context Adapters, and GAPrompt [2] learns geometry-aware point prompts.

Most existing methods extract single-resolution features from the heavily downsampled point tokens of the backbone (e.g., from 2048 to 128), which improves efficiency but discards fine-grained local structures vital for point cloud understanding. In contrast, our proposed PLT introduces an HLN to extract multi-resolution spatial features directly from the original point clouds and an LGF module to adaptively fuse them with global features from the frozen backbone. This design preserves fine-grained local details while retaining rich global priors, enabling more comprehensive point cloud understanding with fewer trainable parameters and superior performance.

3 Methodology

As shown in Fig. 1, PLT follows a *construct-fuse-feedback* principle for parameter efficient adaptation with a frozen point-cloud backbone. Specifically, we decouple fine-tuning into two tightly coupled pathways: (1) a lightweight **Hierarchical Ladder Network (HLN)** that constructs a multi-resolution pyramid of local



(a) Overall Architecture **(b)** Local-Global Fusion Module (LGF)

Fig. 1: Overview of the proposed Point Ladder Tuning (PLT) framework. (a) The overall architecture couples a frozen pre-trained backbone with tunable Hierarchical Ladder Network (HLN) layers for adaptive multi-resolution and multi-scale learning. (b) The Local-Global Fusion (LGF) module encodes local tokens via set abstraction (SA) and fuses them with global tokens from the backbone to generate multi-scale representations. This hierarchical design facilitates efficient parameter adaptation, robust representation learning, and scalable performance on downstream point cloud tasks.

geometric features directly from raw points (§3.1); and (2) a **prompt-based global adaptation** pathway that injects the constructed locality back into the frozen Transformer via instance-aware, multi-scale prompts (§3.3). These pathways form a bidirectional closed loop through a **Local-Global Fusion (LGF)** module: LGF selectively fuses HLN features with intermediate pretrained semantics, and the resulting multi-scale cues are fed back to modulate self-attention within the backbone (§3.2). In this way, PLT allows local geometry to influence global reasoning inside the frozen backbone, rather than being exploited only through late fusion at the task head.

Finally, we employ lightweight task-specific heads (§3.4) and leverage the hierarchical structure for progressive upsampling to recover spatial resolution. This design is particularly beneficial for dense prediction, where preserving spatial fidelity is critical for accurate outputs.

3.1 Hierarchical Ladder Network

While transformer-based point cloud encoders capture global semantics well, they often miss fine-grained spatial details due to weak local geometric biases. Existing PEFT methods (e.g., PointPEFT [31], PointGST [22]) try to improve locality but still depend heavily on frozen backbone features and fixed resolutions, missing key local details needed for dense prediction.

To solve this, we propose a Hierarchical Ladder Network (HLN) that extracts multi-resolution local features from point clouds. HLN better preserves 3D geometric structure, boosting performance in tasks like segmentation. It has two main components: (1) a Set Abstraction (SA) module and (2) a Local-Global Fusion (LGF) module.

Given an input point cloud $\mathbf{P} \in \mathbb{R}^{N\times 3}$, where N is the number of points, we first apply point embedding to map $\mathbf{P}$ into a high-dimensional space, resulting in a initial point cloud features $\mathbf{F} \in \mathbb{R}^{N\times C}$, where C represents the feature dimension. We then use Set Abstraction (SA) for downsampling, starting with farthest point sampling (FPS) to select a set of center points $\mathbf{C} = \{c_1, \ldots, c_n\}$, followed by k-Nearest Neighbors (kNN) to define local neighborhoods:

$$\mathcal{N}(c_i) = \left\{ (\mathbf{p}_{c_i}^j, \mathbf{f}_{c_i}^j) \mid j = 1, \ldots, k \right\}. \tag{1}$$

The local features are aggregated as:

$$\hat{\mathbf{f}}_{c_i}^j = \varphi\left(\left[\mathbf{f}_{c_i}^j ; (\mathbf{p}_{c_i} - \mathbf{p}_{c_i}^j) \right] \right), \tag{2}$$

$$\mathbf{f}'_{c_i} = \mathbf{f}_{c_i} + \rho\left(h_{\boldsymbol{\Theta}}([\hat{\mathbf{f}}_{c_i}^1; \ldots; \hat{\mathbf{f}}_{c_i}^k]) \right), \tag{3}$$

where φ, ρ are learnable functions, $h_{\boldsymbol{\Theta}}$ denotes the aggregation function (e.g., max pooling) and $[\cdot;\cdot]$ denotes concatenation. By stacking multiple SA layers, HLN constructs a hierarchical local representation $\mathbf{F}_l$ that complements global transformer outputs.

3.2 Local-Global Fusion Module

A key challenge in point cloud learning is to effectively integrate fine-grained geometric details with high-level semantic context. Existing fusion approaches, such as naive concatenation or element-wise addition, typically fail to adaptively balance these two information sources.

To address this issue, we propose a novel Local-Global Fusion (LGF) module that performs adaptive feature integration via selective attention. Our design is motivated by two key insights: (1) local geometric features extracted by the HLN pathway are critical for precise spatial reasoning; and (2) global features from the frozen pre-trained backbone encode rich semantic priors. LGF allows the network to dynamically balance and selectively integrate these complementary representations in a content-aware manner, ensuring that both local details and global semantics are effectively leveraged.

Let $\mathbf{T}_l$ and $\mathbf{T}_g$ denote the local and global tokens extracted from the HLN and the frozen backbone, respectively. To align their feature dimensions, the backbone features are first linearly projected using a learnable transformation matrix W. We then apply an aggregation operation, similar to SA but without spatial downsampling, to capture cross-level interactions between hierarchical feature representations. This process yields global features $\mathbf{F}_g$, which encode the interplay between $\mathbf{T}_l$ and $\mathbf{T}_g$, while the local branch $\mathbf{F}_l$ retains fine-grained intra-local dependencies within $\mathbf{T}_l$.

To effectively integrate these complementary cues, we introduce a selective attention fusion mechanism that adaptively emphasizes the most informative features from each branch. Specifically, a global aggregation function $\mathcal{A}$ (implemented as average pooling) is used to obtain compact descriptors:

$$\mathbf{f}_l = \mathcal{A}_l(\mathbf{F}_l), \quad \mathbf{f}_g = \mathcal{A}_g(\mathbf{F}_g). \tag{4}$$

The resulting descriptors are passed through lightweight MLPs to produce attention logits $\mathbf{z}_l$ and $\mathbf{z}_g$, from which normalized attention weights are derived via a softmax-like operation. These weights modulate the contribution of each feature source, yielding a fused multi-scale representation obtained through a weighted summation. Finally, the fused features are refined by an MLP and combined with the residual shortcut to form the final output $\mathbf{F}_o$, preserving both global contextual consistency and local structural fidelity.

3.3 Dynamic Prompt-Based Global Adaption

To enable the pre-trained backbone network to produce task-adaptive global representations, we propose a dynamic multi-scale prompt generation strategy driven by the fused feature representation $\mathbf{F}_o$ from the LGF module. This design aims to inject instance-specific geometric cues into the transformer backbone, allowing the model to preserve general semantic priors while adapting flexibly to downstream tasks.

Firstly, $\mathbf{F}_o$ is mean-pooled to obtain a compact representation. This pooled feature is then scaled and shifted using learnable parameters γ and β, generating a multi-scale prompt $\mathbf{p}$. To maximize parameter efficiency, we reuse W from the LGF module to align the dimensionality of the multi-scale prompts with that of the backbone network features. The formula for this process is as follows:

$$\mathbf{p} = \left(\gamma \cdot \frac{1}{n} \sum_{i=1}^{n} \mathbf{F}_o^i + \beta \right) W^T, \tag{5}$$

where n is the number of tokens in $\mathbf{F}_o$.

Finally, the multi-scale prompts $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_s]$ are incorporated into the backbone network during training, where s is corresponds to the number of stages in the HLN pathway. These prompts are inserted at each transformer layer of the frozen backbone to condition the global representation in a scale-aware, instance-sensitive manner:

$$\mathbf{x}_l = \mathcal{L}_l \left([\mathbf{T}_{cls}; \mathbf{P}; \mathbf{T}_g] \right), \tag{6}$$

where $\mathcal{L}_l$ is the l-th transformer layer.

The injected prompts participate in the attention mechanism, interacting with the original tokens to enrich the contextual modeling without altering the backbone's architecture. This allows the model to adaptively refine global features based on the special task, while maintaining the semantic richness of large-scale pre-training.

To further enhance adaptability, we adopt the lightweight scale-and-shift tuning strategy proposed in SSF [21], applying element-wise affine transformations to the output of each backbone module: $\mathbf{y} = s \cdot \mathbf{x} + t$, where s and t are learnable scaling and shifting vectors that fine-tune the global representation with minimal overhead.

3.4 Lightweight Segmentation Head

Unlike classification tasks, semantic segmentation requires per-point predictions, necessitating the recovery of features for all original points. A common approach is to propagate features from downsampled representations back to the original point cloud through feature interpolation. However, most existing parameter-efficient fine-tuning methods [22,42,46] for point cloud lack the ability to capture multi-resolution information, leading to a considerable loss of local detail.

To address this issue, we introduce an additional HLN branch to extract fine-grained multi-resolution spatial features. Inspired by PointNext [27], we adopt an inverse distance-weighted interpolation strategy to progressively recover point-level features at each resolution. Unlike PointNext [27], our method concatenates global backbone features with HLN's local multi-resolution features during each interpolation step, forming a more comprehensive representation for effective upsampling.

$$\mathbf{T}_{\text{interp}} = [\text{Interp}(\mathbf{T}, \mathbf{T}_g); \text{Interp}(\mathbf{T}, \mathbf{T}_l)], \tag{7}$$

where $\mathbf{T}$ denotes the point features at the current layer. The function $\text{Interp}(\cdot)$ performs inverse distance interpolation [27].

At each resolution, we concatenate the interpolated features $\mathbf{T}_{\text{interp}}$ with the corresponding original features $\mathbf{T}$, and then pass them through a MLP to obtain updated features. This process is repeated iteratively until the resolution of the original point cloud is restored. Finally, point-wise classification is performed using a MLP followed by a softmax activation to predict the semantic category of each point. This design not only enhances segmentation performance but also substantially reduces the number of trainable parameters and overall computational complexity.

4 Experiments

In this section, we comprehensively evaluate PLT on diverse point cloud datasets [3, 12, 32, 37, 40] and tasks, including 3D object classification, few-shot learning and 3D dense prediction. Our experiments systematically assess PLT's state-of-the-art performance and parameter efficiency, supported by extensive comparisons with existing fine-tuning approaches and an ablation study validating each key component.

For fair comparison with prior methods [42, 46], we adopt identical training configurations. All experiments are conducted on an NVIDIA RTX 3090 GPU, freezing the pre-trained backbone and fine-tuning only the lightweight PLT modules.

To ensure a comprehensive comparison, we employ three state-of-the-art pre-trained models as baselines: Point-BERT [41], Point-MAE [24], ACT [14] and PointGPT [7]. These models represent diverse pre-training paradigms, shown with detailed discussions in the experiment section.

Table 2: Classification results on three variants of ScanObjectNN [32] and ModelNet40 [37], including the number of trainable parameters and overall accuracy (OA). All methods apply default data augmentation as used in [46]. * indicates reproduced results. † indicates training with a post–pre-training checkpoint and the use of generated point clouds as additional supervision for the reconstruction loss. For ScanObjectNN [32], results are reported without voting. For ModelNet40 [37], results are shown w/o and w/ voting as (-/-). TP represents the tunable parameters.

Method	Reference	TP (M)	FLOPs (G)	ScanObjectNN			ModelNet40
				OBJ_BG	OBJ_ONLY	PB_T50_RS	
Point-BERT [41]	CVPR 22	22.1 (100%)	4.9	87.43	88.12	83.07	92.7 / 93.2
+ IDPT [42]	ICCV 23	1.7 (7.69%)	7.3	88.12(+0.69)	88.30(+0.18)	83.69(+0.62)	92.6(-0.1) / 93.4(+0.2)
+ DAPT [46]	CVPR 24	1.1 (4.97%)	5.1	91.05(+3.62)	89.67(+1.55)	85.43(+2.36)	93.1(+0.4) / 93.6(+0.4)
+ PMA [43]	CVPR 25	1.1 (4.97%)	-	91.39(+3.96)	**91.05**(+2.93)	85.50(+2.43)	**93.7**(+1.0) / -
+ PointGST [22]	TPAMI 25	0.62 (2.81%)	5.0	91.39(+3.96)	89.67(+1.55)	85.64(+2.57)	93.4(+0.7) / 93.8(+0.6)
+ LST [29]	NeurIPS 22	0.8 (3.38%)	5.0	89.15(+2.72)	89.50(+1.38)	83.17(+0.10)	92.9(+0.2) / 93.3(+0.1)
+ PLT (ours)	-	**0.60 (2.71%)**	5.0	**91.57**(+4.14)	89.85(+1.73)	**86.09**(+3.02)	93.5(+0.8) / **94.2**(+1.0)
Point-MAE [24]	ECCV 22	22.1 (100%)	4.9	90.02	88.29	85.18	93.2 / 93.8
+ IDPT [42]	ICCV 23	1.7 (7.69%)	7.3	91.22(+1.20)	90.02(+1.73)	84.94(-0.24)	93.3(+0.1) / 94.4(+0.6)
+ Point-PEFT* [31]	AAAI 24	0.7 (3.13%)	-	90.19(+0.17)	89.50(+1.21)	84.35(-0.83)	**94.2**(+1.0) / -
+ DAPT [46]	CVPR 24	1.1 (4.97%)	5.1	90.88(+0.86)	90.19(+1.90)	85.08(-0.10)	93.5(+0.3) / 94.0(+0.2)
+ PPT* [45]	ACM MM 25	1.1 (4.97%)	-	89.50(-0.52)	89.50(+1.21)	84.91(-0.27)	93.7(+0.5) / -
+ PMA [43]	CVPR 25	1.1 (4.97%)	-	91.05(+1.03)	**90.89**(+2.60)	**86.43**(+1.25)	94.0(+0.2) / -
+ PointGST [22]	TPAMI 25	0.62 (2.81%)	5.0	**91.74**(+1.72)	90.19(+1.90)	85.29(+0.11)	93.5(+0.3) / 94.0(+0.2)
+ PointLoRA [34]	CVPR 25	0.77 (3.43%)	-	90.71(+0.69)	89.33(+1.04)	85.53(+0.35)	93.3(+0.1) / -
+ LST [29]	NeurIPS 22	0.8 (3.38%)	5.0	89.67(-0.35)	89.67(+1.38)	82.75(-2.43)	93.2(+0.0) / 93.8(+0.0)
+ PLT (ours)	-	**0.60 (2.71%)**	5.0	90.88(+0.86)	90.02(+1.73)	85.53(+0.35)	93.8(+0.6) / 94.0(+0.2)
ACT [14]	ICLR 23	22.1 (100%)	4.9	91.22	89.16	85.81	93.0 / 93.7
+ IDPT [42]	ICCV 23	1.7 (7.69%)	7.3	89.50(-1.72)	89.33(+0.17)	84.42(-1.39)	92.9(-0.1) / 93.6(-0.1)
+ Point-PEFT [31]	AAAI 24	0.7 (3.13%)	-	89.33(-1.89)	**90.88**(+1.72)	84.80(-1.01)	**94.0**(+1.0) / -
+ DAPT [46]	CVPR 24	1.1 (4.97%)	5.1	89.33(-1.89)	88.30(-0.86)	83.80(-2.01)	92.8(-0.2) / 93.4(-0.3)
+ PPT [45]	ACM MM 25	1.1 (4.97%)	-	88.98(-2.24)	88.98(+0.18)	85.05(-0.76)	92.9(-0.1) / 93.8(+0.1)
+ PointGST [22]	TPAMI 25	0.62 (2.81%)	5.0	88.81(-2.41)	89.33(+0.17)	84.28(-1.53)	93.2(+0.2) / 93.5(-0.2)
+ LST [29]	NeurIPS 22	0.8 (3.38%)	5.0	90.02(-1.20)	89.50(+0.34)	82.41(-3.40)	93.4(+0.4) / 93.6(-0.1)
+ PLT (ours)	-	**0.60 (2.71%)**	5.0	**90.71**(-0.51)	90.71(+1.55)	**85.39**(-0.42)	93.6(+0.6) / **94.0**(+0.3)
PointGPT-L† [7]	NeurIPS 23	360.5 (100%)	67.7	97.2	96.6	93.4	94.1 / 94.7
+ IDPT [42]	ICCV 23	10.0 (2.77%)	75.2	98.11(+0.91)	96.04(-0.56)	92.99(-0.41)	93.4(-0.7) / 94.6(-0.1)
+ DAPT [46]	CVPR 24	4.2 (1.17%)	71.6	98.11(+0.91)	96.21(-0.39)	93.02(-0.38)	94.2(+0.1) / 94.9(+0.2)
+ PMA [43]	CVPR 25	4.9 (1.36%)	-	98.97(+1.77)	96.73(+0.13)	95.18(+1.78)	**94.9**(+0.8) / **95.4**(+0.7)
+ PointGST [22]	TPAMI 25	2.4 (0.67%)	68.1	98.97(+1.77)	**97.59**(+0.99)	94.83(+1.43)	94.8(+0.7) / 95.3(+0.6)
+ LST [29]	NeurIPS 22	1.7 (0.47%)	67.9	97.76(+0.56)	95.53(-1.07)	92.75(-0.65)	93.4(-0.7) / 94.5(-0.2)
+ PLT (ours)	-	**1.3 (0.36%)**	68.2	**99.14**(+1.94)	97.25(+0.65)	**95.21**(+1.81)	94.5(+0.4) / 95.0(+0.3)

4.1 3D Object Classification

Object Classification on Synthetic Dataset. We conduct experiments on ModelNet40 [37] under the same settings as DAPT [46] to ensure a fair and consistent comparison.

As shown in Tab. 2, without voting, PLT achieves 93.8%, 93.5%, and 93.6% on Point-MAE [24], Point-BERT [41], and ACT [14], surpassing full fine-tuning by +0.6%, +0.8%, +0.6%. With voting, PLT further improves Point-BERT by +1.0%. These results confirm PLT's effectiveness in point-cloud classification by capturing local structures and leveraging both global and local representations.

Object Classification on Real-World Dataset. Most pre-trained point cloud models rely on synthetic data like ShapeNet [6], while real-world scans introduce noise, missing points, and background clutter. We evaluate PLT on ScanObjectNN [32] (OBJ_BG, OBJ_ONLY, PB_T50_RS) using Point-BERT [41], Point-MAE [24], and ACT [14] as baselines to assess robustness in challenging real-world conditions.

Our PLT achieves significant accuracy gains over full fine-tuning with only 2.71% of parameters, improving Point-BERT [41] by +4.14%, +1.73% and +3.02%

(a) Full **(b)** LP **(c)** IDPT [42] **(d)** DAPT [46] **(e)** PLT (Ours)

Fig. 2: The visualization of the t-SNE [23] from the test sets of ScanObjectNN [32] (PB_T50_RS) by using a pre-trained PointMAE [24] with various fine-tuning strategies. We extract the final classification features from the top linear layer for t-SNE visualizations. LP represents Linear Probing.

across ScanObjectNN [32] variants. Under the most challenging PB_T50_RS setting, PLT further surpasses PointGST by +0.45% on Point-BERT [41] and +0.24% on PointMAE [24], showing strong robustness and efficiency to noisy, occluded data. Furthermore, it also exhibits the least performance drop on ACT [14], maintaining stable accuracy where others significantly decline, underscoring its architectural strength in preserving semantic representations during adaptation.

As shown in Fig. 2, we further visualize the feature manifolds of full fine-tuning, point-cloud-specific PEFT methods and our PLT on the ScanObjectNN PB_T50_RS dataset using t-SNE [23]. Compared with previous methods, Our PLT exhibits significantly greater inter-cluster dispersion and smaller intra-cluster distances, highlighting its superior ability to preserve discriminative features. These results collectively demonstrate that PLT attains an effective balance between representation quality and parameter efficiency, making it well suited for point-cloud learning.

4.2 3D Dense Prediction Task

For dense prediction tasks, including part and semantic segmentation, we employ a lightweight prediction head to fully leverage multi-resolution features while keeping parameters minimal.

As shown in Tab. 4, PLT consistently achieves superior performance across all datasets and backbones with far fewer tunable parameters. On ShapeNet-Part [40], PLT attains competitive instance-level mIoU (Inst. mIoU) and improves class-level mIoU (Cls. mIoU). Specifically, PLT improves Inst. mIoU on PointBERT [41] by 0.5% over DAPT [46], demonstrating its effectiveness in capturing fine-grained geometric details.

As shown in Tab. 3, on the large-scale indoor scene benchmarks S3DIS [3] and ScanNetV2 [12], PLT achieves state-of-the-art results among PEFT methods with only about 2M tunable parameters, far fewer than the 27 M required for full fine-tuning. With ACT as the backbone on S3DIS [3], PLT attains 70.6% mAcc and 61.5% mIoU, surpassing the closest competitor by 2.3% in mIoU. On the more challenging ScanNetV2, PLT leads in both voxel- and point-level mIoU

Table 3: Semantic segmentation on the S3DIS [3] and ScanNetV2 [12]. The mean accuracy (mAcc) and mean IoU (mIoU) are reported on the S3DIS. VmIoU and PmIoU represents Voxel mIoU and Point mIoU respectively. TP represents the tunable parameters.

Methods	Reference	TP (M)	S3DIS		ScanNetV2	
			mAcc	mIoU	VmIoU	PmIoU
Point-BERT [41]	CVPR 22	27.02 (100%)	69.7	62.1	49.9	49.6
+ IDPT [42]	ICCV 23	5.64 (20.9%)	66.9	57.7	33.9	33.6
+ DAPT [46]	CVPR 24	5.61 (20.8%)	68.3	58.9	43.6	43.3
+ PPT [45]	ACM MM 25	5.58 (20.7%)	68.8	60.5	46.7	46.3
+ PointGST [22]	TPAMI 25	5.55 (20.5%)	68.5	59.5	46.3	45.8
+ LST [29]	NeurIPS 22	5.65 (20.9%)	67.4	58.6	34.5	34.2
+ PLT (**ours**)	-	**2.04 (7.55%)**	**69.6**	**61.1**	**48.2**	**47.8**
Point-MAE [24]	ECCV 22	27.02 (100%)	69.9	60.8	51.2	50.8
+ IDPT [42]	ICCV 23	5.64 (20.9%)	66.6	57.6	33.5	33.2
+ Point-PEFT [31]	AAAI 24	5.58 (20.7%)	66.5	56.0	-	-
+ DAPT [46]	CVPR 24	5.61 (20.8%)	68.2	59.3	41.1	40.7
+ PPT [45]	ACM MM 25	5.58 (20.7%)	68.7	60.4	46.2	45.8
+ PointGST [22]	TPAMI 25	5.55 (20.5%)	68.4	58.6	44.6	44.2
+ LST [29]	NeurIPS 22	5.65 (20.9%)	66.9	57.7	35.2	34.9
+ PLT (**ours**)	-	**2.04 (7.55%)**	**70.5**	**61.5**	**47.2**	**46.8**
ACT [14]	ICLR 23	27.02 (100%)	71.1	61.2	50.9	50.5
+ IDPT [42]	ICCV 23	5.64 (20.9%)	64.9	55.2	32.1	31.9
+ Point-PEFT [31]	AAAI 24	5.58 (20.7%)	66.0	54.6	-	-
+ DAPT [46]	CVPR 24	5.61 (20.8%)	67.9	58.5	40.3	40.0
+ PPT [45]	ACM MM 25	5.58 (20.7%)	69.1	59.2	45.5	45.2
+ PointGST [22]	TPAMI 25	5.55 (20.5%)	67.6	57.4	44.2	43.8
+ LST [29]	NeurIPS 22	5.65 (20.9%)	67.6	58.3	34.0	33.7
+ PLT (**ours**)	-	**2.04 (7.55%)**	**70.6**	**61.5**	**46.9**	**46.6**

(48.2%/47.8%) and shows notable gains over DAPT [46] and PointGST [22] for PointBERT [41].

These results highlight PLT's strong generalization and efficiency in dense 3D scene prediction. Its HLN effectively captures multi-resolution semantic cues from raw points, while the LGF module and dynamic prompts harmonize local details with global priors. A lightweight segmentation head further strengthens dense-prediction accuracy with minimal computational overhead. This synergy enhances fine-grained localization and semantic consistency across varying spatial scales, making PLT a lightweight yet powerful PEFT solution for 3D dense prediction.

4.3 Model Scaling

To assess the scalability of the proposed PLT framework, we conduct experiments across point cloud backbones of different model sizes, ranging from regular-scale networks to large-scale foundation models.

As reported in Tab. 2 and Tab. 4, PLT consistently boosts performance on both classification (ScanObjectNN [32], ModelNet40 [37]) and segmentation (ShapeNetPart [40]) benchmarks, regardless of backbone size. In particular, PLT achieves competitive or superior accuracy with only 0.6M tunable parameters

Table 4: Part segmentation on the ShapeNetPart [40]. The mIoU for all classes (Cls.) and for all instances (Inst.) are reported. TP represents the tunable parameters. * denotes reproduced results, while † indicates training initialized from a post-pre-training checkpoint.

Methods	Reference	TP (M)	Cls. mIoU (%)	Inst. mIoU (%)
Point-BERT [41]	CVPR 22	27.06 (100%)	84.11	85.6
+ IDPT* [42]	ICCV 23	5.69 (21.0%)	83.50	85.3
+ DAPT [46]	CVPR 24	5.65 (20.9%)	83.83	85.5
+ PMA [43]	CVPR 25	5.64 (20.8%)	**83.96**	**86.1**
+ PointGST [22]	TPAMI 25	5.58 (20.6%)	83.87	85.7
+ LST [29]	NeurIPS 22	5.69 (21.0%)	83.38	85.2
+ PLT (**ours**)	-	**2.08** (**7.69%**)	83.85	86.0
Point-MAE [24]	ECCV 22	27.06 (100%)	84.19	86.1
+ IDPT [42]	ICCV 23	5.69 (21.0%)	83.79	85.7
+ DAPT [46]	CVPR 24	5.65 (20.9%)	84.01	85.7
+ PPT [45]	ACM MM 25	5.62 (20.6%)	**84.07**	85.7
+ PMA [43]	CVPR 25	5.64 (20.8%)	84.01	**86.1**
+ PointGST [22]	TPAMI 25	5.59 (21.0%)	83.98	85.8
+ LST [29]	NeurIPS 22	5.69 (21.0%)	83.78	85.5
+ PLT (**ours**)	-	**2.08** (**7.69%**)	83.90	85.9
ACT [14]	ICLR 23	27.06 (100%)	84.66	86.1
+ IDPT [42]	ICCV 23	5.69 (21.0%)	83.80	85.7
+ DAPT [46]	CVPR 24	5.65 (20.9%)	83.56	85.7
+ PPT [45]	ACM MM 25	5.62 (20.6%)	83.83	85.7
+ PointGST [22]	TPAMI 25	5.59 (21.0%)	83.80	85.6
+ LST [29]	NeurIPS 22	5.69 (21.0%)	83.70	85.6
+ PLT (**ours**)	-	**2.08** (**7.69%**)	**83.88**	**86.0**
PointGPT-L† [7]	NeurIPS 23	339.4 (100%)	84.8	86.6
PointGPT-L†* [7]	NeurIPS 23	339.4 (100%)	84.16	86.1
+ IDPT [42]	ICCV 23	33.17 (9.77%)	83.14	85.3
+ DAPT [46]	CVPR 24	33.61 (9.90%)	83.17	85.3
+ PointGST [22]	TPAMI 25	31.84 (9.38%)	83.69	85.8
+ LST [29]	NeurIPS 22	32.06 (9.45%)	82.75	85.2
+ PLT (**ours**)	-	**3.85** (**1.13%**)	**84.07**	**86.2**

when applied to Point-BERT [41], Point-MAE [24], and ACT [14], demonstrating its efficiency in small-scale settings. When scaled to large models such as PointGPT-L, PLT continues to deliver notable improvements (e.g., +1.94% OA on ScanObjectNN OBJ_BG and +1.81% mIoU on ShapeNetPart), while requiring just 1.3M and 3.85M trainable parameters for classification and segmentation, corresponding to only 0.36% and 1.13% of the full model size.

These results highlight the strong scalability of PLT: it not only adapts effectively to compact models with minimal overhead, but also remains effective when applied to large-scale foundation models, thereby offering a unified and parameter-efficient tuning paradigm for diverse point cloud architectures.

4.4 Few-shot Learning

We further evaluate PLT's transferability on ModelNet40 [37] dataset for assessing the efficiency of data usage in low-resource settings. Following prior works [42, 46], we adopt the standard n-way m-shot protocol, where $n \in \{5, 10\}$ and $m \in \{10, 20\}$.

As shown in Tab. 5, PLT consistently surpasses full fine-tuning and leading PEFT methods across most settings and backbones (Point-BERT [41], ACT [14]). In the 5-way 20-shot setting with Point-BERT [41], PLT reaches 98.8% accuracy, exceeding full fine-tuning by +2.5% and PointGST [22] by +0.9%. These results demonstrate PLT's strong generalization in data-scarce scenarios, attributed to its hierarchical local feature extraction, which introduces a robust inductive bias and enhances fine-tuning effectiveness.

In addition to superior accuracy, PLT shows improved stability across few-shot settings, yielding consistently lower or comparable standard deviations than other PEFT baselines. For example, in the 5-way 20-shot setting with Point-BERT [41], PLT achieves ±1.1%, outperforming LST [29] (±1.8%) and Point-GST [22] (±2.0%). This consistent trend across configurations highlights PLT's ability to reduce performance variance, which is crucial for robust few-shot learning under limited data.

Table 5: Few-shot learning on ModelNet40 [37], Overall accuracy (%)± standard deviation (%) w/o voting is reported.

Methods	Reference	5-way		10-way	
		10-shot	20-shot	10-shot	20-shot
Point-BERT [41] (baseline)	CVPR 22	94.6±3.1	96.3±2.7	91.0±5.4	92.7±5.1
+ IDPT [42]	ICCV 23	96.0±**1.7**	97.2±2.6	91.9±4.4	93.6±3.5
+ DAPT [46]	CVPR 24	95.8±2.1	97.3±1.3	92.2±4.3	94.2±3.4
+ PointGST [22]	TPAMI 25	96.5±2.4	97.9±2.0	92.7±4.2	95.0±**2.8**
+ LST [29]	NIPS 22	94.3±2.6	97.1±1.8	90.6±4.7	93.7±3.7
+ PLT (**ours**)	-	**96.9**±2.0	**98.8±1.1**	**93.3±4.0**	**95.5**±3.1
ACT [14] (baseline)	ICLR 23	96.8±2.3	98.0±1.4	93.3±4.0	95.6±2.8
+ IDPT [42]	ICCV 23	96.8±1.9	98.3±1.2	92.5±4.1	95.3±3.3
+ DAPT [46]	CVPR 24	95.3±2.8	97.1±1.7	89.8±4.8	94.1±3.6
+ PPT [45]	ACM MM 25	**97.1**±2.3	98.1±1.8	91.8±4.3	94.9±3.4
+ PointGST [22]	TPAMI 25	97.2±1.9	98.3±1.3	92.9±4.2	95.7±**2.6**
+ LST [29]	NIPS 22	96.2±2.6	98.0±1.9	92.6±4.4	94.9±3.4
+ PLT (**ours**)	-	96.9±**1.8**	**98.9±1.0**	**93.4±4.0**	**95.9**±3.1

4.5 Ablation Study

Comparison Between HLN and PLT. Fig. 3 shows that **HLN** alone, while recovering multi-resolution locality from raw points, remains limited (**80.74%**) due to the lack of semantic priors from the pre-trained backbone. Coupling HLN with the backbone via **LGF** raises accuracy to **83.34%** (+2.60%), validating the necessity of *selective* semantic injection. Adding **Dynamic Prompt** further improves to **84.52%** (+1.18 %) with a comparable tunable budget, highlighting the role of instance-conditioned high-level modulation. Finally, **SSF** delivers the full **PLT** performance of **85.53%** (+1.01), while removing only DP drops to **84.66%**, evidencing clear complementarity between prompt adaptation and layer-wise refinement. Overall, PLT improves over HLN by +**4.79** % and even

surpasses full fine-tuning (85.18%) with $< 3\%$ trainable parameters, underscoring PLT as a co-designed locality-aware PEFT framework for point cloud understanding.

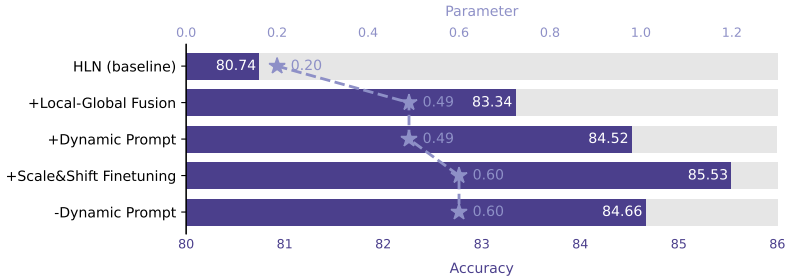


Fig. 3: Ablation study demonstrating the necessity and synergy of each PLT component.

Fusion Strategy for Local and Global Information. The integration strategy of local and global information plays a critical role in downstream performance. In Tab. 6, we evaluate several fusion mechanisms. The proposed Local-Global Fusion (LGF) with Softmax achieves the highest accuracy (85.53%), outperforming additive, concatenative and single-source approaches. Notably, leveraging only local features yields a considerably lower accuracy (82.86%) than relying solely on global features (84.52%), reflecting the benefits of pretrained global representation learning.

Table 6: Fusion way for local and global information in point clouds on PointMAE [24]. The tunable parameters (TP) and the overall accuracy (%) on the hardest variant of ScanObjectNN [32] are reported.

Fusion Way	TP (M)	PB_T50_RS
Add	0.58	85.01
Concat	0.62	84.63
Only Global	0.56	84.52
Only Local	0.56	82.86
LGF with Sigmoid	0.59	84.59
LGF with Softmax	0.60	**85.53**

HLN Source Attribution. Tab. 7 and Tab. 8 isolate the role of the HLN information source in classification and segmentation, respectively. Replacing raw-point HLN with token-center HLN consistently degrades performance, from 85.53% to 84.73% OA in classification and from 60.5% to 59.0% mIoU under the standard-head setting in segmentation. These results show that PLT's gain is not merely due to additional parameters or an auxiliary branch; rather, it stems

from leveraging fine-grained raw-point geometry that complements frozen coarse tokens.

Table 7: Raw-point versus token-center/random-center HLN under the same parameter budget with PointMAE [24] on ScanObjectNN PB_T50_RS [32]. Token-center HLN changes the HLN source from raw points to backbone token centers; Random center uses randomly sampled centers.

Variant	**Ours**	Token-center HLN	Random center
TP(M)	0.60	0.60	0.60
OA(%)	**85.53**	84.73	85.01

Segmentation Head Attribution. As shown in Tab. 8, the improvement in dense prediction is not solely attributable to the lightweight segmentation head. Replacing it with the standard segmentation head increases the number of trainable parameters, yet reduces mIoU to 60.5, while still outperforming most 3D PEFT baselines.

Table 8: Segmentation head and HLN source controls with ACT [14] on S3DIS [3]. Ours + Standard Head replaces the lightweight head with the standard head. Token-center HLN further changes the HLN source from raw points to backbone token centers.

Variant	**Ours**	Ours + Standard Head	Token-center HLN + Standard Head
TP(M)	2.04	5.86	5.86
mIoU(%)	**61.5**	60.5	59.0

5 Conclusion

We proposed Point Ladder Tuning (PLT), a parameter-efficient fine-tuning framework for point-cloud analysis. PLT freezes the pre-trained backbone and employs a Hierarchical Ladder Network (HLN) to extract local cues, integrated with global priors through Local-Global Fusion (LGF). A dynamic prompt generator further guides the frozen backbone toward task-aware adaptation, while a lightweight segmentation head boosts dense-prediction accuracy with minimal overhead. Experiments show PLT achieves strong results in classification and segmentation with minimal added parameters.

Acknowledgements

This work is supported by Mobile Information Networks-National Science and Technology Major Project under Grant No.2025ZD1303200.

References

1. Afham, M., Dissanayake, I., Dissanayake, D., Dharmasiri, A., Thilakarathna, K., Rodrigo, R.: Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. In: CVPR. pp. 9902–9912 (2022)
2. Ai, Z., Liu, Z., Lei, Y., Cui, Z., Zou, X., Zhou, J.: Gaprompt: Geometry-aware point cloud prompt for 3d vision model. In: ICML. pp. 796–808. PMLR (2025)
3. Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S.: 3d semantic parsing of large-scale indoor spaces. In: CVPR. pp. 1534–1543 (2016)
4. Brown, T.B.: Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020)
5. Casado-Coscolla, A., Sanchez-Belenguer, C., Wolfart, E., Sequeira, V.: Rendering massive indoor point clouds in virtual reality. VR **27**(3), 1859–1874 (2023)
6. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
7. Chen, G., Wang, M., Yang, Y., Yu, K., Yuan, L., Yue, Y.: Pointgpt: Auto-regressively generative pre-training from point clouds. NeurIPS **36**, 29667–29679 (2023)
8. Chen, K., Lopez, B.T., Agha-mohammadi, A.a., Mehta, A.: Direct lidar odometry: Fast localization with dense point clouds. IEEE RAL **7**(2), 2000–2007 (2022)
9. Chen, S., Liu, B., Feng, C., Vallespi-Gonzalez, C., Wellington, C.: 3d point cloud processing and learning for autonomous driving. arXiv preprint arXiv:2003.00601 (2020)
10. Chen, X., Fan, H., Girshick, R., He, K.: Improved baselines with momentum contrastive learning. arXiv preprint arXiv:2003.04297 (2020)
11. Christen, S., Yang, W., Pérez-D'Arpino, C., Hilliges, O., Fox, D., Chao, Y.W.: Learning human-to-robot handovers from point clouds. In: CVPR. pp. 9654–9664 (2023)
12. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: CVPR. pp. 5828–5839 (2017)
13. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
14. Dong, R., Qi, Z., Zhang, L., Zhang, J., Sun, J., Ge, Z., Yi, L., Ma, K.: Autoencoders as cross-modal teachers: Can pretrained 2d image transformers help 3d representation learning? In: ICLR (2022)
15. Furuya, T.: Token adaptation via side graph convolution for efficient fine-tuning of 3d point cloud transformers. Machine Vision and Applications **37**(3), 44 (2026)
16. Garrido, D., Rodrigues, R., Augusto Sousa, A., Jacob, J., Castro Silva, D.: Point cloud interaction and manipulation in virtual reality. In: AIVR. pp. 15–20 (2021)
17. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR. pp. 9729–9738 (2020)
18. Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., Gelly, S.: Parameter-efficient transfer learning for nlp. In: ICML. pp. 2790–2799 (2019)
19. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. In: ACL (2021)

20. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. NeurIPS **31** (2018)
21. Lian, D., Zhou, D., Feng, J., Wang, X.: Scaling & shifting your features: A new baseline for efficient model tuning. NeurIPS pp. 109–123 (2022)
22. Liang, D., Feng, T., Zhou, X., Zhang, Y., Zou, Z., Bai, X.: Parameter-efficient fine-tuning in spectral domain for point cloud learning. IEEE TPAMI (2025)
23. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. JMLR **9**(11) (2008)
24. Pang, Y., Wang, W., Tay, F.E., Liu, W., Tian, Y., Yuan, L.: Masked autoencoders for point cloud self-supervised learning. In: ECCV. pp. 604–621 (2022)
25. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR. pp. 652–660 (2017)
26. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. NeurIPS **30** (2017)
27. Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M., Ghanem, B.: Pointnext: Revisiting pointnet++ with improved training and scaling strategies. NeurIPS **35**, 23192–23204 (2022)
28. Song, Z., Yang, L., Xu, S., Liu, L., Xu, D., Jia, C., Jia, F., Wang, L.: Graphbev: Towards robust bev feature alignment for multi-modal 3d object detection. In: ECCV. pp. 347–366 (2024)
29. Sung, Y.L., Cho, J., Bansal, M.: Lst: Ladder side-tuning for parameter and memory efficient transfer learning. NeurIPS **35**, 12991–13005 (2022)
30. Tang, L., Chen, Z., Tao, D.: On geometry-enhanced parameter-efficient fine-tuning for 3d scene segmentation. NeurIPS **38**, 168533–168556 (2026)
31. Tang, Y., Zhang, R., Guo, Z., Ma, X., Zhao, B., Wang, Z., Wang, D., Li, X.: Point-peft: Parameter-efficient fine-tuning for 3d pre-trained models. In: AAAI. vol. 38, pp. 5171–5179 (2024)
32. Uy, M.A., Pham, Q.H., Hua, B.S., Nguyen, T., Yeung, S.K.: Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: ICCV. pp. 1588–1597 (2019)
33. Wang, G., Li, W., Jiang, C., Zhu, D., Li, Z., Xu, W., Zhao, H., Ding, H.: Trajectory planning and optimization for robotic machining based on measured point cloud. IEEE T-RO **38**(3), 1621–1637 (2021)
34. Wang, S., Liu, X., Kong, L., Xu, J., Hu, C., Fang, G., Li, W., Zhu, J., Wang, X.: Pointlora: Low-rank adaptation with token selection for point cloud learning. In: CVPR. pp. 6605–6615 (2025)
35. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM TOG **38**(5), 1–12 (2019)
36. Wu, X., Jiang, L., Wang, P.S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T., Zhao, H.: Point transformer v3: Simpler faster stronger. In: CVPR. pp. 4840–4851 (2024)
37. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: CVPR. pp. 1912–1920 (2015)
38. Xie, S., Gu, J., Guo, D., Qi, C.R., Guibas, L., Litany, O.: Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In: ECCV. pp. 574–591 (2020)
39. Yang, Z., Chen, L., Sun, Y., Li, H.: Visual point cloud forecasting enables scalable autonomous driving. In: CVPR. pp. 14673–14684 (2024)
40. Yi, L., Kim, V.G., Ceylan, D., Shen, I.C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.: A scalable active framework for region annotation in 3d shape collections. ACM TOG **35**(6), 1–12 (2016)

41. Yu, X., Tang, L., Rao, Y., Huang, T., Zhou, J., Lu, J.: Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In: CVPR. pp. 19313–19322 (2022)
42. Zha, Y., Wang, J., Dai, T., Chen, B., Wang, Z., Xia, S.T.: Instance-aware dynamic prompt tuning for pre-trained point cloud models. In: ICCV. pp. 14161–14170 (2023)
43. Zha, Y., Wang, Y., Guo, H., Wang, J., Dai, T., Chen, B., Ouyang, Z., Yuerong, X., Chen, K., Xia, S.T.: Pma: Towards parameter-efficient point cloud understanding via point mamba adapter. In: CVPR. pp. 16976–16986 (2025)
44. Zhang, R., Guo, Z., Gao, P., Fang, R., Zhao, B., Wang, D., Qiao, Y., Li, H.: Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. NeurIPS pp. 27061–27074 (2022)
45. Zhang, S., Qi, Z., Dong, R., Bai, X., Wei, X.: Positional prompt tuning for efficient 3d representation learning. In: ACM MM (2025)
46. Zhou, X., Liang, D., Xu, W., Zhu, X., Xu, Y., Zou, Z., Bai, X.: Dynamic adapter meets prompt tuning: Parameter-efficient transfer learning for point cloud analysis. In: CVPR. pp. 14707–14717 (2024)