PLT: Point Ladder Tuning with Multi-Scale Prompt for Point Cloud Parameter-Efficient Learning

Anonymous CVPR submission

Paper ID 16372

Abstract

Point cloud analysis has achieved outstanding performance by transferring point cloud pre-trained models. However, existing methods for model adaptation usually update all model parameters, i.e., full fine-tuning paradigm, which is inefficient and ineffective. In this paper, we aim to study parameter-efficient transfer learning for point cloud analysis with an ideal tradeoff between task performance and parameter efficiency. To achieve this goal, we freeze the parameters of the default pre-trained models and utilizes a hierarchical Ladder Network (HLN) to directly extract local information from the input point cloud. To aggregate this local information with the backbone network's intermediate global activation, we introduce a Local-Global Fusion (LGF) module. Additionally, to further enhance performance, fused multi-scale features are used to generate dynamic prompts, enabling the backbone network to produce global features optimized for downstream tasks. A series of experiments on tasks such as point cloud object classification and dense prediction have demonstrated the effectiveness of our proposed method. Compared to full fine-tuning, our approach achieves superior performance while requiring significantly fewer parameters, which uses only 2.71% and 7.69% of the parameters required for full fine-tuning in the object classification and the dense prediction respectively. The code will be released when received.

1. Introduction

With the growing accessibility of 3D scanning technology, 3D point cloud learning has emerged as a popular area of research with wide-ranging applications in fields such as autonomous driving, VR/AR, and robotics.In contrast to images, point clouds are unstructured, sparse, and permutation-invariant, presenting significant challenges for analysis and processing.Therefore, deep learning methods[12, 26, 30, 33–35, 38, 48, 50, 51, 59, 63] for point cloud learning have been developed with specialized modules designed to directly process point clouds, resulting in significant performance improvements.

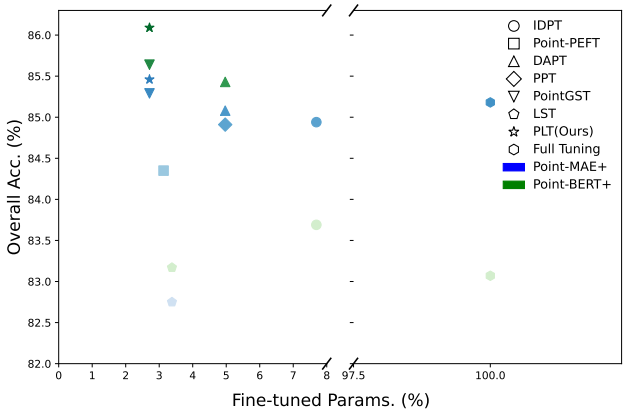

Figure 1. The comparison of several methods on the hardest variant of ScanObjectNN[46]. Different shapes stand for different methods. Blue and green respectively represent the results under the PointMAE[32] and PointBERT[56] baselines. The darker the color, the higher the accuracy (The accuracy is mapped to the range [0,1] by using max-min scaling[31]).

Recently, inspired by the success of pre-trained models in natural language processing[3, 9, 23, 39, 44, 45] and computer vision[6, 7, 14, 15, 54, 55], a series of works[1, 10, 29, 32, 36, 47, 53, 56, 60] have also emerged in the field of point cloud analysis.After pre-training, these methods employ a classic full fine-tuning strategy to adapt to downstream tasks, resulting in significant performance improvements and faster convergence compared to training from scratch.However, fully fine-tuning techniques may be suboptimal for point cloud processing for three reasons: 1) Updating all parameters of the pre-trained model can lead to overfitting and catastrophic forgetting, undermining the rich embedding knowledge acquired during the pre-training phase and resulting in poor performance. 2) Each point cloud analysis task and dataset requires separate copies of model parameters, which can pose storage challenges as demand increases. 3) To fully leverage the prior knowl-

edge from the pre-training dataset, large models are often necessary. Full fine-tuning can incur significant computational costs, including increased GPU memory usage and extended training times, limiting accessibility for researchers with poor hardware resources.

To alleviate these problem, we will shift our research focus to Parameter Efficient Fine-tuning(PEFT)[5, 17–20, 22, 24, 25, 27, 42, 57] in NLP and computer vision, which freezes most parameters of pretrained models, only a few selected parameters or some other ones inserted are trainable during tuning, allowing us to achieve performance comparable to or even better than full fine-tuning. For language and visual models, several pioneering works have emerged: 1) Adapter[5, 17] inserts lightweight networks after the Attention and Feed-Forward Network (FFN) layers in Transformers. 2) Prompt Tuning[19, 24, 25] adds a small number of learnable parameters to the token sequence. 3) Ladder Side Tuning[42] utilizes a small, independent network called Ladder, which takes intermediate activations as input and fine-tunes them through quick connections to the backbone network.These methods have achieved impressive results.

However, we empirically find that directly applying fine-tuning methods developed for language and vision models to the point cloud domain can lead to suboptimal performance. As illustrated in Tab.1, using VPT[19] for fine-tuning in PointMAE[32] resulted in a notable performance decline compared to full fine-tuning across various configurations of the ScanObjectNN[46], particularly with a 4.09% decrease in the PB_T5_RS setting. Similarly, while adapters[17] showed slight improvements on the OBJ_ONLY task, they also resulted in significant performance drops in other settings. Ladder Side Tuning (LST)[42] demonstrated substantial gains on OBJ_ONLY, but performance in other settings remained inadequate. This raises a critical question: **how can we design an efficient and effective fine-tuning method tailored specifically for point clouds that achieves performance comparable to or even surpassing that of full fine-tuning?**

Therefore, in this paper, we propose a novel point cloud fine-tuning method based on Ladder Side Tuning, called Point Ladder Tuning (PLT). While the attention mechanism in the backbone network effectively captures global semantic information, it usually lacks local detail. To this end, we introduce a hierarchical Ladder Network designed to capture local information directly from the raw input. Additionally, we propose a local-global fusion module(LGF) that adaptively aggregates local information from the Ladder Network with global information activated in the backbone network, resulting in multi-scale representations that are crucial for enhancing network performance. To better adapt the pre-trained backbone network to downstream tasks, we also propose a simple and adaptive prompt generation method, which involves learnable scaling and translation of the fused features. Each prompt captures instance-specific multi-scale features and is easy to optimize, enabling the backbone network to adjust its instance-specific features more effectively.

Table 1. The overall accuracy (%) for classical fine-tuning strategies on three variants of ScanObjectNN [46] is reported. '#TP' means the number of tunable parameters. Linear probing indicates head-tuned only.

Tuning Strategy	#TP(M)	OBJ_BG	OBJ_ONLY	PB_T50_RS
Point-MAE [32]	22.1	90.02	88.29	85.18
Linear probing	0.3	87.26(-2.76)	84.85(-3.44)	75.99(-9.19)
+ Adapter [17]	0.9	89.50(-0.52)	88.64(+0.35)	83.93(-1.25)
+ VPT [19]	0.4	87.26(-2.76)	87.09(-1.20)	81.09(-4.09)
+ LST [42]	0.8	89.67(-0.25)	89.67(+1.38)	82.75(-2.43)

Extensive experiments conducted across various point cloud datasets and settings have demonstrated the effectiveness of our method. As illustrated in the figure, our approach achieves an accuracy increase of 0.28% compared to PointMAE[32] while utilizing only 2.71% of the parameters in the PB_T5_S setting of the ScanObjectNN dataset[46], and a 3.02% increase compared to PointBert with full fine-tuning.On the ShapeNetPart dataset[52], our method achieves an instance mIoU improvement of 0.1 on PointMAE[32] and 0.3 on PointBert[56] utilizing less than 40% of the parameters used by the current state-of-the-art method PointGST. Furthermore, on the S3DIS dataset[2] for point cloud semantic segmentation, our method improves mIoU by 0.7 over PointMAE and by 1.9 over ACT[10] compared to PointGST[28], underscoring its superior performance.

Our main contributions can be summarized as follows:

- We present Point Ladder Tuning (PLT), a fine-tuning method for point clouds based on Ladder Side Tuning (LST). PLT utilizes a hierarchical Ladder Network to extract local information and a local-global fusion module to integrate this with global features, yielding rich multi-scale representations.
- To further enhance performance, we present a simple and effective prompt generation module that linearly maps the output of the local-global fusion to inject multi-scale information into the backbone network.
- Extensive experiments have demonstrated that our PLT achieves performance that is comparable to, or even surpasses, full fine-tuning across various tasks and datasets while utilizing very few parameters.

2. Related Work

2.1. 3D Pre-trained Models

Recently, self-supervised pre-trained point cloud models have garnered significant attention due to their outstanding

performance. These models are trained on vast amounts of unlabeled data and subsequently fine-tuned for various downstream tasks. Point cloud pre-training can be categorized into three main types: contrastive learning-based, reconstruction-based, and a combination of both. In contrastive learning, models like PointContrast[53] and CrossPoint[1] leverage rich semantic priors by comparing and learning features from different perspectives of a unified point cloud. PointBERT[56] adapts concepts from BERT[9], learning point cloud features by predicting masked patches and comparing them with the output features from dVAE-based point cloud tokenizers. PointMAE[32], inspired by MAE[15], directly predicts the coordinates of the masked points using an autoencoder. Meanwhile, ReCon[36] combines contrastive learning and mask reconstruction, integrating additional information such as images and text to enhance the quality of pre-training.

Currently, 3D pre-trained models are usually transferred to downstream tasks through full fine-tuning. However, this is often inefficient and may undermine the valuable prior knowledge obtained during pre-training, potentially leading to catastrophic forgetting. Therefore, in this paper, we mainly focus on strategies for efficiently and effectively transferring 3D pre-trained models to downstream tasks.

2.2. Parameter Efficient Fine-tuning

Fine-tuning pre-trained models can be costly in terms of computation and storage. To address this challenge, many researchers in NLP and computer vision have explored parameter-efficient fine-tuning techniques that enable the transfer of pre-trained knowledge to downstream tasks using only a minimal number of parameters. the Adapter-based methods[5, 17, 18] typically involve inserting lightweight networks into frozen backbone models to adjust the pre-trained architecture. For instance, AdaptFormer[5] adds adapters in parallel to the feed-forward network (FFN) for visual recognition tasks.the Prompt-based methods[19, 25] generally incorporate a small number of learnable parameters into the input sequence. For example, VPT-Deep[19] inserts learnable parameters into the input of each layer. Different from the previous two methods, LST[42] introduces additional branches that utilize the intermediate activations of the pretrained model as inputs for prediction. However, simply applying these methods to the field of 3D point clouds does not yield satisfactory results.

Recently, several PEFT methods[28, 43, 58, 62, 65] designed for pre-training 3D point cloud models have shown significant performance improvements. IDPT[58] is the first PEFT method specifically tailored for point clouds, utilizing DGCNN[48] to generate instance-level prompts, effectively replacing traditional VPT. Point-PEFT[43] employs adapters to aggregate local features during the fine-tuning process, while DAPT[65] introduces dynamic adapters that assess the importance of each token in downstream tasks, generating dynamic weights for each token. Although these methods successfully reduce training costs, achieving consistent performance improvements across various tasks remains challenging. We believe that these approaches depend on freezing the features of pre-trained models as input, which hinders the learning of discriminative local features. Furthermore, they do not fully integrate the global and local information of point clouds, a critical aspect for dense prediction tasks[8]. To this end, we propose Point Ladder Tuning(PLT), which utilizes a hierarchical Ladder network to directly extract local information from point cloud inputs and fuses it with the global features of the backbone network through attention to obtain multi-scale features. This method significantly reduces adjustable parameters and achieves remarkable performance.

3. Preliminary

Ladder Side Tuning[42] trains a ladder side network, a small and separate network that takes intermediate activations from the backbone transformer as input and makes predictions. It includes a downward projection $h_i^f = x_i W_d^T$ to decrease dimension of the feature and a gated connection to integrate the intermediate activation of the backbone f and the features of the side network g Adaptively. The output is calculated by

$$\boldsymbol{h}_i^g = \Phi(\mu_i * \boldsymbol{h}_i^f + (1-\mu_i) * \boldsymbol{h}_{i-1}^g) \tag{1}$$

where $\mu_i = \text{sigmoid}\left(\frac{\alpha_i}{T}\right)$ is a gate parameterized with a learnable zero-initialized scalar α_i and temperature T(= 0.1), Φ is a linear layer that projects the fused features.

Prompt tuning[19, 25] freezes the backbone network during fine-tuning while inserting learnable tokens into the input token sequence. Prompt tokens interact with the original tokens through attention mechanisms. Given the class token $x_{cls} \in R^{1\times d}$ and patch tokens $T \in R^{N\times d}$, where N is the num of patch tokens, we can insert prompt tokens $P_i \in R^{n\times d}$ for each layer following VPT-Deep[19], where $n \ll N$ is the num of prompt tokes.The output of i-th layer($L(\cdot)$) is expressed as follows:

$$\boldsymbol{x}_i = L_i\left([\boldsymbol{T}_{cls}; \boldsymbol{P}_i; \boldsymbol{T}]\right) \tag{2}$$

4. Methodology

Fine-tuning a pre-trained model has achieved impressive results, but it requires a significant training cost.To address this, We combine ladder side tuning with prompt tuning to propose a novel point cloud fine-tuning method called Point Ladder Tuning (PLT), which achieves a better balance between performance and parameter efficiency for 3d

CVPR
#16372

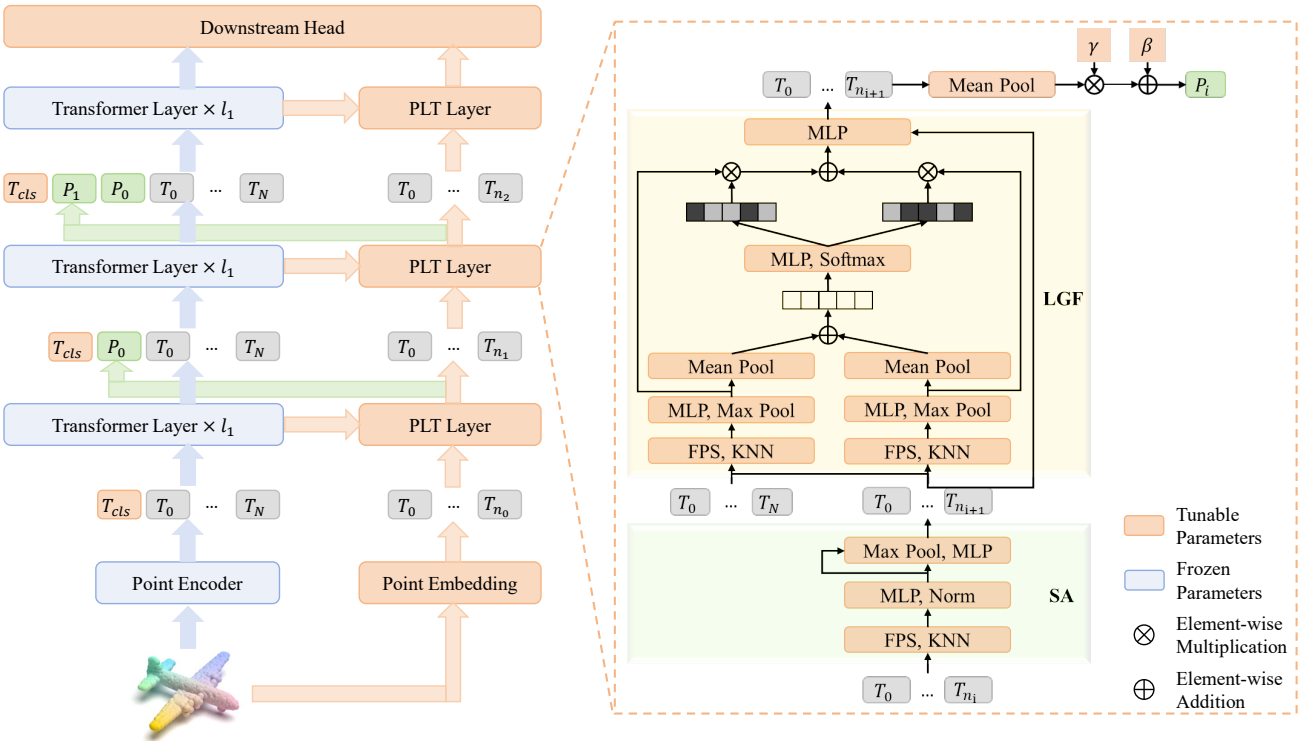

Figure 2. The overall of our PLT. During fine-tuning, we froze the pre-trained backbone network and only fine-tune the PLT branch and the class token. The PLT branch consists of two main components: 1) Set Abstraction (SA), and 2) Local-Global Fusion Module (LGF).

point cloud learning. The overall framework is illustrated in Fig..In our method, we introduce a hierarchical ladder network based on LST to extract local information from the raw input and fuse it with the global intermediate activations of the backbone network through a local-global fusion module(LGF). To further enhance performance, we also generate prompt tokens derived from the fused features, which are used to fine-tune the global features of the backbone network.

4.1. Hierarchical Ladder Network

Pre-trained models typically learn the global information of input point clouds through attention mechanisms but often overlook the importance of locality, which is crucial for downstream tasks, especially dense prediction tasks. Although Point PEF introduces a Geometry-Aware Adapter to capture local information, it still relies on the global intermediate activations of pre-trained models, which limits its ability to effectively learn local features within the original point cloud. Additionally, because pre-trained models often operate at a single scale throughout the learning process, lacking multi-scale information, this constraint further impedes performance on downstream tasks.

To address these problems, we introduce a hierarchical Ladder network to better capture the intrinsic local information of the input point cloud. The module includes multiple PLT layers that are capable of capturing point cloud features at different scales, which has been shown to be effective in subsequent experiments. The PLT Layer consists of two main components: 1) Set Abstraction (SA), and 2) Local-Global Fusion Module (LGF).

Given the input point cloud $P \in R^{N\times 3}$, where N is the num of the input point cloud, we first apply Point Embedding to map it into a high-dimensional space, obtaining the point cloud feature $F \in R^{N\times C}$, where is the dimension of the point cloud feature. Next, we use Set Abstraction (SA) to perform downsampling. This process begins with farthest point sampling (FPS) to select a set of center points $C = \{c_1, \ldots, c_n\}$. Then, we use k-nearest neighbors (KNN) to construct local neighborhoods $P(c) = \{p_c^1, \ldots, p_c^k\}$, $F(c) = \{f_c^1, \ldots, f_c^k\}$ for each center point c. Finally, we apply max pooling followed by a multi-layer perceptron (MLP) to obtain the output point cloud. The formula for this process is as follows:

$$\boldsymbol{f}_c = \varphi\left(\left[\boldsymbol{f}_c^j; \left(\boldsymbol{p}_c - \boldsymbol{p}_c^j\right)\right]\right) \tag{3}$$

$$\boldsymbol{f}_c^{'} = \boldsymbol{f}_c + \rho\left(h_{\boldsymbol{\Theta}}\left(\boldsymbol{f}_c\right)\right) \tag{4}$$

where ρ and φ represent a MLP respectively, $[\cdot,\cdot]$ represents concat operation. And $h_{\boldsymbol{\Theta}}$ represents the aggregation function. Unless otherwise specified, we use Max-Pooling.

4.2. Local-Global Fusion Module

To make better use of the local information captured in the PLT Branch and the global information captured in the pre-trained backbone network, we propose a local-global fusion module that adaptively aggregates both types of information through selective attention. Given the input point cloud P_s and their corresponding features F_s from the PLT layer, as well as the output point cloud P_m and its corresponding features F_m from the backbone network, we firstly use W to map the output features of the backbone network to match the same feature dimension as the input point cloud feature vectors of the PLT layer:

$$\boldsymbol{F}_m^{'} = \boldsymbol{F}_m W \tag{5}$$

Next, we use the same operations as Set Abstraction (SA) to extract both the global features F_g, which are derived from the interaction between the input point cloud P_s of the PLT layer and the output point cloud P_m of the backbone network, and the local features F_l, which are obtained from the interaction between the input point clouds P_s of the PLT layer.

Finally, we apply a selective attention mechanism to adaptively fuse the global and local features. This mechanism allows the model to focus on the most relevant information from each source, ensuring that both global context and fine-grained local details are effectively integrated. Firstly, we apply the aggregation function $\mathcal{A}$, with average pooling as the default, to the global and local features to obtain the corresponding global and local feature vectors:

$$\boldsymbol{f}_g, \boldsymbol{f}_l = \mathcal{A}(\boldsymbol{F}_g), \mathcal{A}(\boldsymbol{F}_l) \tag{6}$$

Then, we add the global and local features and pass them through a MLP to obtain z_g and z_l:

$$\boldsymbol{z}_g, \boldsymbol{z}_l = \alpha\left(\boldsymbol{f}_g + \boldsymbol{f}_l\right) \tag{7}$$

where α represents a MLP.

Next, we apply Softmax to compute the attention weights s_g and s_l for the global and local features:

$$\boldsymbol{s}_g^i = \frac{e^{\boldsymbol{z}_g^i}}{e^{\boldsymbol{z}_g^i} + e^{\boldsymbol{z}_l^i}}, \boldsymbol{s}_l^i = \frac{e^{\boldsymbol{z}_l^i}}{e^{\boldsymbol{z}_g^i} + e^{\boldsymbol{z}_l^i}} \tag{8}$$

Subsequently, the global and local features are weighted according to their attention scores to produce the fused multi-scale features:

$$\boldsymbol{F}_{ms} = \boldsymbol{s}_g \boldsymbol{F}_g + \boldsymbol{s}_l \boldsymbol{F}_l \tag{9}$$

Finally, we use an MLP τ to perform feature mapping and add the result to the input features:

$$\boldsymbol{F}_o = \boldsymbol{F}_s + \tau\left(\boldsymbol{F}_{ms}\right) \tag{10}$$

By selectively emphasizing the most relevant local and global features through attention, our LGF ensures the effective integration of both types of information, enhancing the model's ability to capture multi-scale features. This is crucial for tasks that require a balance between fine-grained local details and the broader global structure of point clouds, especially in dense prediction tasks.

4.3. Fine-tuning on the Backbone Network

To enable the pre-trained backbone network to generate more effective global features for downstream tasks, we propose a dynamic prompt generation method based on the multi-scale point cloud features output F_o by the Local-Global Fusion (LGF) module. Firstly, F_o is mean-pooled to obtain a compact representation. These pooled features are then scaled and translated using learnable parameters γ and β to generate multi-scale prompt p. To efficiently utilize parameters, we reuse W to align the dimensions of the multi-scale prompts with the dimensions of the backbone network features. The formula can be expressed as follows:

$$p = \left(\boldsymbol{\gamma} \times \frac{1}{n} \sum_{i}^{n} \boldsymbol{F}_o^i + \boldsymbol{\beta}\right) W^T \tag{11}$$

where n is the num of tokens in F_o.

Finally, we incorporate multi-scale prompts into the backbone network for learning. The output of the l-th transformer layer x_l can be expressed as

$$\boldsymbol{x}_l = L_l\left(\left[\boldsymbol{T}_{cls}; \boldsymbol{p}_0, \ldots, \boldsymbol{p}_{i-1}; \boldsymbol{T}\right]\right) \tag{12}$$

To enhance performance further, we follows ssf[27] and applies scaling and shifting to the output of each module within the backbone network, enabling the extraction of task-specific global information. Given an input x, the output y can be expressed as:

$$\boldsymbol{y} = \boldsymbol{s} \times \boldsymbol{x} + \boldsymbol{t} \tag{13}$$

where s and t represents scaling and shifting parameter respectively, which are learnable.

5. Experiments

5.1. Experimental Settings

To ensure a fair comparison, we used the same experimental setup as the default fine-tuning method[58, 65] for each baseline. During training, we freeze the weights of the pre-trained model and only fine-tune a small number of parameters for adding modules. All experiments were conducted on a single GeForce RTX 3090 GPU.

5.2. 3D Object Classification

Object Classification on Real-World Dataset. Pre-trained Point cloud models are typically trained on the ShapeNet

Table 2. Classification on three variants of the ScanObjectNN [46] and the ModelNet40[52], including the number of trainable parameters and overall accuracy (OA). All methods utilize the default data argumentation as the baseline[65]. * denotes reproduced results. We report ScanObjectNN[46] results without voting. ModelNet40[52] results are without and with voting, referred to (-/-).

Method	Reference	Tunable params. (M)	FLOPs (G)	ScanObjectNN			ModelNet40	
				OBJ_BG	OBJ_ONLY	PB_T50_RS	Points Num.	OA (%)
Supervised Learning Only								
PointNet [34]	CVPR 17	3.5	0.5	73.3	79.2	68.0	1k	- / 89.2
PointNet++ [35]	NeurIPS 17	1.5	1.7	82.3	84.3	77.9	1k	- / 90.7
DGCNN [48]	TOG 19	1.8	2.4	82.8	86.2	78.1	1k	- / 92.9
MVTN [13]	ICCV 21	11.2	43.7	-	-	82.8	1k	- / 93.8
PointNeXt [38]	NeurIPS 22	1.4	1.6	-	-	87.7	1k	- / 94.0
PointMLP [30]	ICLR 22	13.2	31.4	-	-	85.4	1k	- / 94.5
RepSurf-U [40]	CVPR 22	1.5	0.8	-	-	84.3	1k	- / 94.4
ADS [16]	ICCV 23	-	-	-	-	87.5	1k	- / 95.1
Self-Supervised Representation Learning (Full fine-tuning)								
OcCo [47]	ICCV 21	22.1	4.8	84.85	85.54	78.79	1k	- / 92.1
Point-BERT [56]	CVPR 22	22.1	4.8	87.43	88.12	83.07	1k	- / 93.2
MaskPoint [29]	ECCV 22	22.1	-	89.70	89.30	84.60	1k	- / 93.8
Point-MAE [32]	ECCV 22	22.1	4.8	90.02	88.29	85.18	1k	- / 93.8
Point-M2AE [60]	NeurIPS 22	15.3	3.6	91.22	88.81	86.43	1k	- / 94.0
ACT [10]	ICLR 23	22.1	4.8	93.29	91.91	88.21	1k	- / 93.7
RECON [36]	ICML 23	43.6	5.3	94.15	93.12	89.73	1k	- / 93.9
Self-Supervised Representation Learning (Efficient fine-tuning)								
Point-BERT [56] (baseline)	CVPR 22	22.1 (100%)	4.8	87.43	88.12	83.07	1k	92.7 / 93.2
+ IDPT [58]	ICCV 23	1.7 (7.69%)	7.2	88.12(+0.69)	88.30(+0.18)	83.69(+0.62)	1k	92.6(-0.1) / 93.4(+0.2)
+ DAPT [65]	CVPR 24	1.1 (4.97%)	5.0	91.05(+3.62)	89.67(+1.55)	85.43(+2.36)	1k	93.1(+0.4) / 93.6(+0.4)
+ PointGST [28]	Arxiv 24	**0.6** (**2.71**%)	5.0	91.39(+3.96)	89.67(+1.55)	85.64(+2.57)	1k	93.4(+0.7) / 93.8(+0.6)
+ LST [42]	NeurIPS 22	0.8 (3.38%)	-	89.15(+2.72)	89.50(+1.38)	83.17(+0.10)	1k	92.9(+0.2) / 93.3(+0.1)
+ PLT (ours)	-	**0.6** (**2.71**%)	5.0	91.57(+4.14)	89.85(+1.73)	86.09(+3.02)	1k	93.5(+0.8) / 94.2(+1.0)
Point-MAE [32] (baseline)	ECCV 22	22.1 (100%)	4.8	90.02	88.29	85.18	1k	93.2 / 93.8
+ IDPT [58]	ICCV 23	1.7 (7.69%)	7.2	91.22(+1.20)	90.02(+1.73)	84.94(-0.24)	1k	93.3(+0.1) / 94.4(+0.6)
+ Point-PEFT* [43]	AAAI 24	0.7 (3.13%)	-	90.19(+0.17)	89.50(+1.21)	84.35(-0.83)	1k	94.2(+0.4) / -
+ DAPT [65]	CVPR 24	1.1 (4.97%)	5.0	90.88(+0.86)	90.19(+1.90)	85.08(-0.10)	1k	93.5(+0.3) / 94.0(+0.2)
+ PPT* [62]	Arxiv 24	1.1 (4.97%)	5.0	89.50(-0.52)	89.50(+1.21)	84.91(-0.27)	1k	93.7(+0.5) / -
+ PointGST[28]	Arxiv 24	**0.6** (**2.71**%)	5.0	91.74(+1.72)	90.19(+1.90)	85.29(+0.11)	1k	93.5(+0.3) / 94.0(+0.2)
+ LST [42]	NeurIPS 22	0.8 (3.38%)	5.0	89.67(-0.25)	89.67(+1.38)	82.75(-2.43)	1k	93.2(+0.0) / 93.8(+0.0)
+ PLT (ours)	-	**0.6** (**2.71**%)	5.0	90.88(+0.86)	90.02(+1.73)	85.46(+0.28)	1k	93.8(+0.6) / 94.0(+0.2)

dataset[4], which consists of clean, uniformly distributed point clouds. However, real-world point clouds often suffer from issues such as noise and missing point, leading to diverse and challenging distributions. To evaluate the performance in these more realistic conditions, we use the ScanObjectNN dataset[46], which contains approximately 15k point cloud samples across 15 categories. These point clouds are captured in indoor scenes and often include background interference and occlusions from other objects. As shown in Tab.2 We conduct experiments on on three variants of the ScanObjectNN dataset[46] (OBJ_BG, OBJ_ONLY, and PB_T50_RS), using two baseline models, PointBERT[56] and PointMAE[32], to assess the effectiveness of our PLT. Notably, PLT achieved better accuracy than full fine-tuning across all settings while utilizing only 2.71% of the parameters. Especially, PLT achieved increases of 4.14%, 1.73%, and 3.02% in the three ScanObjectNN[46] variants on Point-BERT. Compared to the state-of-the-art model PointGST, our LST improved accuracy by 0.45% on PointBERT[56] and 0.17% on PointMAE[32] under the most challenging setting, PB_T50_RS, in ScanObjectNN[46].

Object Classification on Synthetic Dataset. We also conduct experiments on the ModelNet dataset[52], which includes 12,311 clean 3D CAD models across 40 categories. Following DAPT[65], we split the ModelNet40 dataset into 9,843 training samples and 2,468 testing samples. During training, we applied standard data augmentation techniques, including random scaling and random translation. As shown in Tab.2, without voting, our PLT achieved accuracy rates of 93.8% and 93.5% on PointMAE and PointBERT, respectively, which were 0.6% and 0.8% higher than full fine-tuning.. With voting, PLT continues to outperform full fine-tuning, especially on PointBERT, with an accuracy increase of 1.0%.

Few-shot Learning. We further conduct experiments on

Table 3. Few-shot learning on ModelNet40[52]. Overall accuracy (%)±the standard deviation (%) without voting is reported.

Methods	Reference	5-way		10-way	
		10-shot	20-shot	10-shot	20-shot
with Self-Supervised Representation Learning (Full fine-tuning)					
OcCo [47]	ICCV 21	94.0±3.6	95.9±2.3	89.4±5.1	92.4±4.6
Point-BERT [56]	CVPR 22	94.6±3.1	96.3±2.7	91.0±5.4	92.7±5.1
MaskPoint [29]	ECCV 22	95.0±3.7	97.2±1.7	91.4±4.0	93.4±3.5
Point-MAE [32]	ECCV 22	96.3±2.5	97.8±1.8	92.6±4.1	95.0±3.0
Point-M2AE [60]	NeurIPS 22	96.8±1.8	98.3±1.4	92.3±4.5	95.0±3.0
ACT [10]	ICLR 23	96.8±2.3	98.0±1.4	93.3±4.0	95.6±2.8
VPP [37]	NeurIPS 23	96.9±1.9	98.3±1.5	93.0±4.0	95.4±3.1
I2P-MAE [61]	CVPR 23	97.0±1.8	98.3±1.3	92.6±5.0	95.5±3.0
RECON [36]	ICML 23	97.3±1.9	98.9±1.2	93.3±3.9	95.8±3.0
with Self-Supervised Representation Learning (Efficient fine-tuning)					
Point-BERT [56] (baseline)	CVPR 22	94.6±3.1	96.3±2.7	91.0±5.4	92.7±5.1
+ IDPT [58]	ICCV 23	96.0±**1.7**	97.2±2.6	91.9±4.4	93.6±3.5
+ DAPT [65]	CVPR 24	95.8±2.1	97.3±1.3	92.2±4.3	94.2±3.4
+ PointGST [28]	Arxiv 24	96.5±2.4	97.9±2.0	92.7±4.2	95.0±**2.8**
+ PLT (**ours**)	-	**96.9**±2.0	**98.8±1.1**	**93.3±4.0**	**95.5**±3.1
Point-MAE [32] (baseline)	ECCV 22	96.3±2.5	97.8±1.8	92.6±4.1	95.0±3.0
+ IDPT [58]	ICCV 23	97.3±2.1	97.9±1.1	92.8±4.1	95.4±**2.9**
+ DAPT [65]	CVPR 24	96.8±**1.8**	98.0±1.0	93.0±**3.5**	**95.5**±3.2
+ PLT (**ours**)	-	**97.2**±2.2	**98.9±0.9**	**93.2**±4.2	**95.5±2.9**

ModelNet40 to evaluate its transfer learning ability in few-shot setting. Following prior work[32, 58, 65], we adopt the n-way, n-shot setting. As shown in Tab.3, PLT outperforms fully fine-tuned models and state-of-the-art models like IDPT[58] and DAPT[65] in most cases, demonstrating its effectiveness in few-shot learning.

Compared with Other PETL Methods. We also compare PEFT methods from different fields on the most challenging PB_T50_RS variant of ScanObjectNN[46]. As shown in the Tab.4, our methods are more effective than those proposed in NLP, 2D, and 3D areas.

5.3. 3D Dense Prediction Task

For dense prediction tasks, such as part segmentation and semantic segmentation, we employ a prediction head similar to that of PointNext. This design allows us to leverage multi-scale information, enhancing performance while minimizing the number of trainable parameters.

We validate the effectiveness of PLT on ShapeNetPart dataset[4]. As shown in the Tab.5, PLT achieves comparable results on Inst. mIoU, while significantly improving Cls. mIoU, particularly on PointBERT, where it outperforms DAPT by 0.5%.

We evaluate the proposed PLT on the semantic segmentation task using the S3DIS dataset[2], with results shown in Tab.7. It is clear that our PLT significantly outperforms other methods. When using ACT as the baseline, PLT achieves improvements of 7.2%, 4.7%, 4.8%, and 1.9% over IDPT, PointPEF, DAPT, and PointGST, respectively. Similarly, when using PointMAE as the baseline, PLT consistently outperforms other methods, further validating its effectiveness in dense prediction tasks.

Table 4. Comparisons of parameter efficient transfer learning methods from NLP and 2D Vision on the hardest variant of ScanObjectNN [46]. Overall accuracy (%) without voting is reported. #TP represents the tunable parameters. * denotes reproduced results.

Method	Reference	Design for	#TP (M)	PB_T50_RS
Point-MAE [32]	ECCV 22	-	22.1	85.18
Linear probing	-	-	0.3	75.99
+ Adapter [17]	ICML 19	NLP	0.9	83.93
+ Perfix tuning [25]	ACL 21	NLP	0.7	77.72
+ BitFit [57]	ACL 21	NLP	0.3	82.62
+ LST [42]	NeurIPS 22	NLP	1.1	82.75
+ LoRA [18]	ICLR 22	NLP	0.9	81.74
+ DEPT [41]	ICLR 24	NLP	0.3	79.70
+ FourierFT [11]	ICML 24	NLP	0.3	78.57
+ VPT-Deep [19]	ECCV 22	2D	0.4	81.09
+ AdaptFormer [5]	NeurIPS 22	2D	0.9	83.45
+ SSF [27]	NeurIPS 22	2D	0.4	82.58
+ FacT [20]	AAAI 23	2D	0.5	78.76
+ BI-AdaptFormer [21]	ICCV 23	2D	0.4	83.66
+ SCT [64]	IJCV 24	2D	0.3	80.40
+ IDPT [58]	ICCV 23	3D	1.7	84.94
+ Point-PEFT* [43]	AAAI	3D	0.7	84.35
+ DAPT [65]	CVPR 24	3D	1.1	85.08
+ PPT* [62]	Arxiv 24	3D	1.1	84.91
+ PointGST [28]	Arxiv 24	3D	0.6	85.29
+ PLT (**ours**)	-	3D	0.6	**85.46**

Table 5. Part segmentation on the ShapeNetPart [4]. The mIoU for all classes (Cls.) and for all instances (Inst.) are reported. #TP represents the tunable parameters. * denotes reproduced results.

Methods	Reference	#TP (M)	Cls. mIoU (%)	Inst. mIoU (%)
Supervised Learning Only				
PointNet [34]	CVPR 17	-	80.39	83.7
PointNet++ [35]	NeurIPS 17	-	81.85	85.1
DGCNN [48]	TOG 19	-	82.33	85.2
APES [49]	CVPR 23	-	83.67	85.8
Self-Supervised Representation Learning (Full fine-tuning)				
OcCo [47]	ICCV 21	27.09	83.42	85.1
MaskPoint [29]	ECCV 22	-	84.60	86.0
Point-BERT [56]	CVPR 22	27.09	84.11	85.6
Point-MAE [32]	ECCV 22	27.06	84.19	86.1
ACT [10]	ICLR 23	27.06	84.66	86.1
Self-Supervised Representation Learning (Efficient fine-tuning)				
Point-BERT [56] (baseline)	CVPR 22	27.09	84.11	85.6
+ IDPT* [58]	ICCV 23	5.69	83.50	85.3
+ DAPT [65]	CVPR 24	5.65	83.83	85.5
+ PLT (**ours**)	-	**2.08**	**83.85**	**86.0**
Point-MAE [32] (baseline)	ECCV 22	27.06	84.19	86.1
+ IDPT [58]	ICCV 23	5.69	83.79	85.7
+ DAPT [65]	CVPR 24	5.65	84.01	85.7
+ PLT (**ours**)	-	**2.08**	83.90	**85.9**

5.4. Ablation Study

Analysis on each component. We conduct experiments to prove the effectiveness of the proposed components of Our PLT. As shown in the Tab.8, when all modules are used, the performance on PB T50 RS reaches 85.46%. However, removing any of the proposed modules leads to performance degradation, further highlighting the necessity of each module.

Table 6. Ablation on hyper-parameters and settings of LST, including the num of neighbors K, feature dim d and the num of layers in Hierarchical Ladder Network (HLN). The tunable parameters (#TP) and the overall accuracy (%) on the hardest variant of ScanObjectNN[46] are reported.

(a) Ablation on K in HLN.

K	#TP (M)	PB_T50_RS
[16, 16, 16]	0.60	84.46
[4, 4, 4]	0.60	84.66
[4, 8, 16]	0.60	84.91
[16, 8, 4]	0.60	**85.46**

(b) Ablation on d in HLN.

Feature Dim	#TP (M)	PB_T50_RS
[8, 16, 32, 64]	0.46	83.17
[32, 64, 128, 256]	1.00	84.25
[16, 32, 64, 128]	0.60	**85.46**

(c) Ablation on the num of layers.

Layer num	#TP (M)	PB_T50_RS
1	0.40	83.55
2	0.45	84.84
3	0.60	**85.46**
4	1.09	85.05

Table 7. Semantic segmentation on the S3DIS [2]. The mean accuracy (mAcc) and mean IoU (mIoU) are reported. Params. represents the trainable parameters. #TP represents the tunable parameters.

Methods	Reference	#TP (M)	mAcc (%)	mIoU (%)
Point-MAE [32] (baseline)	ECCV 22	27.02	69.9	60.8
+ Linear probing	ICCV 23	5.20	63.4	52.5
+ IDPT [58]	ICCV 23	5.64	65.0	53.1
+ Point-PEFT [43]	ICCV 23	5.58	66.5	56.0
+ DAPT [65]	CVPR 24	5.61	67.2	56.2
+ PointGST [28]	Arxiv 24	5.55	68.4	58.6
+ PLT (**ours**)	-	**2.04**	**70.7**	**59.3**
ACT [10] (baseline)	ICLR 23	27.02	71.1	61.2
+ Linear probing	ICCV 23	5.20	64.1	52.0
+ IDPT [58]	ICCV 23	5.64	64.1	52.1
+ Point-PEFT [43]	ICCV 23	5.58	66.0	54.6
+ DAPT [65]	CVPR 24	5.61	64.7	54.5
+ PointGST [28]	Arxiv 24	5.55	67.6	57.4
+ PLT (**ours**)	-	**2.04**	**70.8**	**59.3**

Table 8. The effect of each component of our LST. The tunable parameters (#TP) and the overall accuracy (%) on the hardest variant of ScanObjectNN[46] are reported.

SSF	DP	HLN	#TP (M)	PB_T50_RS
Full fine-tuning			22.1	85.18
Linear Probing			0.27	75.99
	✔	✔	0.49	84.52
✔		✔	0.60	84.66
✔			0.37	83.34
✔	✔	✔	0.60	**85.46**

Table 9. Fusion way for local and global information in point clouds. The tunable parameters (#TP) and the overall accuracy (%) on the hardest variant of ScanObjectNN[46] 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
LGA with Sigmoid	0.59	84.59
LGA with Softmax	0.60	**85.46**

Ablation on hyper-parameters. We conduct an in-depth exploration of the hyperparameters of Hierarchical Ladder Network (HLN), as shown in the Tab.6, and found that the performance is optimal when the number of layers is 3, with feature dimensions of [16, 32, 64, 128], and the number of neighbors set to [16, 8, 4].

Fusion way for or local and global information in point clouds. As shown in the Tab.9, we explored various fusion methods and found that the proposed LGF with Softmax achieved the best performance. Additionally, when using only local information, the performance was worse than using only global information. This may be due to the fact that local information is learned from scratch, while global information benefits from pre-trained prior knowledge.

Table 10. Comparison between HLN and PLT. The tunable parameters (#TP) and the overall accuracy (%) on the hardest variant of ScanObjectNN are reported.

Method	#TP (M)	PB_T50_RS
Point-MAE	22.1	85.18
HLN	0.2	80.74
PLT(Ours)	0.6	**85.46**

Comparison between HLN and PLT. As shown in the Tab.10, when training with only HLN, we observed a significant performance drop of 4.72% compared to full fine-tuning, likely due to the absence of pre-trained global prior information. However, by using our proposed PLT to integrate the features of both HLN and the backbone network, we achieved superior performance than full fine-tuning.

6. Conclusion and Limitation

In this study, we propose a parameter-efficient fine-tuning strategy, named PLT, for point cloud analysis. PLT effectively freezes the parameters of the backbone network and utilizes a hierarchical Ladder Network (HLN) to directly extract local information from the input point cloud. To aggregate this local information with the backbone network's intermediate global activation, we introduce a Local-Global Fusion (LGF) module. Additionally, to further enhance performance, fused multi-scale features are used to generate dynamic prompts, enabling the backbone network to produce global features optimized for downstream tasks. Our method demonstrates impressive performance in tasks such as object classification and dense prediction, while maintaining a minimal parameter footprint. However, a limitation of this approach is that it has not yet been validated on additional tasks, such as point cloud object detection and point cloud generation, which we leave as future work.

References

[1] Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9902–9912, 2022. 1, 3

[2] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1534–1543, 2016. 2, 7, 8

[3] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 1

[4] 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. 6, 7

[5] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *NeurIPS*, 35:16664–16678, 2022. 2, 3, 7

[6] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 1

[7] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9640–9649, 2021. 1

[8] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022. 3

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1, 3

[10] Runpei Dong, Zekun Qi, Linfeng Zhang, Junbo Zhang, Jianjian Sun, Zheng Ge, Li Yi, and Kaisheng Ma. Autoencoders as cross-modal teachers: Can pretrained 2d image transformers help 3d representation learning? In *ICLR*, 2022. 1, 2, 6, 7, 8

[11] Ziqi Gao, Qichao Wang, Aochuan Chen, Zijing Liu, Bingzhe Wu, Liang Chen, and Jia Li. Parameter-efficient fine-tuning with discrete fourier transform. In *International Conference on Machine Learning*, 2024. 7

[12] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7:187–199, 2021. 1

[13] Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. Mvtn: Multi-view transformation network for 3d shape recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1–11, 2021. 6

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 1

[15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 1, 3

[16] Cheng-Yao Hong, Yu-Ying Chou, and Tyng-Luh Liu. Attention discriminant sampling for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14429–14440, 2023. 6

[17] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR, 2019. 2, 3, 7

[18] Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *ICLR*, 2021. 3, 7

[19] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022. 2, 3, 7

[20] Shibo Jie and Zhi-Hong Deng. Fact: Factor-tuning for lightweight adaptation on vision transformer. In *Proceedings of the AAAI conference on artificial intelligence*, pages 1060–1068, 2023. 2, 7

[21] Shibo Jie, Haoqing Wang, and Zhi-Hong Deng. Revisiting the parameter efficiency of adapters from the perspective of precision redundancy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17217–17226, 2023. 7

[22] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035, 2021. 2

[23] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020. 1

[24] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. 2

[25] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Annual Meeting of the Association for Computational Linguistics*, 2021. 2, 3, 7

[26] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31, 2018. 1

[27] Dongze Lian, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Scaling & shifting your features: A new baseline

for efficient model tuning. *Advances in Neural Information Processing Systems*, 35:109–123, 2022. 2, 5, 7

[28] Dingkang Liang, Tianrui Feng, Xin Zhou, Yumeng Zhang, Zhikang Zou, and Xiang Bai. Parameter-efficient fine-tuning in spectral domain for point cloud learning. *arXiv preprint arXiv:2410.08114*, 2024. 2, 3, 6, 7, 8

[29] Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. In *European Conference on Computer Vision*, pages 657–675. Springer, 2022. 1, 6, 7

[30] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*, 2022. 1, 6

[31] Sanjaya K Panda, Subhrajit Nag, and Prasanta K Jana. A smoothing based task scheduling algorithm for heterogeneous multi-cloud environment. In *2014 International Conference on Parallel, Distributed and Grid Computing*, pages 62–67. IEEE, 2014. 1

[32] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pages 604–621. Springer, 2022. 1, 2, 3, 6, 7, 8

[33] Chunghyun Park, Yoonwoo Jeong, Minsu Cho, and Jaesik Park. Fast point transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16949–16958, 2022. 1

[34] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 6, 7

[35] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 1, 6, 7

[36] Zekun Qi, Runpei Dong, Guofan Fan, Zheng Ge, Xiangyu Zhang, Kaisheng Ma, and Li Yi. Contrast with reconstruct: Contrastive 3d representation learning guided by generative pretraining. In *International Conference on Machine Learning*, pages 28223–28243. PMLR, 2023. 1, 3, 6, 7

[37] Zekun Qi, Muzhou Yu, Runpei Dong, and Kaisheng Ma. Vpp: Efficient conditional 3d generation via voxel-point progressive representation. *Advances in Neural Information Processing Systems*, 36, 2024. 7

[38] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in neural information processing systems*, 35:23192–23204, 2022. 1, 6

[39] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020. 1

[40] Haoxi Ran, Jun Liu, and Chengjie Wang. Surface representation for point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18942–18952, 2022. 6

[41] Zhengxiang Shi and Aldo Lipani. Dept: Decomposed prompt tuning for parameter-efficient fine-tuning. In *International Conference on Learning Representations*, 2024. 7

[42] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. *Advances in Neural Information Processing Systems*, 35:12991–13005, 2022. 2, 3, 6, 7

[43] Yiwen Tang, Ray Zhang, Zoey Guo, Xianzheng Ma, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Point-peft: Parameter-efficient fine-tuning for 3d pre-trained models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5171–5179, 2024. 3, 6, 7, 8

[44] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024. 1

[45] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 1

[46] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1588–1597, 2019. 1, 2, 6, 7, 8

[47] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J Kusner. Unsupervised point cloud pre-training via occlusion completion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9782–9792, 2021. 1, 6, 7

[48] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics*, 38(5):1–12, 2019. 1, 3, 6, 7

[49] Chengzhi Wu, Junwei Zheng, Julius Pfrommer, and Jürgen Beyerer. Attention-based point cloud edge sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5333–5343, 2023. 7

[50] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022. 1

[51] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024. 1

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

Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1912–1920, 2015. 2, 6, 7

[53] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 574–591. Springer, 2020. 1, 3

[54] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9653–9663, 2022. 1

[55] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. In *European conference on computer vision*, pages 668–684. Springer, 2022. 1

[56] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19313–19322, 2022. 1, 2, 3, 6, 7

[57] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Annual Meeting of the Association for Computational Linguistics*, 2022. 2, 7

[58] Yaohua Zha, Jinpeng Wang, Tao Dai, Bin Chen, Zhi Wang, and Shu-Tao Xia. Instance-aware dynamic prompt tuning for pre-trained point cloud models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14161–14170, 2023. 3, 5, 6, 7, 8

[59] Cheng Zhang, Haocheng Wan, Xinyi Shen, and Zizhao Wu. Patchformer: An efficient point transformer with patch attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11799–11808, 2022. 1

[60] 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. *Advances in neural information processing systems*, 35:27061–27074, 2022. 1, 6, 7

[61] Renrui Zhang, Liuhui Wang, Yu Qiao, Peng Gao, and Hongsheng Li. Learning 3d representations from 2d pre-trained models via image-to-point masked autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21769–21780, 2023. 7

[62] Shaochen Zhang, Zekun Qi, Runpei Dong, Xiuxiu Bai, and Xing Wei. Positional prompt tuning for efficient 3d representation learning. *arXiv preprint arXiv:2408.11567*, 2024. 3, 6, 7

[63] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 1

[64] Henry Hengyuan Zhao, Pichao Wang, Yuyang Zhao, Hao Luo, Fan Wang, and Mike Zheng Shou. Sct: A simple baseline for parameter-efficient fine-tuning via salient channels. *International Journal of Computer Vision*, 132(3):731–749, 2024. 7

[65] Xin Zhou, Dingkang Liang, Wei Xu, Xingkui Zhu, Yihan Xu, Zhikang Zou, and Xiang Bai. Dynamic adapter meets prompt tuning: Parameter-efficient transfer learning for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14707–14717, 2024. 3, 5, 6, 7, 8