2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

# Fine-Tuning Point Cloud Transformers with Dynamic Aggregation

Jiajun Fei, Zhidong Deng
Institute for Artificial Intelligence at Tsinghua University (THUAI), State Key Laboratory of Intelligent Technology and Systems, Beijing National Research Center for Information Science and Technology (BNRist), Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
Email: {feijj20@mails,michael@mail}.tsinghua.edu.cn

***Abstract*— Point clouds play an important role in 3D analysis, which has broad applications in robotics and autonomous driving. The pre-training fine-tuning paradigm has shown great potential in the point cloud domain. Full fine-tuning is generally effective but leads to a heavy storage and computational burden, which becomes inefficient and unacceptable as the size of pre-trained models scales. Although efficient fine-tuning approaches have significant progress in other domains, they generally perform worse for point clouds. To overcome this dilemma, we revisit the official Point-MAE implementation and find the critical role of aggregation in fine-tuning performances. Inspired by such discoveries, we propose a novel dynamic aggregation (DA) method to replace previous static aggregation like mean or max pooling for pre-trained point cloud Transformers. Besides standard metrics such as accuracy or mIoU, we evaluate the number of tunable parameters and additional FLOPs for a fair comparison of our method to different fine-tuning approaches. We construct several DA variants and validate them through extensive experiments. Experimental results demonstrate that DA has competitive performances against full fine-tuning and other efficient fine-tuning approaches. The code is publicly available at https://github.com/JaronTHU/DynamicAggregation.**

## I. INTRODUCTION

Point cloud processing is an important research topic in the 3D domain, with broad applications in robotics and autonomous driving. Ever since PointNet [1] was invented, various deep learning models have been proposed to process point clouds directly [2], [3], [4], [5]. As the amount of point cloud data and the size of networks grow, the pre-training fine-tuning paradigm attracts great research interest, where a single pre-trained model can be fine-tuned on many downstream tasks and has competitive performances compared to task-specific models [6], [7], [8], [9], [10], [11]. However, full fine-tuning, a commonly-used tuning approach, learns different sets of parameters for various tasks. With the wide spread of Transformers [12] and the continuous growth of the pre-trained model size, full fine-tuning leads to severe storage inefficiency.

To address this limitation, efficient fine-tuning approaches attract broad research interest. Head-only fine-tuning, a representative method, is only tuning task-specific heads while keeping all pre-trained parameters frozen [13], [14], [15]. It is widely employed as a probe to validate the effectiveness of pre-training. However, the performance gaps between head-only and full fine-tuning are always significant, making the head-only fine-tuned models practically useless in downstream tasks. Some methods [16], [17] add bias terms into tunable parameters, achieving the balance between parameter requirements and performances. Adapters, as inserted learnable modules, are first proposed to fine-tune large language models efficiently [18], [19], [20], [21], and then employed in fine-tuning vision Transformers [22], [23], [24]. Recently, prompt tuning has been a hot research topic in both Natural Language Processing (NLP) and Computer Vision (CV) [25], [26], [27], [28], [29], [30]. It transforms inputs to suit pre-trained models instead of tuning them directly. Recently, though pre-trained Transformers [9], [10], [11] are investigated in the point cloud domain, efficient fine-tuning approaches are yet to be explored for them. Zha et al. [31] try to apply prompt tuning on pre-trained point cloud Transformers. They find static prompts unsuitable for point cloud models and turn to dynamic instance-aware prompts. Although their method only introduces a small number of parameters while having similar or superior performances, the dynamic graph convolutions lead to a heavy computational burden, making the inference process more inefficient compared to full fine-tuning.

In this paper, we propose a novel dynamic aggregation (DA) method to fine-tune pre-trained point cloud Transformers efficiently. Compared to previous approaches, DA has two main advantages. First, DA is more suitable for Transformers trained with point cloud data. Zha et al. [31] show that many well-performed approaches cannot mitigate the domain gap among point cloud datasets. Experimental results demonstrate that DA can close this gap by dynamically aggregating patch sequences. Second, DA is highly flexible. We construct and evaluate three DA variants of different sizes, i.e., DA-Naive, DA-Light, and DA-Heavy. In most experiments, DA-Light achieves similar performances to full fine-tuning using negligible additional parameters and FLOPs, and DA-Heavy matches the state-of-the-art (SOTA) methods.

The main contributions are summarized as follows.

- The important role of aggregation in fine-tuning point cloud Transformers is identified by revisiting the official implementation of Point-MAE [10] as well as the classification token proposed in ViT [32].
- Inspired by the above discovery, we propose a dynamic aggregation (DA) strategy to fine-tune point cloud Transformers efficiently. Besides the task-specific head, DA only needs to adjust the dynamic aggregation module.
- As a general module, DA shows competitive perfor-

mances on a variety of tasks and experiments. DA-Light gives comparable results with negligible additional parameters and FLOPs, while DA-Heavy generally performs better using more computational resources.

## II. RELATED WORK

### A. Point Cloud Pre-training

With the growth of dataset sizes and model capacities, large pre-trained models have shown excellent capabilities in NLP [12], [33], [34], [35] and CV [36], [37], [38], [39]. Inspired by the achievements in texts and images, researchers also investigate point cloud pre-training. Early works [13], [14], [40], [15], [41] mainly focus on generative pre-training tasks, where small pre-trained models (encoders) are employed on the ModelNet [42] classification and other simple downstream tasks. Later, many methods apply contrastive learning on point cloud pre-training. Info3D [6] improves the learned representation by maximizing the mutual information between 3D shapes and their variants. PointContrast [7] pre-trains the SR-UNet using the PointInfoNCE loss on ScanNet [43] and successfully transfers the learned representation to high-level semantic understanding tasks. Hou et al. [8] extend point-level matching in [7] to patch-level ones and get good performances on complex tasks. Motivated by successful Transformers in other domains [34], [39], researchers construct similar models in point cloud pre-training. Point-BERT [9] proposes a masked point modeling task to pre-train Transformers. Point-MAE [10] designs another efficient pre-training scheme entirely based on standard Transformers [12]. Point-M2AE [11] further applies multi-scale masked autoencoders to capture low-level and high-level features. A recent work, named ReCon [44], analyzes characteristics of contrastive learning and reconstruction learning, and unifies them to share their merits in point cloud pre-training. Besides, a series of works [45], [46], [47], [48], [49] explore effective pre-training methods for real-world point clouds and achieve significant improvements in robotics and autonomous driving.

### B. Efficient Fine-Tuning

Transfer learning is an important research topic aimed at re-using knowledge learned from one task to others. Full fine-tuning, a widely-used transfer learning technique, tunes all pre-trained parameters according to the requirements of downstream tasks [34], [37], [38], [39]. Since pre-trained parameters are only used for initialization, full fine-tuning has to learn many sets of parameters for different tasks and faces severe problems of parameter inefficiency. Many efficient fine-tuning approaches have been proposed to address this dilemma.

*Partial Fine-Tuning:* Partial fine-tuning approaches only tune part of the pre-trained models. A popular protocol is freezing all pre-trained parameters and only tuning task-specific ones, as adopted in many works [13], [14], [15]. Besides, some works [50], [51], [52], [39] fine-tune the last several layers of pre-trained models while freezing the others. SpotTune [53] employs an additional policy network to decide which layers to be fine-tuned for various inputs. TinyTL [16] and BitFit [17] show that fine-tuning the bias terms is competitive with low memory cost. LoRA [54] trains trainable rank decomposition matrices for modest storage requirements and low additional inference latency. Touvron et al. [55] fine-tune the multi-head attention layers of ViT [32] for deployment.

*Adapter Tuning:* Adapters are tunable layers added between fixed layers of pre-trained models. They have made significant progress in NLP as a parameter-efficient fine-tuning technique [18], [19], [20], [21]. In CV, Rebuffi et al. [22] proposes residual adapters to keep the majority of domain-agnostic parameters and learns the minority of domain-specific parameters. AdaptFormer [23] adds additional trainable bottleneck modules to feed-forward networks in ViT [32], and efficiently transfers pre-trained knowledge to both image and video downstream tasks. SSF [24] inserts trainable scale & shift modules to modulate features between fixed layers. PointCLIP [56] proposes an inter-view adapter to transfer CLIP's [57] 2D knowledge to 3D few-shot learning.

*Prompt Tuning:* Prompt tuning, also first proposed in NLP, deploys pre-trained models through reformulating downstream tasks into pre-trained ones [25], [26], [27], [58], [59]. Inspired from these advances, some works introduce prompts to the CV domain. VPT [28] freezes the entire pre-trained model and prepends learnable prompts to layer inputs during downstream training. Bahng et al. [29] propose visual prompting that directly perturbs input images for CLIP [57] and vision models. P2P [60] employs the Point-to-Pixel prompting to tune pre-trained image models for point cloud analysis. For high-diversity datasets, DAM-VP [30] first clusters downstream data into several subsets and then learns prompts separately. IDPT [31] shows that static prompts are vulnerable to real-world point clouds and utilize dynamic prompts on pre-trained point cloud models. Besides, NOAH [61] subsumes Adapter [18], LoRA [54], and VPT [28] and demonstrates its superiority on many vision tasks.

Efficient fine-tuning has attracted broad research interest in texts and images. However, it is yet to be explored for point clouds. Some existing methods focus on applying large language and 2D vision models on point clouds [56], [60], while only a few are aimed at tuning pre-trained point cloud models [31].

## III. METHODOLOGY

In this section, we first revisit the classification tokens in vision Transformers. Then, we propose our DA method with experimental results and theoretical analysis.

### A. Static Aggregation

The classification token ([CLS], $\boldsymbol{c}_0 \in \mathbb{R}^C$) is first introduced in BERT [34], corresponding to which the final hidden state $\boldsymbol{c}_D$ is used as the aggregate features for classification. In the field of CV, ViT [32] also adopts this design by prepending a learnable embedding to the input sequence. Suppose that the input sequence is $\boldsymbol{x}_{0j}\ j = 1, \cdots, N$,

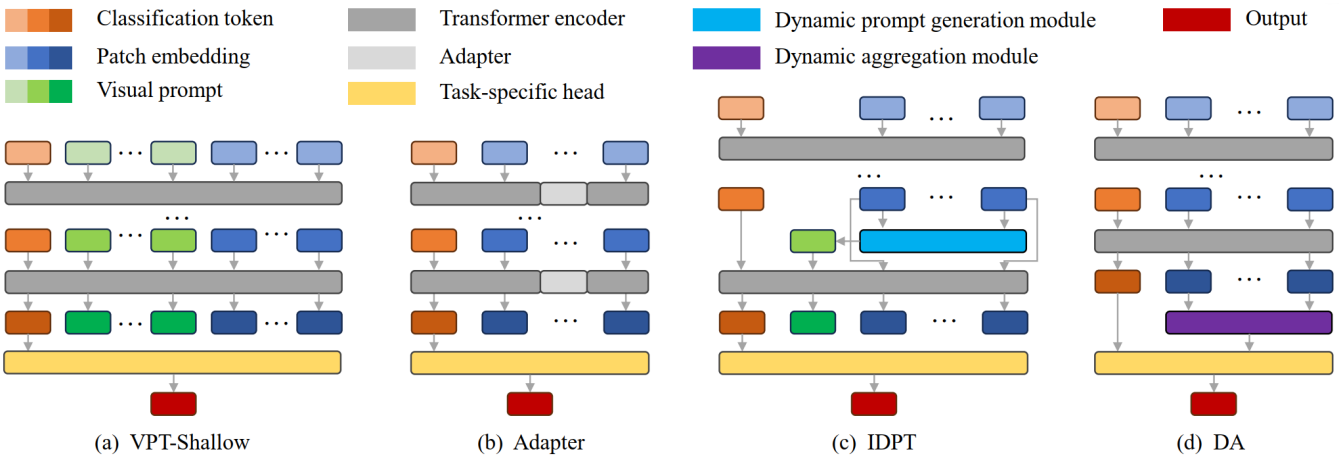


Fig. 1. A comparison of DA with other representative efficient fine-tuning approaches. Compared to VPT, DA does not introduce additional learnable tokens or increase the computational burden. Compared to Adapter and IDPT [31], DA does not alter the internal structure of the Transformer backbone. Best viewed in color.

the output sequences for the $i$-th encoder layer $f_i$ are $\boldsymbol{c}_i, \boldsymbol{x}_{ij}$ $i = 1, \cdots, D$, and the classification head is $h$. Then the prediction logits $\boldsymbol{y}$ are computed as

$$\boldsymbol{c}_i, \{\boldsymbol{x}_{ij}\}_{j=1}^{N} = f_i\left(\boldsymbol{c}_{i-1}, \{\boldsymbol{x}_{i-1,j}\}_{j=1}^{N}\right), \ i = 1, \cdots, D \quad (1)$$

$$\boldsymbol{y} = h\left(\boldsymbol{c}_D\right). \quad (2)$$

However, this design only partially fits point clouds. We notice a slight modification to Equ. 2 is made in the official implementation of Point-MAE [10]. Besides the final hidden state $\boldsymbol{c}_D$, they also use the max pooling on $\{\boldsymbol{x}_{iD}\}_{i=1}^{N}$ as the aggregate features, as shown in Equ. 3.

$$\boldsymbol{y} = h\left(\boldsymbol{c}_D, \mathrm{MAX}\left(\{\boldsymbol{x}_{Dj}\}_{j=1}^{N}\right)\right). \quad (3)$$

To find out whether the aggregation influences the fine-tuning performances on downstream tasks, we utilize the pre-trained backbone from Point-MAE [10] and conduct a classification experiment on the PB_T50_RS variant of ScanObjectNN [62] dataset. We evaluate three representative fine-tuning approaches, i.e., head-only fine-tuning (Head), bias fine-tuning (Bias) [16], [17], and full fine-tuning (Full). Three aggregation modes, i.e., the classification token ($\boldsymbol{c}_D$), the mean pooling (MEAN $\left(\{\boldsymbol{x}_{Dj}\}_{j=1}^{N}\right)$), and the max pooling (MAX $\left(\{\boldsymbol{x}_{Dj}\}_{j=1}^{N}\right)$) are tested.

According to the experimental results in Tab. I, we can draw several important conclusions. First, there are significant performance gaps between efficient fine-tuning (Head and Bias) and full fine-tuning on point clouds regardless of aggregation modes. Second, it is insufficient to yield the best fine-tuning performance with only the classification token as the aggregate feature. Last, the accuracy range decreases as the number of tunable parameters increases. Therefore, the aggregation modes do influence fine-tuning performances, especially for efficient fine-tuning approaches (Head and Bias). Since the static aggregation (mean/max pooling) cannot further improve the performances of efficient fine-tuning methods, we turn to dynamic aggregation with a small number of tunable parameters.

TABLE I

FINE-TUNING PERFORMANCES ON SCANOBJECTNN [62] PB_T50_RS WITH DIFFERENT AGGREGATION MODES AND FINE-TUNING APPROACHES. ALL MODELS ARE FINE-TUNED WITH RANDOM ROTATION AUGMENTATION. RANGE MEANS THE DIFFERENCE BETWEEN THE MAXIMUM AND THE MINIMUM IN ONE COLUMN.

| Aggregation | | | Accuracy (%) | | |
|---|---|---|---|---|---|
| CLS | MEAN | MAX | Head | Bias [16], [17] | Full |
| ✓ | × | × | 79.63 | 84.52 | 88.03 |
| × | ✓ | × | 82.03 | 84.94 | 87.65 |
| × | × | ✓ | 77.52 | 84.11 | 88.06 |
| ✓ | ✓ | × | 83.17 | 85.50 | 87.99 |
| ✓ | × | ✓ | 79.91 | 84.35 | 88.51 |
| × | ✓ | ✓ | 82.34 | 84.56 | 88.62 |
| ✓ | ✓ | ✓ | 82.34 | 85.08 | 88.24 |
| Range | | | 5.65 | 1.39 | 0.97 |

## B. Dynamic Aggregation

Mathematically, the dynamic aggregation module (DAM) $\Omega$ has the following form:

$$\boldsymbol{z} = \Omega\left(\{\boldsymbol{x}_j\}_{j=1}^{N}; \boldsymbol{\theta}\right), \quad (4)$$

where $\boldsymbol{z}$ is the aggregate feature, $\{\boldsymbol{x}_j\}_{j=1}^{N}$ is the embedding sequence/set, and $\boldsymbol{\theta}$ represents tunable parameters of $\Omega$. In fact, after combining all encoder layers in Equ. 1, the final hidden state $\boldsymbol{c}_D$ in head-only fine-tuning can be rewritten as the aggregate feature from a DAM parameterized by the classification token $\boldsymbol{c}_0$, as Equ. 5 shows.

$$\boldsymbol{c}_D = \Omega\left(\{\boldsymbol{x}_{0j}\}_{j=1}^{N}; \boldsymbol{c}_0\right). \quad (5)$$

Using similar procedures, the final hidden states $\boldsymbol{c}_D$ in other efficient fine-tuning approaches all share the form of Equ. 4.

Despite the theoretical equivalence, our DA implementation has two main differences from previous efficient fine-tuning methods, as Fig. 1 shows. First, the internal

structures of Transformers are not altered like Adapters [18], [19], [20] and IDPT [31]. When applying our DA to fine-tune Transformers, the pre-trained Transformer can be completely treated as a black-box model. Second, the input sequence is not extended like VPT [28]. Suppose the original input sequence has $N$ embeddings, then a normal self-attention operation leads to $\mathcal{O}\left(N^2\right)$ complexity. With prepended $k$ more prompts, the additional complexity is $\mathcal{O}\left(\left(N+k\right)^2 - N^2\right) = \mathcal{O}\left(k^2 + 2Nk\right)$, exhibiting quadratic growth to the number of prompts. Nevertheless, there is no such additional computational cost using our DA. If the DAM consists of too many parameters or introduces too high a computational burden, then DA becomes less meaningful than full fine-tuning. Therefore, we require the DAM to be either parameter-efficient or computation-efficient. In this paper, we investigate the following DA variants.

*1) DA-Naive:* As a direct generalization of static aggregation, we insert a multi-layer perceptron (MLP) between the last Transformer encoder layer and mean/max pooling. However, DA-Naive hardly gives promising results, especially in the classification tasks.

*2) DA-Light:* DuMLP-Pin [63] is a light permutation-invariant network designed for set data. Using dual MLPs to aggregate global features, it introduces negligible parameters and FLOPs but has excellent performances on various tasks. However, DA-Light still lags behind full fine-tuning sometimes.

*3) DA-Heavy:* As illustrated in Fig. 1, though the dynamic prompt generation module in IDPT [31] and our DAM are inserted at different positions, they both process embedding sequences. Inspired by its success, we move the DGCNN [3] in IDPT [31] after the last layer as the DAM. Compared to DA-Light, DA-Heavy introduces more additional parameters and incurs a heavier computational burden, but its performance is generally outstanding.

## IV. EXPERIMENTS

### A. Experimental Settings

We use pre-trained point cloud Transformers from Point-MAE [10] as backbones and completely adopt their fine-tuning experimental settings. All experiments are done on a GeForce RTX 3090 GPU. We compare our DA with the following approaches in tuning point cloud Transformers.

*1) Training from scratch (Scratch):* It is meaningless to utilize pre-trained models if an efficient fine-tuning method cannot perform better than training from scratch. We use it to justify whether an approach is meaningful on point cloud data.

*2) Full fine-tuning (Full):* All pre-trained and task-specific parameters are tuned.

*3) Head-only fine-tuning (Head):* The backbone is fixed, and only the task-specific head is tuned.

*4) Bias fine-tuning (Bias) [16], [17]:* Besides the task-specific head, all bias terms in the Transformer are tuned.

*5) VPT [28]:* Besides the task-specific head, prepended visual prompts are tuned.

*6) Adapter [23], [24]:* Besides the task-specific head, inserted adapters are tuned.

*7) IDPT [31]:* Besides the task-specific head, dynamic prompt generation modules are tuned.

We apply grid search to find the optimal hyper-parameters for methods with tunable hyper-parameters, i.e., VPT [28], Adapter [23], [24], and DA. Besides standard metrics like accuracy or IoU, we also evaluate the number of tunable parameters (#TP) and FLOPs, which reflect how efficient these fine-tuning approaches are.

### B. Synthetic Point Cloud Classification

As a pioneering 3D shape dataset, the ModelNet40 [64] has 9,843 CAD models in the training set and 2,468 ones in the test set. We use 1,024 points without normal vectors from the sampled point cloud of [1]. We take the overall accuracy (OA) as the evaluation metric. For fair comparisons, we do not employ the voting strategy for evaluation.

TABLE II

CLASSIFICATION RESULTS ON THE MODELNET40 [64] DATASET. NO VOTING IS USED. * THE FLOPS OF VPT-SHALLOW ARE LARGER THAN THOSE OF VPT-DEEP, BECAUSE THEIR OPTIMAL NUMBER OF PROMPTS ARE DIFFERENT.

| Method | #TP(M) | FLOPs(G) | OA(%) |
|---|---|---|---|
| Scratch | 22.10 | 2.44 | 92.22 |
| Full | 22.10 | | 93.03 |
| Head | 0.28 | | 92.54 |
| Bias [16], [17] | 0.31 | | 93.19 |
| VPT-Shallow* [28] | 0.29 (+0.02) | 2.89 (+0.45) | 92.54 |
| VPT-Deep* [28] | 0.33 (+0.05) | 2.66 (+0.22) | 92.59 |
| Adapter [23], [24] | 3.84 (+3.56) | 2.67 (+0.23) | 93.15 |
| IDPT [31] | 1.71 (+1.43) | 3.61 (+1.17) | 93.31 |
| DA-Light | 0.46 (+0.18) | 2.44 (+0.00) | 93.19 |
| DA-Heavy | 2.15 (+1.87) | 3.41 (+0.97) | **93.44** |

According to the experimental results in Tab. II, Bias [16], [17] and DA-Light slightly outperform full fine-tuning using only $1 \sim 2\%$ tunable parameters. Despite more parameters and sophisticated structures, we do not observe noticeable improvements in IDPT [31], Adapter [23], [24], and DA-heavy. One possible reason is that the performance gap between training from scratch and full fine-tuning is insignificant, which means that pre-training may not be helpful in this task. To this end, we turn to challenging real-world datasets and complex downstream tasks to evaluate fine-tuning methods.

### C. Real-World Point Cloud Classification

The ScanObjectNN [62] is a real-world point cloud dataset based on scanned indoor scene data. Compared to synthetic datasets like the ModelNet40 [64], it is more challenging with backgrounds and occlusions. We choose three representative variants of ScanObjectNN for evaluation, i.e., OBJ_ONLY (only objects), OBJ_BG (objects with backgrounds), and PB_T50_RS (objects with backgrounds, random translation up to 50% of the size, random rotation, and scaling). We also use the overall accuracy (OA) as the evaluation metric. According to the experimental results in

TABLE III

CLASSIFICATION RESULTS ON THE SCANOBJECTNN [62] DATASET.

| Method | OBJ_ONLY | | | OBJ_BG | | | PB_T50_RS | | |
|---|---|---|---|---|---|---|---|---|---|
| | #TP(M) | FLOPs(G) | OA(%) | #TP(M) | FLOPs(G) | OA(%) | #TP(M) | FLOPs(G) | OA(%) |
| Scratch | 22.09 | 4.93 | 91.22 | 22.09 | 4.93 | 90.88 | 22.09 | 4.93 | 86.26 |
| Full | 22.09 | | 91.74 | 22.09 | | 92.94 | 22.09 | | **88.51** |
| Head | 0.27 | | 89.33 | 0.27 | | 88.81 | 0.27 | | 83.17 |
| Bias [16], [17] | 0.31 | | 91.05 | 0.31 | | 89.67 | 0.31 | | 85.50 |
| VPT-Shallow [28] | 0.28 (+0.01) | 5.16 (+0.24) | 90.19 | 0.28 (+0.01) | 5.40 (+0.48) | 89.85 | 0.28 (+0.01) | 5.40 (+0.48) | 83.93 |
| VPT-Deep [28] | 0.37 (+0.10) | 5.40 (+0.48) | 91.39 | 0.37 (+0.10) | 5.40 (+0.48) | 90.53 | 0.37 (+0.10) | 5.40 (+0.48) | 85.43 |
| Adapter [23], [24] | 7.40 (+7.11) | 5.84 (+0.91) | 92.25 | 7.40 (+7.11) | 5.84 (+0.91) | 93.29 | 2.05 (+1.78) | 5.15 (+0.22) | 87.86 |
| IDPT [31] | 1.71 (+1.43) | 7.28 (+2.35) | **93.12** | 1.71 (+1.43) | 7.28 (+2.35) | **93.63** | 1.71 (+1.43) | 7.28 (+2.35) | **88.51** |
| DA-Light | 0.42 (+0.15) | 4.94 (+0.01) | 92.08 | 0.53 (+0.26) | 4.95 (+0.02) | 92.25 | 0.42 (+0.15) | 4.94 (+0.01) | 87.37 |
| DA-Heavy | 2.51 (+2.24) | 6.00 (+1.07) | 92.43 | 2.51 (+2.24) | 6.91 (+1.98) | 93.46 | 1.60 (+1.33) | 6.82 (+1.89) | 87.99 |

TABLE IV

FEW-SHOT LEARNING ON THE MODELNET40 [64] DATASET.

| Method | #TP(M) | FLOPs(G) | OA(%) | #TP(M) | FLOPs(G) | OA(%) |
|---|---|---|---|---|---|---|
| | 5-way 10-shot | | | 5-way 20-shot | | |
| Full | 22.10 | 2.44 | 96.50 ± 2.22 | 22.10 | 2.44 | 97.80 ± 0.92 |
| Head | 0.28 | | 93.90 ± 2.77 | 0.28 | | 97.10 ± 2.08 |
| Bias [16], [17] | 0.31 | | 94.20 ± 2.97 | 0.31 | | 97.10 ± 2.47 |
| VPT-Shallow [28] | 0.29 (+0.01) | 2.89 (+0.45) | 93.60 ± 3.78 | 0.29 (+0.01) | 2.89 (+0.45) | 97.30 ± 1.77 |
| VPT-Deep [28] | 0.38 (+0.10) | 2.89 (+0.45) | 95.30 ± 2.67 | 0.33 (+0.05) | 2.66 (+0.22) | 97.70 ± 1.70 |
| Adapter [23], [24] | 7.40 (+7.12) | 2.90 (+0.46) | 96.70 ± 2.21 | 3.84 (+3.56) | 2.67 (+0.23) | 97.90 ± 1.45 |
| IDPT [31] | 1.71 (+1.43) | 3.61 (+1.17) | **97.30** ± 2.15 | 1.71 (+1.43) | 3.61 (+1.17) | 97.90 ± 1.14 |
| DA-Light | 0.76 (+0.48) | 2.46 (+0.02) | 96.50 ± 2.37 | 0.76 (+0.48) | 2.46 (+0.02) | 97.80 ± 1.32 |
| DA-Heavy | 2.15 (+1.87) | 2.95 (+0.51) | 97.20 ± 2.35 | 1.79 (+1.51) | 3.39 (+0.95) | **98.30** ± 1.57 |
| | 10-way 10-shot | | | 10-way 20-shot | | |
| Full | 22.10 | 2.44 | 92.85 ± 4.30 | 22.10 | 2.44 | 95.25 ± 2.76 |
| Head | 0.28 | | 90.30 ± 4.90 | 0.28 | | 93.95 ± 3.70 |
| Bias [16], [17] | 0.31 | | 90.30 ± 5.02 | 0.31 | | 94.20 ± 3.74 |
| VPT-Shallow [28] | 0.29 (+0.01) | 2.66 (+0.22) | 89.80 ± 5.63 | 0.29 (+0.01) | 2.89 (+0.45) | 94.25 ± 3.34 |
| VPT-Deep [28] | 0.38 (+0.10) | 2.89 (+0.45) | 91.45 ± 4.74 | 0.33 (+0.05) | 2.66 (+0.22) | 94.70 ± 3.61 |
| Adapter [23], [24] | 7.40 (+7.12) | 2.90 (+0.46) | 92.60 ± 4.40 | 3.84 (+3.56) | 2.67 (+0.23) | 95.15 ± 3.35 |
| IDPT [31] | 1.71 (+1.43) | 3.61 (+1.17) | 92.75 ± 4.06 | 1.71 (+1.43) | 3.61 (+1.17) | 95.40 ± 2.87 |
| DA-Light | 0.65 (+0.37) | 2.46 (+0.02) | 92.35 ± 4.87 | 0.85 (+0.57) | 2.47 (+0.03) | 94.95 ± 3.19 |
| DA-Heavy | 2.51 (+2.23) | 2.97 (+0.53) | **92.95** ± 3.76 | 2.15 (+1.87) | 3.41 (+0.97) | **95.45** ± 3.06 |

Tab. III, most methods (Head, Bias [16], [17], and VPT [28]) cannot reach the level of training from scratch, let alone full fine-tuning. Therefore, pre-trained knowledge cannot be effectively transferred to downstream tasks. On the contrary, DA-Light outperforms training from scratch and achieves close results to full fine-tuning with negligible additional parameters and FLOPs. If more parameters and higher computational burden are allowed, DA-Heavy, Adapter [23], [24] and IDPT [31] can reach or even surpass full fine-tuning. As for the performance gaps between our DA and the SOTA IDPT [31] (0.70% in OBJ_ONLY and 0.52% in PB_T50_RS), we rerun these two experiments using its official code and get 91.05% in OBJ_ONLY and 88.06% in PB_T50_RS. Random seeds may have an unignorable influence on the performances. Considering this, our results are still competitive compared with the SOTA ones.

### D. Few-shot Learning

Besides real-world datasets, we also conduct experiments on the ModelNet40 [64] under challenging few-shot settings [9], [10], [11]. We present the results in Tab. IV. In most cases, DA-Light performs similarly to full fine-tuning with around 2% tunable parameters and 1% additional FLOPs, while DA-Heavy sets SOTA results with more parameters and extra FLOPs.

### E. Point Cloud Part Segmentation

The ShapeNetPart [65] is a 3D object dataset designed for the part segmentation task. There are 14,006 objects in the training set and 2,874 in the test one from 16 shape categories and 50 part categories. We use 2,048 points from the sampled point clouds of [1] for fair comparisons and predict their part labels. Two metrics, i.e., instance mean IoU (ins. mIoU) and category mean IoU (cat. mIoU), are evaluated for each approach.

TABLE V

PART SEGMENTATION RESULTS ON THE SHAPENETPART [65] DATASET. MIOU = INS. MIOU / CAT. MIOU.

| Method | #TP(M) | FLOPs(G) | mIoU(%) |
|---|---|---|---|
| Scratch | 27.06 | 15.62 | 85.53 / 83.72 |
| Full | 27.06 | | **86.00 / 84.67** |
| Head | 5.24 | | 84.99 / 82.66 |
| Bias [16], [17] | 5.28 | | 85.04 / 82.40 |
| VPT-Shallow [28] | 5.26 (+0.02) | 16.10 (+0.48) | 84.36 / 80.93 |
| VPT-Deep [28] | 5.34 (+0.10) | 16.10 (+0.48) | 84.68 / 81.26 |
| Adapter [23], [24] | 7.03 (+1.79) | 15.85 (+0.23) | 85.41 / 83.50 |
| IDPT [31] | 5.69 (+0.45) | 15.69 (+0.07) | 85.71 / 83.40 |
| DA-Naive | 6.29 (+1.05) | 15.50 (-0.12) | 85.51 / 83.46 |
| DA-Light | 5.35 (+0.11) | 15.64 (+0.02) | 85.54 / 83.27 |

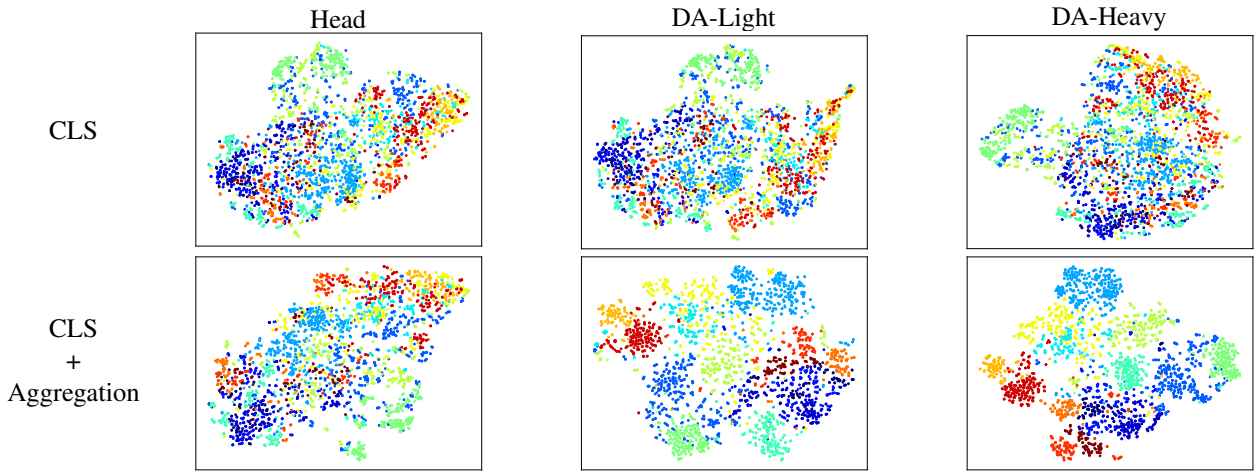


Fig. 2. The t-SNE visualization of the aggregate features on the ScanObjectNN [62] (PB_T50_RS). The first row is the visualization for the classification token, while the second row is for the classification token with static/dynamic aggregation. The visualization methods from left to right are head-only fine-tuning, DA-Light, and DA-Heavy, respectively. Best viewed in color.

Similar to IDPT [31], we also find that DA-Heavy with DGCNN [3] does not perform better than the simpler variants, so we do not include it in this experiment. As shown in Tab. V, DA-Naive and DA-Light have similar performances to training from scratch, slightly worse than IDPT [31]. As most efficient fine-tuning approaches cannot significantly outperform training from scratch, we argue that this dataset remains largely unsolved for efficient fine-tuning.

### F. Ablation Study

We conduct several ablation studies on the ScanObjectNN [62] to explore the architecture design of DA. We report the experimental results in Tab. VI.

*1) Dynamic Aggregation:* Besides DA-Light and DA-Heavy, we also conduct experiments with DA-Naive. However, most experimental results, especially those in the classification tasks, are not promising, as listed in Tab. VI. On the ScanObjectNN [62], DA-Naive generally introduces a similar number of parameters to DA-Heavy but even performs worse than DA-Light in most cases.

TABLE VI
ABLATION STUDIES ON THE SCANOBJECTNN [62] DATASET.

| Method | OA(%) | | |
|---|---|---|---|
| | OBJ_ONLY | OBJ_BG | PB_T50_RS |
| Baselines | | | |
| DA-Light | 92.08 | 92.25 | 87.37 |
| DA-Heavy | **92.43** | **93.46** | **87.99** |
| DA-Naive w/ different numbers of MLP layers | | | |
| 1-layer MLP | 91.05 | 91.74 | 85.98 |
| 2-layer MLP | 91.22 | 92.08 | 86.75 |
| 3-layer MLP | 91.05 | 92.60 | 86.50 |
| DA w/o classification tokens | | | |
| DA-Light | 91.57 | 91.91 | 86.95 |
| DA-Heavy | 91.39 | 92.60 | 87.26 |

*2) Classification Token:* Although DAM can replace the role of the classification token, it is still reserved in our implementation. According to Tab. VI, all variants perform poorly without the classification token. Since DAM is only inserted after the last layer of Transformer encoder, domain shifts in the first several layers may not be handled appropriately. On the other hand, classification tokens are placed before the first layer, filling in the gap between pre-trained models and downstream tasks. Through the combination of both, we get the best result in the experiment.

*3) Visualization:* Similar to IDPT [31], we also employ t-SNE [66] to visualize the effectiveness of DAMs. As shown in Fig. 2, static aggregation such as max pooling cannot enhance the distinguishability of head-only fine-tuning. In contrast, dynamic aggregation can effectively improve the performances of DA-Light and DA-Heavy.

## V. CONCLUSIONS

In this paper, we propose a novel DA method that fine-tunes point cloud Transformers efficiently. DA replaces previous static aggregation strategies, like mean/max pooling, to achieve better domain adaptation ability. Compared to other efficient fine-tuning approaches, DA has high flexibility and is more suitable for point cloud scenarios with promising results. Specifically, DA-Light with competitive performances only requires negligible parameters and FLOPs, while DA-Heavy performs as excellent as other SOTA methods. Although DA is designed for pre-trained point cloud Transformers, it can be simply deployed to fine-tune other pre-trained Transformers in a wide range of domains. Meanwhile, DA can also be applied in other non-Transformer point cloud architectures since it can dynamically aggregate features from embedding sets. We will expand our work along this line in the future.

## ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation of China (NSFC) under Grant No. 62176134 and by research and application on AI technologies for smart mobility funded by SAIC Motor.

## REFERENCES


[1] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

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

[3] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, oct 2019.

[4] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, "Rethinking network design and local geometry in point cloud: A simple residual MLP framework," in *Int. Conf. on Learning Representations (ICLR)*, 2022.

[5] G. Qian, Y. Li, H. Peng, J. Mai, H. Hammoud, M. Elhoseiny, and B. Ghanem, "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 23 192–23 204.

[6] A. Sanghi, "Info3d: Representation learning on 3d objects using mutual information maximization and contrastive learning," in *Computer Vision – ECCV*, Cham, 2020, pp. 626–642.

[7] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany, "Pointcontrast: Unsupervised pre-training for 3d point cloud understanding," in *Computer Vision – ECCV*, Cham, 2020, pp. 574–591.

[8] J. Hou, B. Graham, M. Niessner, and S. Xie, "Exploring data-efficient 3d scene understanding with contrastive scene contexts," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 15 587–15 597.

[9] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, "Point-bert: Pre-training 3d point cloud transformers with masked point modeling," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 19 313–19 322.

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

[11] R. Zhang, Z. Guo, P. Gao, R. Fang, B. Zhao, D. Wang, Y. Qiao, and H. Li, "Point-m2ae: Multi-scale masked autoencoders for hierarchical point cloud pre-training," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 27 061–27 074.

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

[13] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 80, 10–15 Jul 2018, pp. 40–49.

[14] Y. Yang, C. Feng, Y. Shen, and D. Tian, "Foldingnet: Point cloud auto-encoder via deep grid deformation," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[15] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, "Pointflow: 3d point cloud generation with continuous normalizing flows," in *Proc. of the IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, October 2019.

[16] H. Cai, C. Gan, L. Zhu, and S. Han, "Tinytl: Reduce memory, not parameters for efficient on-device learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 11 285–11 297.

[17] E. Ben Zaken, Y. Goldberg, and S. Ravfogel, "BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models," in *Proc. of the Annu. Meeting of the Association for Computational Linguistics (ACL)*, Dublin, Ireland, May 2022, pp. 1–9.

[18] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for NLP," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 97, 09–15 Jun 2019, pp. 2790–2799.

[19] J. Pfeiffer, A. Rücklé, C. Poth, A. Kamath, I. Vulić, S. Ruder, K. Cho, and I. Gurevych, "AdapterHub: A framework for adapting transformers," in *Proc. of the Conf. on Empirical Methods in Natural Language Processing: System Demonstrations*, Online, Oct. 2020, pp. 46–54.

[20] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych, "AdapterFusion: Non-destructive task composition for transfer learning," in *Proc. of the Conf. of the European Chapter of the Association for Computational Linguistics*, Online, Apr. 2021, pp. 487–503.

[21] R. Karimi Mahabadi, S. Ruder, M. Dehghani, and J. Henderson, "Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks," in *Proc. of the Annu. Meeting of the Association for Computational Linguistics (ACL)*, Online, Aug. 2021, pp. 565–576.

[22] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[23] S. Chen, C. GE, Z. Tong, J. Wang, Y. Song, J. Wang, and P. Luo, "Adaptformer: Adapting vision transformers for scalable visual recognition," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 16 664–16 678.

[24] D. Lian, D. Zhou, J. Feng, and X. Wang, "Scaling & shifting your features: A new baseline for efficient model tuning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 109–123.

[25] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 1877–1901.

[26] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," in *Proc. of the Annu. Meeting of the Association for Computational Linguistics (ACL)*, Aug. 2021, pp. 4582–4597.

[27] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *Proc. of the Conf. on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, Nov. 2021, pp. 3045–3059.

[28] M. Jia, L. Tang, B.-C. Chen, C. Cardie, S. Belongie, B. Hariharan, and S.-N. Lim, "Visual prompt tuning," in *Computer Vision – ECCV*, Cham, 2022, pp. 709–727.

[29] H. Bahng, A. Jahanian, S. Sankaranarayanan, and P. Isola, "Exploring visual prompts for adapting large-scale models," 2022, arXiv:2203.17274.

[30] Q. Huang, X. Dong, D. Chen, W. Zhang, F. Wang, G. Hua, and N. Yu, "Diversity-aware meta visual prompting," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2023, pp. 10 878–10 887.

[31] Y. Zha, J. Wang, T. Dai, B. Chen, Z. Wang, and S.-T. Xia, "Instance-aware dynamic prompt tuning for pre-trained point cloud models," in *Proc. of the IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, October 2023, pp. 14 161–14 170.

[32] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *Int. Conf. on Learning Representations (ICLR)*, 2021.

[33] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. of the Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, Minnesota, June 2019, pp. 4171–4186.

[35] L. Floridi and M. Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, pp. 681–694, 2020.

[36] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 119, 13–18 Jul 2020, pp. 1597–1607.

[37] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[38] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 15 750–15 758.

[39] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proc. of the IEEE/CVF*

*Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 16 000–16 009.

[40] C.-L. Li, M. Zaheer, Y. Zhang, B. Póczos, and R. Salakhutdinov, "Point cloud GAN," in *Int. Conf. on Learning Representations Workshop*, 2019.

[41] Y. Zhao, T. Birdal, H. Deng, and F. Tombari, "3d point capsule networks," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[42] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[43] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[44] Z. Qi, R. Dong, G. Fan, Z. Ge, X. Zhang, K. Ma, and L. Yi, "Contrast with reconstruct: contrastive 3d representation learning guided by generative pretraining," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, ser. ICML'23, 2023.

[45] L. Nunes, R. Marcuzzi, X. Chen, J. Behley, and C. Stachniss, "Segcontrast: 3d point cloud feature representation learning through self-supervised segment discrimination," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2116–2123, 2022.

[46] D. Wang and Z.-X. Yang, "Self-supervised point cloud understanding via mask transformer and contrastive learning," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 184–191, 2023.

[47] A. H. Gebrehiwot, P. Vacek, D. Hurych, K. Zimmermann, P. Pérez, and T. Svoboda, "Teachers in concordance for pseudo-labeling of 3d sequential data," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 536–543, 2023.

[48] J. Sanchez, J.-E. Deschaud, and F. Goulette, "Cola: Coarse label pre-training for 3d semantic segmentation of sparse lidar datasets," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2023, pp. 11 343–11 350.

[49] S. Paul, Z. Patterson, and N. Bouguila, "Crossmoco: Multi-modal momentum contrastive learning for point cloud," in *Conf. on Robots and Vision (CRV)*, 2023, pp. 273–280.

[50] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 27, 2014.

[51] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Computer Vision – ECCV*, Cham, 2016, pp. 69–84.

[52] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Computer Vision – ECCV*, Cham, 2016, pp. 649–666.

[53] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. Feris, "Spottune: Transfer learning through adaptive fine-tuning," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[54] E. J. Hu, yelong shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "LoRA: Low-rank adaptation of large language models," in *Int. Conf. on Learning Representations (ICLR)*, 2022.

[55] H. Touvron, M. Cord, A. El-Nouby, J. Verbeek, and H. Jégou, "Three things everyone should know about vision transformers," in *Computer Vision – ECCV*, Cham, 2022, pp. 497–515.

[56] R. Zhang, Z. Guo, W. Zhang, K. Li, X. Miao, B. Cui, Y. Qiao, P. Gao, and H. Li, "Pointclip: Point cloud understanding by clip," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 8552–8562.

[57] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, ser. Proceedings of Machine Learning Research, vol. 139, 18–24 Jul 2021, pp. 8748–8763.

[58] X. Liu, K. Ji, Y. Fu, W. Tam, Z. Du, Z. Yang, and J. Tang, "P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks," in *Proc. of the Annu. Meeting of the Association for Computational Linguistics (ACL)*, Dublin, Ireland, May 2022, pp. 61–68.

[59] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Comput. Surv.*, vol. 55, no. 9, jan 2023.

[60] Z. Wang, X. Yu, Y. Rao, J. Zhou, and J. Lu, "P2p: Tuning pre-trained image models for point cloud analysis with point-to-pixel prompting," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 14 388–14 402.

[61] Y. Zhang, K. Zhou, and Z. Liu, "Neural prompt search," 2022, arXiv:2206.04673.

[62] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *Proc. of the IEEE/CVF Int. Conf. on Computer Vision (ICCV)*, October 2019.

[63] J. Fei, Z. Zhu, W. Liu, Z. Deng, M. Li, H. Deng, and S. Zhang, "Dumlp-pin: A dual-mlp-dot-product permutation-invariant network for set feature extraction," *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, vol. 36, no. 1, pp. 598–606, Jun. 2022.

[64] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1912–1920.

[65] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *ACM Trans. on Graphics*, vol. 35, no. 6, 2016.

[66] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *J. of machine learning research*, vol. 9, no. 11, 2008.