Point-PEFT: Parameter-Efficient Fine-Tuning for 3D Pre-trained Models

Yiwen Tang[1*], Ray Zhang[1*], Zoey Guo[1*],
Xianzheng Ma[1], Dong Wang[1], Zhigang Wang[1], Bin Zhao[1,2], Xuelong Li[1,2],

[1] Shanghai AI Laboratory, [2] Northwestern Polytechnical University

Abstract

*The popularity of pre-trained large models has revolutionized downstream tasks across diverse fields, such as language, vision, and multi-modality. To minimize the adaption cost for downstream tasks, many **P**arameter-**E**fficient **F**ine-**T**uning (**PEFT**) techniques are proposed for language and 2D image pre-trained models. However, the specialized PEFT method for 3D pre-trained models is still under-explored. To this end, we introduce **Point-PEFT**, a novel framework for adapting point cloud pre-trained models with minimal learnable parameters. Specifically, for a pre-trained 3D model, we freeze most of its parameters, and only tune the newly added PEFT modules on downstream tasks, which consist of a Point-prior Prompt and a Geometry-aware Adapter. The Point-prior Prompt adopts a set of learnable prompt tokens, for which we propose to construct a memory bank with domain-specific knowledge, and utilize a parameter-free attention to enhance the prompt tokens. The Geometry-aware Adapter aims to aggregate point cloud features within spatial neighborhoods to capture fine-grained geometric information through local interactions. Extensive experiments indicate that our Point-PEFT can achieve better performance than the full fine-tuning on various downstream tasks, while using **only 5% of the trainable parameters**, demonstrating the efficiency and effectiveness of our approach. Code is released at* https://github.com/Ivan-Tang-3D/PEFT-3D.

1. Introduction

The recent advancements in large-scale pre-training with numerous data have gained widespread attention in both industry and academia. In natural language processing, GPT series [2, 20] pre-trained by extensive text corpora exhibit superior language generative capabilities and interactivity. For 2D image recognition, ViT [5] and the multi-modal CLIP [21] can also reveal strong visual generalizability and robustness. However, the full fine-tuning of these large models normally requires substantial computation resources. To alleviate this, many efforts on parameter-efficient fine-tuning (PEFT) have been proposed and applied in both language and image domains, significantly reducing the consumption of tuning resources. The principal concept involves freezing most trained parameters in large models, and optimizing only the newly inserted PEFT modules on downstream tasks. The popular techniques include adapters [12], prompt tuning [14, 15], Low-Rank Adaptation (LoRA) [13], and side tuning [31].

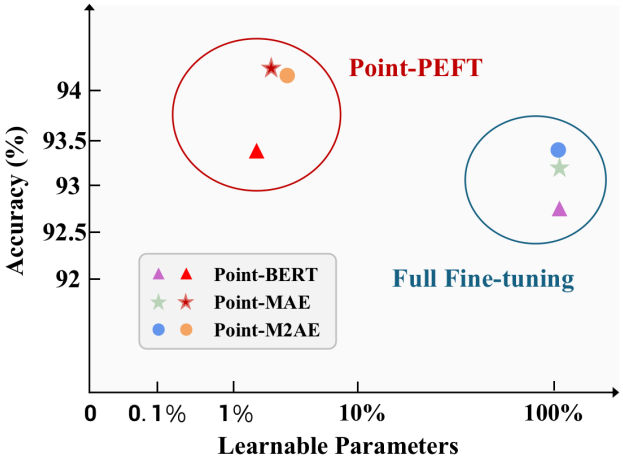

Figure 1. **Our Point-PEFT vs. Full Fine-tuning** on ModelNet40 [25] dataset. We compare the fine-tuning of three popular pre-trained models, Point-BERT [28], Point-MAE [17], and Point-M2AE [32], where our Point-PEFT achieves superior performance and parameter efficiency.

In 3D domains, pre-trained models for point clouds have also shown promising results, e.g., Point-MAE [17], Point-M2AE [32], and I2P-MAE [37]. However, the downstream adaption of these 3D transformers is dominated by expensive full fine-tuning, and the specialized 3D PEFT method still remains an open question. Therefore, inspired by the success in language and 2D methods, we ask the question: ***can we develop a PEFT framework specialized for***

* Equal Contribution.

3D point clouds with both efficiency and effectiveness?

To tackle this issue, we propose **Point-PEFT**, a novel parameter-efficient fine-tuning framework for 3D pre-trained models, as shown in Figure 1. Aiming at the sparse and irregular characters of point clouds, we introduce a Point-prior Prompt and a Geometry-aware Adapter, which can efficiently incorporate downstream 3D semantics into the pre-trained models. On different downstream 3D tasks, we freeze most of the pre-trained parameters, and only fine-tune the task-specific heads and our Point-PEFT components, which we illustrate as follows:

- **Point-prior Prompt.** Before every transformer block, we prepend a set of learnable prompt tokens to the input point cloud tokens, which are enhanced by a proposed point-prior bank with parameter-free attention mechanisms. The bank is constructed by downstream training-set 3D features, which enhances prompt tokens with domain-specific 3D knowledge.
- **Geometry-aware Adapter.** Within each transformer block, we insert the Geometry-aware Adapter after the pre-trained self-attention layer and Feed-Forward Networks (FFN). As the pre-trained attention mainly explores long-range dependencies of the global shape, our adapters are complementary to aggregate local geometric information and grasp the fine-grained 3D structures.

With the proposed two components, our Point-PEFT achieves better performance than full fine-tuning, while utilizing only 5% of the trainable parameters. As an example, for the pre-trained Point-MAE [17], Point-PEFT with 0.8M parameters attains 94.2% on ModelNet40 [25], and 89.1% on ScanObjectNN [24], surpassing the full fine-tuning with 22.1M parameters by +1.0% and +1.0%, respectively. We also evaluate Point-PEFT on other 3D pre-trained models with competitive results and efficiency, e.g., Point-BERT [28] and Point-M2AE [32], which fully indicates our generalizability and significance.

The contributions of our paper are as follows:

- We propose Point-PEFT, a specialized PEFT framework for 3D pre-trained models, which achieves competitive performance to full fine-tuning, and significantly reduces the computational resources.
- We develop a Geometry-aware Adapter to extract fine-grained local geometries, and a Point-prior Prompt with parameter-free attention, leveraging domain-specific knowledge to facilitate the downstream fine-tuning.
- Extensive experiments indicate the superior effectiveness and efficiency of our approach, which has the potential to serve as a 3D PETT baseline for future research.

2. Related Work

Pre-training in 3D Vision. The recent spotlight in the 3D domain has shifted from the supervised training [18, 19] towards self-supervised and large-model pre-training methods due to the challenge of data scarcity. These approaches adopt pretext tasks to pre-train large models, learning the latent representations embedded within point clouds. When fine-tuning on downstream tasks, such as classification [24, 25], segmentation [27], and 3D visual grounding [8], the pre-trained weights are optimized to acquire the knowledge related to the task. Some prior works [1, 26] utilize contrastive pretext tasks to pre-train the model, discriminating the different views of a single instance from views of other instances. Some concurrent works [33, 39] follow a training-free paradigm, leveraging pre-trained models like CLIP [21] for downstream tasks. More research works [17, 28] introduce the masked point modeling strategy for a stronger 3D encoder. Point-BERT [28] employs a point tokenizer to obtain the point tokens. Then the Encoder-Decoder architecture is used to model and predict masked point tokens. Point-MAE [17] and Point-M2AE [32] directly utilize Masked Autoencoders (MAE) [11], achieving superior representation capabilities. Recently, I2P-MAE [37], ACT [4], and Point-Bind [9] integrate rich knowledge from pre-trained multi-modal encoders to assist in 3D learning, indicating the potential of introducing external guidance. Joint-MAE [7] and Pi-MAE [3] conduct 2D-3D joint pre-training to incorporate cross-modal knowledge. Despite the success of 3D pre-training, the adaption for downstream tasks still demands the resource-intensive full fine-tuning method. Thus, we explore the PEFT techniques in the 3D domain for parameter-efficient fine-tuning.

Parameter-efficient Fine-tuning. Given that the full fine-tuning of large models is both computationally intensive and resource-demanding, the Parameter-Efficient Fine-Tuning (PEFT) approaches are proposed to address the challenge by freezing the trained weights and introducing the newly trainable modules. Various PEFT techniques have been proposed with favorable performance, including adapters [6, 12], prompt tuning [14, 15], Low-Rank Adaptation (LoRA) [10, 13], bias tuning [29] and side tuning [23, 31]. Specifically, the adapter tuning inserts additional bottleneck-shaped neural networks within blocks of the pre-trained model to learn task-specific representations. Prompt tuning facilitates task adaption by prepending natural language prompts or learnable prompt tokens to the input. The LoRA technique employs a low-rank decomposition approach to learn the adaptation matrix in each block. Bias tuning achieves competitive results by adjusting the model's bias terms, and side tuning only adjusts side mod-

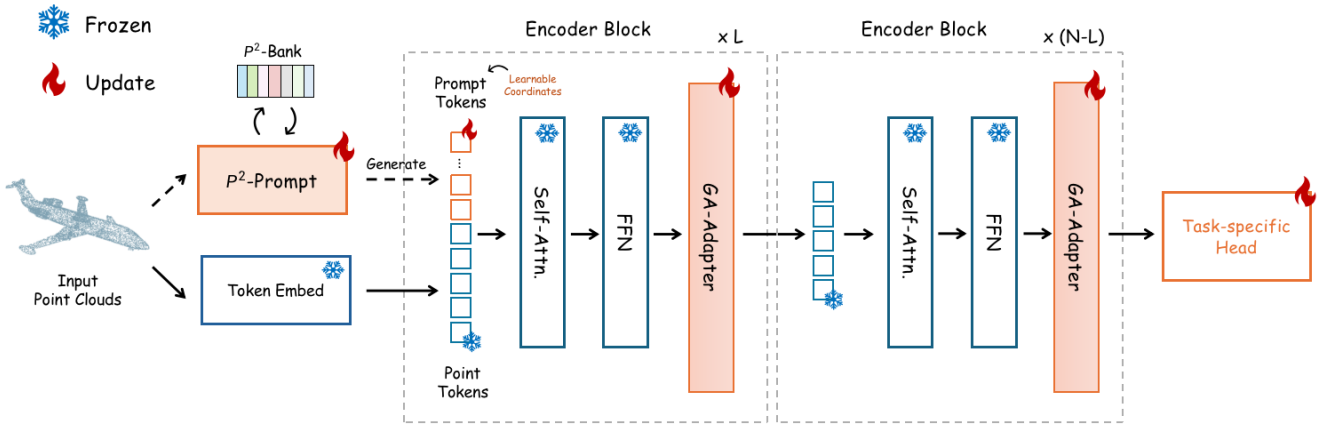

Figure 2. **Overall Pipeline of Point-PEFT.** For efficiently fine-tuning a pre-trained 3D encoder, our Point-PEFT contains two components: a Point-prior Prompt (P^2-Prompt) in the first L blocks, which aggregates prior 3D knowledge from a P^2-Bank module, and a Geometry-aware Adapter inserted at the end of each block to effectively grasp the local geometric information.

ules concurrent to pre-trained networks. One concurrent work IDPT [30] also introduces PEFT into 3D point cloud transformers, but only investigates prompt tuning methods. In this paper, we propose a PEFT framework specialized for the 3D domain, which introduces a Point-prior Prompt and Geometry-aware Adapter. Different from existing techniques, the former utilizes 3D domain-specific knowledge to enhance the prompt tokens, and the latter grasps the local geometric information.

3. Method

We illustrate the details of our Point-PEFT framework for efficiently fine-tuning 3D point cloud pre-trained models. We first present our overall pipeline in Section 3.1. Then, in Section 3.2 and 3.3, we respectively elaborate on the designs for Point-prior Prompt and Geometry-aware Adapter.

3.1. Overall Pipeline

As shown in Figure 2, given a pre-trained 3D transformer $\mathrm{E}_{3\mathrm{D}}(\cdot)$ with 12 blocks and a specific downstream task, we freeze most of its parameters for fine-tuning, and only update our introduced Point-PEFT modules, task-specific heads, and all the bias terms within the transformer blocks [29].

For an input point cloud PC, we follow the original pipeline of the pre-trained transformer to first encode it into M point tokens via the 'Token Embed' module, which normally consists of a mini-PointNet [18]. We denote the point tokens as $F_0 \in \mathbb{R}^{M\times D}$, where D denotes the feature dimension of the transformer. Then, we prepend our proposed K-length Point-prior Prompt, denoted as $P_0 \in \mathbb{R}^{K\times D}$ to these point tokens. Each token of P_0 is assigned with a learnable 3D coordinate to indicate its spatial location. Specifically, P_0 is generated by a 'P^2-Prompt$(\cdot)$' module, which takes the point cloud PC as input, and aggregates domain-specific knowledge from a constructed 'P^2-Bank', as shown in Figure 2. We formulate it as

$$P_0 = P^2\text{-Prompt}(PC), \tag{1}$$
$$C_0 = \text{Concat}\left(P_0, F_0\right), \tag{2}$$

where $C_0 \in R^{(K+M)\times D}$ denotes the initial input tokens for the first transformer block.

For the i-th transformer block ($2 \le i \le 12$), we denote the point tokens from the last block as F_{i-1}, and concatenate them with the Point-prior Prompt P_i, which obtains C_i as the input tokens. Then, we feed C_i into the pre-trained self-attention layer and the Feed-Forward Networks (FFN) with residual connections. After that, we adopt our introduced Geometry-aware Adapter ('GA-Adapter') to encode fine-grained local 3D structures, formulated as

$$C_i' = \text{FFN}\big(\text{Self-Attn.}(C_i)\big), \tag{3}$$
$$F_i = \text{GA-Adapter}(C_i'), \tag{4}$$

where F_i denotes the output point features from the i-th block. Note that the prompt tokens are only adopted in the earlier L blocks for better adapting shallower point features. After all 12 transformer blocks, the learnable downstream task head is adopted to produce the final predictions.

3.2. Point-prior Prompt

As shown in Figure 3, our adopted prompt tokens P_i for the i-th transformer block are generated by a constructed point-prior bank and parameter-free attention.

To create the bank, we employ the pre-trained 3D transformer $\mathrm{E}_{3\mathrm{D}}$ to encode all the point clouds in the downstream training dataset $\mathcal{T}$, denoted as $\{PC_n\}_{n=1}^{|\mathcal{T}|}$. Then, we concatenate the training-set features along the sample

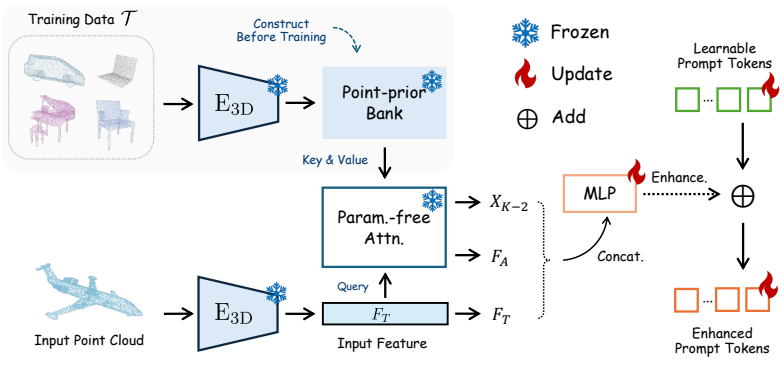

Figure 3. **Point-prior Prompt.** To generate the prompt token with 3D prior knowledge, we construct a point-prior bank before fine-tuning, and conduct parameter-free attention for feature aggregation, which adaptively enhances the learnable prompt token with domain-specific semantics.

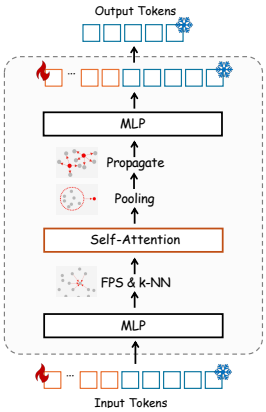

Figure 4. **Geometry-aware Adapter.** Inserted into every transformer block, the adapter aims to extract the fine-grained geometric information by local interactions.

dimension, and store them as the prior knowledge of the downstream 3D domain, formulated as

$$X = \text{Concat}\left(\{E_{3D}(PC_n)\}_{n=1}^{|\mathcal{T}|}\right) \in \mathbb{R}^{|\mathcal{T}|\times D}. \quad (5)$$

Such point-prior bank is constructed in a training-free manner, referring to existing cache-based methods in 2D [22, 35, 36] and 3D [34, 38, 40] vision.

For the input point cloud PC, we also utilize the pre-trained 3D transformer E_{3D} to acquire its 3D feature, denoted as $F_T \in \mathbb{R}^{1\times D}$. Then, we conduct the parameter-free attention for F_T to adaptively aggregate informative semantics from the point-prior bank X. In detail, the input point cloud feature F_T serves as the query, and the pre-encoded training-set features X of the point-prior bank serve as the key and value. Within the attention mechanism, we first calculate the cosine similarity S between the query and key, formulated as

$$S = \frac{F_T X^\top}{|F_T| \cdot |X|} \in \mathbb{R}^{1\times|\mathcal{T}|}. \quad (6)$$

The similarity S denotes the attention scores of the input point cloud to all prior training-set 3D knowledge. Subsequently, we sort the similarity S and obtain the top-$(K-2)$ scores as $S_{K-2} \in \mathbb{R}^{1\times(K-2)}$. Accordingly, we select the corresponding $(K-2)$ training-set features in the value, denoted as $X_{K-2} \in \mathbb{R}^{(K-2)\times D}$. On top of this, we aggregate the prior knowledge in $X_{K-2} \in \mathbb{R}^{(K-2)\times D}$ weighted by $S_{K-2} \in \mathbb{R}^{1\times(K-2)}$ as

$$F_A = \text{Softmax}(S_{K-2})X_{K-2}, \quad (7)$$

where F_A represents the input point cloud feature after aggregating the prior knowledge from the point-prior bank.

After that, we concatenate the original feature F_T with F_A and X_{K-2} to obtain a comprehensive representation of the current point cloud and all its relevant 3D prior semantics, which is then transformed by an MLP with bottleneck layers. We formulate it as

$$P_{prior} = \text{MLP}\big(\text{Concat}(F_T, F_A, X_{K-2})\big), \quad (8)$$

where $P_{prior} \in \mathbb{R}^{K\times D}$ denotes the point prompt adaptively generated by the point-prior bank.

For the i-th transformer block, we acquire the final Point-prior prompt by element-wisely adding P_{prior} with a set of learnable prompt tokens, $R_i \in \mathbb{R}^{K\times D}$, formulated as

$$P_i = R_i + P_{prior}. \quad (9)$$

The former component, R_i denotes the learnable downstream knowledge specific to the i-th block, while the latter adaptively enhances it by the prior domain-specific semantics, contributing to better fine-tuning performance.

3.3. Geometry-aware Adapter

In the i-th transformer block, after being processed by the pre-trained self-attention layer and FFN, the point tokens $C_i' \in \mathbb{R}^{(K+M)\times D}$ are fed into the Geometry-aware Adapter. The adapter aims to grasp the fine-grained geometric information through local aggregation, complementary to the pre-trained global interactions of the self-attention layer.

As shown in Figure 4, the input C_i' is first transformed by an MLP with bottleneck layers, obtaining $T_i \in \mathbb{R}^{(K+M)\times D}$. Then, we adopt farthest point sampling (FPS) to downsample the token number from $(K+M)$ to N, denoted as T_i^c, which serves as a set of local centers. After that, we acquire the neighboring points, T_i^n, for each local center by the k-nearest neighbor (k-NN) algorithm. We formulated the above process as

$$T_i^c = \text{FPS}(T_i) \in \mathbb{R}^{N\times D}, \quad (10)$$

$$T_i^n = k\text{-NN}(T_i^c, T_i) \in \mathbb{R}^{N\times k\times D}. \quad (11)$$

Method	#Param (M)	Acc. (%)
Training from Scratch		
Point-NN	0.0	64.9
PointNet	3.5	68.0
PointNet++	1.5	77.9
PointMLP	14.9	85.2
Point-PN†	0.8	87.1
Self-supervised Pre-training		
Point-BERT	22.1	83.1
w/ Point-PEFT	**0.6**	**85.0**
	↓97.4%	**+1.9**
Point-M2AE	15.3	86.4
Point-M2AE†	15.3	88.1
w/ Point-PEFT†	**0.7**	**88.2**
	↓95.4%	**+0.1**
Point-MAE	22.1	85.2
Point-MAE†	22.1	88.1
w/ Point-PEFT†	**0.7**	**89.1**
	↓96.8%	**+1.0**

Table 1. **Real-world 3D Classification on ScanObjectNN**. We report the number of learnable parameters (#Param) and the accuracy (%) on the "PB-T50-RS" split of ScanObjectNN. † indicates utilizing a stronger data augmentation [37] during fine-tuning.

Method	#Param (M)	Acc. (%)
Training from Scratch		
Point-NN	0.0	81.8
PointNet	3.5	89.2
PointNet++	1.5	90.7
PointCNN	0.6	92.2
DGCNN	1.8	92.9
PCT	2.9	93.2
Point-PN	0.8	93.8
PointMLP	14.9	94.1
Self-supervised Pre-training		
Point-BERT	22.1	92.7
w/ Point-PEFT	**0.6**	**93.4**
	↓97.4%	**+0.7**
Point-M2AE	15.3	93.4
w/ Point-PEFT	**0.6**	**94.1**
	↓96.1%	**+0.7**
Point-MAE	22.1	93.2
w/ Point-PEFT	**0.8**	**94.2**
	↓96.4%	**+1.0**

Table 2. **Synthetic 3D Classification on ModelNet40**. We report the number of learnable parameters (#Param) and the accuracy (%) on ModelNet40. Note that, during the testing process, we do not employ the voting strategy.

To grasp the fine-grained local semantics within each group, T_i^n is fed into a self-attention layer for intra-group interactions. The weights of the self-attention layer are shared across all transformer blocks, which effectively reduces the trainable parameters. We formulate it as

$$T_i^{n\prime} = \text{Self-Attn.}(T_i^n). \tag{12}$$

On top of this, we utilize a pooling operation to integrate the features within each local neighborhood, and conduct a weighted summation between the integrated features and the original local-center features as

$$T_i^{c\prime} = \text{Pooling}(T_i^{n\prime}) + \alpha \cdot T_i^c, \tag{13}$$

where $T_i^{c\prime} \in \mathbb{R}^{N\times D}$ denotes the enhanced local-center features with fine-grained geometries, and α denotes a balance factor. Finally, referring to PointNet++ [19], we propagate $T_i^{c\prime}$ from each local center to its corresponding k neighboring points with a weighted summation as

$$T_i' = \text{Propagate}(T_i^{c\prime}) + \beta \cdot T_i, \tag{14}$$

where $T_i' \in \mathbb{R}^{(K+M)\times D}$, and β denotes a balance factor. After incorporating the fine-grained 3D semantics, T_i' is further processed by an MLP to obtain the output tokens of the i-th block, F_i.

4. Experiments

We evaluate the performance of our proposed Point-PEFT framework for 3D shape classification. We utilize three pre-trained models (Point-BERT [28], Point-MAE [17], and Point-M2AE [32]) as our baselines. Please refer to the Supplementary Material for experiments of part segmentation and additional ablation studies.

4.1. Experimental Settings

4.1.1 ScanObjectNN.

The ScanObjectNN [24] dataset is a real-world 3D point cloud classification dataset, containing about 15,000 3D objects from 15 distinct categories. We focus on the hardest "PB-T50-RS" split, where the rotation (R) and scaling (S) augmentation methods are applied to objects. For all considered models, we employ the AdamW optimizer [16] coupled with a cosine learning rate decay strategy. The initial learning rate is set as 0.0005, with a weight decay factor of 0.05. We fine-tune the models in 300 epochs, utilizing a batch size of 32. As shown in Table 1, † indicates that the fine-tuning utilizes a stronger data augmentation in I2P-MAE [37], including random scaling, translation, and rotation. Otherwise, we only adopt random scaling and translation. Respectively for Point-BERT, Point-MAE, and Point-M2AE, we set the prompting layers and prompt length (L,

Method	#Param (M)	Acc. (%)
Full Fine-Tuning	22.1 M	88.1
Prompt Tuning	0.3 M	83.6
Adapter Tuning	0.4 M	86.7
LoRA	0.4 M	86.3
Bias Tuning	0.3 M	85.0
Point-PEFT	**0.7 M**	**89.1**

Table 3. **Ablation Study on Different PEFT Methods**.

Point-prior Prompt	GA-Adapter	Bias Tuning	Acc. (%)
-	-	-	88.1
✓	-	-	84.7
-	✓	-	87.6
✓	✓	-	88.6
✓	✓	✓	**89.1**

Table 4. **Ablation Study on Main Components.**

Point-prior Bank	Learnable Positions	Acc. (%)
-	-	87.9
✓	-	88.4
✓	✓	**89.1**

Table 5. **Ablation Study on Point-prior Prompt.**

Local Aggregation	Self-Attn.	MLP_{final}	Acc. (%)
-	-	-	87.0
✓	-	-	88.2
✓	✓	-	88.7
✓	✓	✓	**89.1**

Table 6. **Ablation Study on Geometry-aware Adapter.** 'Local Aggregation' refers to the FPS, k-NN, pooling, and propagation operations in the adapter.

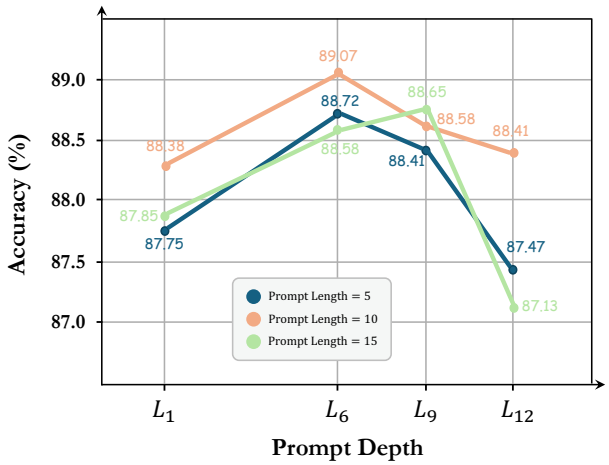

Figure 5. **Ablation Study on Prompt Length and Depth.** The deep blue, light orange, and light green lines represent a prompt length of 5, 10, and 15, respectively.

K) as (6, 5), (6, 10), and (15, 16).

4.1.2 ModelNet40.

The ModelNet40 dataset [25] comprises a total of 12,311 3D CAD models across 40 categories. The point cloud objects are complete and uniform. For experiments on ModelNet40, we adopt the same fine-tuning settings as ScanObjectNN. For all models, we adopt the default data augmentation random scaling and translation. Respectively for Point-BERT, Point-MAE, and Point-M2AE, we set the prompting layers and prompt length (L, K) as (9, 16), (12, 16), and (15, 16). Note that, during the testing process, we do not employ the voting strategy.

4.2. Quantitative Analysis

4.2.1 Performance on ScanObjectNN.

As shown in Table 1, our Point-PEFT framework surpasses the full fine-tuning method with less than 5% trainable parameters. The improvements brought by our framework are +1.9%, +0.1%, and +1.0% for Point-BERT, Point-M2AE†, and Point-MAE† respectively, indicating our great advantages under complex 3D scenes by the extracted fine-grained geometric information. Compared to the full fine-tuning method, our Point-PEFT framework has the stronger ability to be adapted to the tasks related to the real-world scanned objects by pre-trained prior knowledge.

4.2.2 Performance on ModelNet40.

As shown in Table 2, employing our Point-PEFT framework with less than 4% learnable parameters, we achieve performances of 93.4%, 94.1%, and 94.2% for Point-BERT, Point-M2AE, and Point-MAE respectively with the gains of +0.7%, +0.7%, and +1.0%. These results point out the effectiveness of our framework in handling the sparse and irregular point cloud features. For the synthetic point cloud objects, the Point-PEFT framework could grasp the global shape and understand the local 3D structures concurrently.

5. Ablation Study

In this section, we conduct extensive ablation studies to explore the effectiveness of different components within our Point-PEFT framework. We adopt Point-MAE† as the pre-trained model, and report the classification accuracy (%) on the "PB-T50-RS" split of the ScanObjectNN dataset.

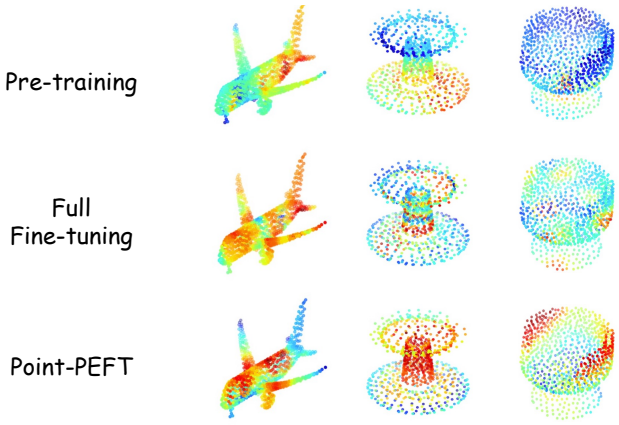

Figure 6. **Visualization of the Captured Fine-grained Information.** We visualize the point feature responses respectively for the pre-trained model, the full fine-tuning method, and our proposed Point-PEFT. **The red color indicates higher responses.**

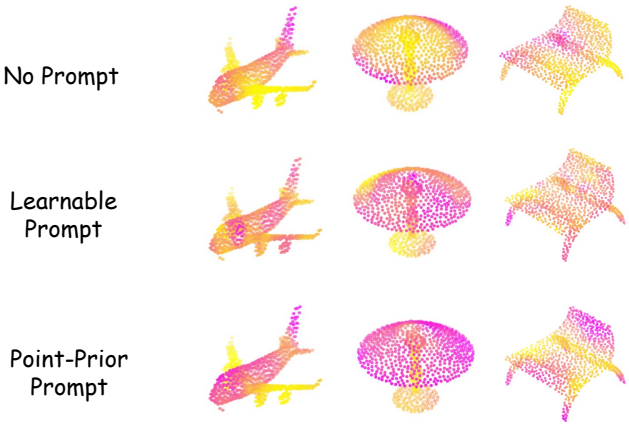

Figure 7. **Visualization of Different Prompt Tuning Methods.** For the three rows, we respectively visualize the attention scores of the **[CLS]** token, the learnable prompt token, and the Point-prior prompt token to other point cloud tokens. **The pink color indicates higher values.**

5.1. Comparison to Traditional PEFT Methods.

As shown in Table 3, our Point-PEFT framework can surpass conventional PEFT techniques with huge gains, e.g., +5.5% over Prompt Tuning, +2.4% over Adapters, +2.8% over Low-Rank Adaptation (LoRA), and +4.1% over Bias Tuning. The comprehensive experiments have exhibited the superiority of our framework over the traditional PEFT methods, indicating that our proposed method effectively integrates 3D domain-specific knowledge into the PEFT framework. In contrast to the PEFT techniques in language and 2D image domains, our framework focuses more on the complex and irregular point cloud structures, specifically designed for 3D.

5.2. Effectiveness of Main Components.

As shown in Table 4, to substantiate the effectiveness of each component, we conduct the ablation experiments by incrementally introducing components to the baseline (Point-MAE† model) until the complete Point-PEFT framework. The light gray row indicates the baseline with the transformer encoder, and the dark gray row represents the complete structure of the Point-PEFT framework. Introducing either the Point-prior Prompt or the Geometry-aware Adapter component leads to a certain degree of performance degradation compared to full fine-tuning. When both components are utilized, the performance has been improved to 88.6% with a gain of +0.5%. Further adding Bias Tuning can result in an additional +0.5% improvement. Therefore, the experiments indicated the effectiveness of each design within the Point-PEFT framework.

5.3. Effects of Prompt Length and Depth.

In Figure 5, we ablate the number of earlier layers applying the prompt tokens ('Prompt Depth') and the 'Prompt Length'. As shown, longer prompt tokens don't necessarily lead to better performance. For different prompt depths, the length of 10 almost yields the best results, indicating that a moderate length is the most appropriate for 3D prompt learning. In addition, the optimal depth varies with different prompt token lengths. Notably, the insertion of prompt tokens in all blocks fails to bring significant performance improvement. Introducing them in a certain number of earlier blocks is the preferable strategy, suggesting that prompt tokens in earlier layers carry more significance than those in later ones.

5.4. Components of Point-prior Prompt.

As shown in Table 5, we investigate the effects of the point-prior bank and the learnable coordinates for the Point-prior Prompt. The first row in light gray represents that we only utilize the learnable prompt tokens, which are randomly initialized before training. The last row in dark gray shows the complete structure of the Point-prior Prompt. By constructing the point-prior bank with the 3D domain-specific semantics to enhance the prompt tokens, our method achieves a performance gain of +0.5%, indicating the importance of prior knowledge enhancement. The assignment of the shared learnable coordinates boosts +0.7% in performance, which represents the spatial locations of the prompt tokens. The coordinates enable the prompt tokens to engage in the local interactions of the Geometry-aware Adapter alongside the features. The experiments confirm the effectiveness of each component in the Point-prior Prompt to leverage the prior 3D semantics for downstream tasks.

5.5. Components of Geometry-aware Adapter.

In Table 6, we conduct ablation studies by incrementally introducing components of our Geometry-aware Adapter. The row in light gray signifies the baseline adapter, consisting of only an MLP with bottleneck layers. The row in dark gray indicates the complete structure of the Geometry-aware Adapter. By integrating the local-aggregation operations, comprised of FPS, k-NN, pooling, and propagation operations, our approach achieves a significant performance improvement of +1.2%. These operations effectively extract the 3D fine-grained structures of local neighborhoods. The self-attention layer brings an additional performance gain of +0.5%, boosting the perception of fine-grained geometric information through the intra-group feature interactions. Lastly, by introducing the final bottleneck-layer MLP to further process point cloud features, the performance is improved by +0.4%. The experiments fully demonstrate the effectiveness of each component in our Geometry-aware Adapter to aggregate the local geometric information, which is complementary to the global attention in pre-trained 3D models.

6. Visualization

6.1. Fine-grained Geometric Information.

The Geometry-aware Adapter captures local geometric knowledge through the interactions within local regions. In Figure 6, we visualize the feature responses based on Point-MAE. In each row, we show the responses for the pre-trained model, the full fine-tuning method and our Point-PEFT respectively, where warm colors indicate the high responses. As shown, compared with others' randomly focusing on the less significant parts or concentrating consistently on the whole object, our Point-PEFT more focuses on the discriminative parts of the objects, such as the wings and engines of airplanes, the brackets and shades of lamps, which are critical for distinguishing similar 3D shapes. This indicates that our Point-PEFT with the Geometry-aware Adapter not only emphasizes global information but also boosts the understanding of the fine-grained crucial structures.

6.2. Different Prompt Tuning Methods.

The Point-prior Prompt utilizes 3D domain-specific knowledge from the point-prior bank to enhance the prompt tokens. In Figure 7, we respectively visualize the attention scores of the [CLS] token, the learnable prompt token, and the Point-prior prompt token to other point cloud tokens, where the pink color indicates higher values. As illustrated, the [CLS] token grasps useless information, and the learnable prompt tokens fail to capture the crucial 3D semantics. Compared with them, our proposed Point-prior Prompt tokens focus more on the salient and important object parts, such as the fuselages and tails of airplanes, the entire shades of lamps, and the backrests and legs of chairs, which indicates that the Point-prior Prompt with prior pre-trained semantics effectively grasps the critical information and further benefits 3D point cloud understanding.

7. Conclusion

In this paper, we introduce Point-PEFT, a parameter-efficient fine-tuning framework specialized for pre-trained 3D models. Our approach achieves comparable performance to full fine-tuning on downstream tasks, while significantly reducing the number of learnable parameters. The framework consists of a Geometry-aware Adapter and a Point-prior Prompt. The Geometry-aware Adapter leverages local interactions to extract fine-grained geometric information. The Point-prior Prompt utilizes pre-trained semantic information to enhance the prompt tokens. Extensive experiments validate the effectiveness of Point-PEFT. We expect Point-PEFT can serve as a baseline for future 3D PEFT research.

A. Implementation Details

The inner dimension of the bottleneck-shaped MLP in the Point-prior Prompt is set as 8 for all pre-trained models on two datasets. The inner dimension of the first bottleneck-shaped MLP in the Geometry-aware Adapter is decided as 16 for all pre-tarined mdoels. We also set different N and k for the k-NN algorithm at different pre-trained models. The N and k are set as (32, 16), (32, 8), and (64, 16) for Point-BERT [28], Point-M2AE [32], and Point-MAE [17] respectively. On the ScanObjectNN [24] and ModelNet40 [25], we set the inner dimension of the last bottleneck-shaped MLP in the Adapter as (8, 16, 16) and (16,16, 16) for Point-BERT, Point-M2AE, and Point-MAE respectively.

B. Additional Experiments

Shape Classification. Due to the page limitation, we only provide part of the results of Point-PEFT for 3D shape classification in the main paper. In Table 7, we report more quantitative results of Point-PEFT with the stronger data augmentation (random scaling, translation and rotation) and default augmentation method (random scaling and translation), which are denoted as with and without † respectively. As reported, under both of the two data augmentation settings, our Point-PEFT outperforms the full fine-tuning method based on almost all pre-trained models. This indicates the robustness of Point-PEFT on different pre-trained 3D point cloud models and data augmentation methods.

Part Segmentation. To evaluate the understanding capacity of Point-PEFT on dense 3D data, we further conduct

Methods	#Param(M)	Acc.(%)
Training from Scratch		
Point-NN	0.0	64.9
PointNet	3.5	68.0
PointNet++	1.5	77.9
PointMLP	14.9	85.2
Point-PN†	0.8	87.1
Self-supervised Pre-training		
Point-BERT	22.1	83.1
w/ Point-PEFT	**0.6**	**85.0**
	↓97.4%	**+1.9**
Point-M2AE	15.3	86.4
w/ Point-PEFT	**0.7**	**86.4**
	↓95.4%	**+0.0**
Point-M2AE†	15.3	88.1
w/ Point-PEFT†	**0.7**	**88.2**
	↓95.4%	**+0.1**
Point-MAE	22.1	85.2
w/ Point-PEFT	**0.7**	**85.5**
	↓96.8%	**+0.3**
Point-MAE†	22.1	88.1
w/ Point-PEFT†	**0.7**	**89.1**
	↓96.8%	**+1.0**

Table 7. **Performance of Shape Classification on ScanObjectNN Dataset.** We report the number of learnable parameters (#Param) and the accuracy (%) on the "PB-T50-RS" split of ScanObjectNN [24]. † indicates utilizing a stronger data augmentation [37] during fine-tuning.

Method	#Param(M)	mIoU$_C$(%)	mIoU$_I$(%)
Training from Scratch			
PointNet	-	80.39	83.7
PointNet++	-	81.85	85.1
DGCNN	-	82.33	85.2
Self-supervised Pre-training			
Transformer	27.09	83.42	85.1
MaskPoint	-	84.60	86.0
Point-MAE‡	22.26	84.19	86.1
w/ Point-PEFT	**0.45**	83.81	85.3
Point-M2AE‡	13.03	**84.86**	**86.5**
w/ Point-PEFT	**0.39**	84.35	86.1

Table 8. **Part Segmentation on ShapeNetPart [27].** 'mIoU$_C$' (%) and 'mIoU$_I$' (%) denote the mean IoU across all part categories and all instances in the dataset, respectively. ‡ denotes that we do not include classification heads into the parameter calculation.

part segmentation experiments on ShapeNetPart [27] that contains 14,007 and 2,874 samples for training and validation, respectively. We follow previous works to adopt the same segmentation head and sample 2048 points for each object as input. For each pre-trained model, we fine-tune Point-PEFT for 300 epochs with a batch size of 16, and adopt the learning rate as 2×10^{-4}. In Table 8, we report the mean IoU across all part categories and instances, denoted as mIoU$_C$ and mIoU$_I$, respectively. As reported, on each pre-trained model, our Point-PEFT attains the competitive performance on both metrics, which illustrates the strong understanding capacity of our designs on the fine-grained dense data. Compared to the full fine-tuning method, the number of learnable parameters has dropped **98%** and **97%** for the Point-MAE and Point-M2AE respectively.

Positions of GA-Adapter.				Insertion Depth	Acc.(%)
After	Before	Both	Parallel		
✓	-	-	-	6	88.0%
✓	-	-	-	12	**89.1%**
-	✓	-	-	6	88.3%
-	✓	-	-	12	88.0%
-	-	✓	-	6	86.0%
-	-	✓	-	12	86.1%
-	-	-	✓	6	87.9%
-	-	-	✓	12	88.4%

Table 9. **Ablation Study on Adapter Positions and Depth.** The 'After', 'Before', 'Both', 'Parallel' denote block structure of putting the "Geometry-aware Adapter" after, before, wrapping around, and in parallel to the "FFN" layer respectively. The insertion depth denotes the certain number of earlier blocks.

C. Additional Ablation Study

The Adapter Positions and Depth We further conduct ablation study on the positions and depth of the proposed Geometry-aware Adapter in Table 9, where we utilize the final version of Point-PEFT as the baseline and compare the accuracy(%) on the "PB-T50-RS" split of the ScanObjectNN with Point-MAE† as the pre-trained model. As shown in Table 9, the last position within the block and the depth of 12 perform the best. That is because, compared to other positions, after the global interactions among point cloud features through pre-trained self-attention layer and Feed-forward Networks(FFN), the local interactions within the adapter could further boost the extraction of the fine-grained geometric information, which facilitates the fine-tuning process. And the deeper insertion could utilize the high-level knowledge to grasp the better understanding of 3D structures. Note that, for all pre-trained point cloud models, we insert the Geometry-aware Adapter in the last position across all blocks.

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. 2

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1

[3] Anthony Chen, Kevin Zhang, Renrui Zhang, Zihan Wang, Yuheng Lu, Yandong Guo, and Shanghang Zhang. Pimae: Point cloud and image interactive masked autoencoders for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5291–5301, 2023. 2

[4] 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? *arXiv preprint arXiv:2212.08320*, 2022. 2

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 1

[6] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021. 2

[7] Ziyu Guo, Xianzhi Li, and Pheng Ann Heng. Joint-mae: 2d-3d joint masked autoencoders for 3d point cloud pre-training. *arXiv preprint arXiv:2302.14007*, 2023. 2

[8] Ziyu Guo, Yiwen Tang, Renrui Zhang, Dong Wang, Zhigang Wang, Bin Zhao, and Xuelong Li. Viewrefer: Grasp the multi-view knowledge for 3d visual grounding with gpt and prototype guidance. *arXiv preprint arXiv:2303.16894*, 2023. 2

[9] Ziyu Guo, Renrui Zhang, Xiangyang Zhu, Yiwen Tang, Xianzheng Ma, Jiaming Han, Kexin Chen, Peng Gao, Xianzhi Li, Hongsheng Li, and Pheng-Ann Heng. Point-bind & point-llm: Aligning point cloud with multi-modality for 3d understanding, generation, and instruction following. *arXiv preprint arXiv:2309.00615*, 2023. 2

[10] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021. 2

[11] 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. 2

[12] 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. 1, 2

[13] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 1, 2

[14] 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. 1, 2

[15] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021. 1, 2

[16] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[17] 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, 5, 8

[18] 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. 2, 3

[19] Charles Ruizhongtai 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. 2, 5

[20] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 1

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2

[22] Jintao Rong, Hao Chen, Tianxiao Chen, Linlin Ou, Xinyi Yu, and Yifan Liu. Retrieval-enhanced visual prompt learning for few-shot classification. *arXiv preprint arXiv:2306.02243*, 2023. 4

[23] 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

[24] 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. 2, 5, 8, 9

[25] 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. 1, 2, 6, 8

[26] 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. 2

[27] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016. 2, 9

[28] 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, 5, 8

[29] Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv preprint arXiv:2106.10199*, 2021. 2, 3

[30] 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. *arXiv preprint arXiv:2304.07221*, 2023. 3

[31] Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. Side-tuning: a baseline for network adaptation via additive side networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 698–714. Springer, 2020. 1, 2

[32] 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, 2, 5, 8

[33] Renrui Zhang, Ziyu Guo, Wei Zhang, Kunchang Li, Xupeng Miao, Bin Cui, Yu Qiao, Peng Gao, and Hongsheng Li. Pointclip: Point cloud understanding by clip. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8552–8562, 2022. 2

[34] Renrui Zhang, Liuhui Wang, Ziyu Guo, and Jianbo Shi. Nearest neighbors meet deep neural networks for point cloud analysis. In *WACV 2023*, 2022. 4

[35] Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free adaption of clip for few-shot classification. In *ECCV 2022*. Springer Nature Switzerland, 2022. 4

[36] Renrui Zhang, Xiangfei Hu, Bohao Li, Siyuan Huang, Hanqiu Deng, Hongsheng Li, Yu Qiao, and Peng Gao. Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners. *CVPR 2023*, 2023. 4

[37] 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. 1, 2, 5, 9

[38] Renrui Zhang, Liuhui Wang, Yali Wang, Peng Gao, Hongsheng Li, and Jianbo Shi. Parameter is not all you need: Starting from non-parametric networks for 3d point cloud analysis. *CVPR 2023*, 2023. 4

[39] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyao Zeng, Shanghang Zhang, and Peng Gao. Pointclip v2: Adapting clip for powerful 3d open-world learning. *ICCV 2023*, 2022. 2

[40] Xiangyang Zhu, Renrui Zhang, Bowei He, Ziyu Guo, Jiaming Liu, Hao Dong, and Peng Gao. Less is more: Towards efficient few-shot 3d semantic segmentation via training-free networks. *arXiv preprint arXiv:2308.12961*, 2023. 4