On Geometry-Enhanced Parameter-Efficient Fine-Tuning for 3D Scene Segmentation

Liyao Tang
The University of Sydney
`ltan9687@uni.sydney.edu.au`

Zhe Chen[1*]
La Trobe University
`zhe.chen@latrobe.edu.au`

Dacheng Tao[2*]
Nanyang Technological University
`dacheng.tao@gmail.com`

Abstract

The emergence of large-scale pre-trained point cloud models has significantly advanced 3D scene understanding, but adapting these models to specific downstream tasks typically demands full fine-tuning, incurring high computational and storage costs. Parameter-efficient fine-tuning (PEFT) techniques, successful in natural language processing and 2D vision tasks, would underperform when naively applied to 3D point cloud models due to significant geometric and spatial distribution shifts. Existing PEFT methods commonly treat points as orderless tokens, neglecting important local spatial structures and global geometric contexts in 3D modeling. To bridge this gap, we introduce the Geometric Encoding Mixer (GEM), a novel geometry-aware PEFT module specifically designed for 3D point cloud transformers. GEM explicitly integrates fine-grained local positional encodings with a lightweight latent attention mechanism to capture comprehensive global context, thereby effectively addressing the spatial and geometric distribution mismatch. Extensive experiments demonstrate that GEM achieves performance comparable to or sometimes even exceeding full fine-tuning, while only updating 1.6% of the model's parameters, fewer than other PEFT methods. With significantly reduced training time and memory requirements, our approach thus sets a new benchmark for efficient, scalable, and geometry-aware fine-tuning of large-scale 3D point cloud models. Code is available at https://github.com/LiyaoTang/GEM.

1 Introduction

Point cloud semantic segmentation is a fundamental task for scene understanding that underpins many real-world applications, from autonomous driving and unmanned aerial vehicles to augmented reality [20, 83, 46]. Recent breakthroughs in large-scale models for language and 2D vision [16, 50, 59, 14, 58, 49] have spurred an analogous trend toward training powerful point cloud backbones capable of capturing rich semantics from 3D scenes [76, 77, 79].

Despite significant capabilities, adapting large pre-trained backbone models to specific downstream target domains is non-trivial. The typical strategy heavily relies on full fine-tuning, a process that is both computationally expensive and storage-intensive, since each downstream task demands an independent copy of all model parameters. Furthermore, fine-tuning large models typically requires extensive datasets to prevent overfitting and catastrophic forgetting [4, 51], while large datasets are usually inaccessible when adapted to smaller, specialized tasks. To address these issues, parameter-efficient fine-tuning (PEFT) methods offer a promising performance in reducing the cost of adaptation while reaching the performance of full fine-tuning with significantly fewer parameters.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

[*]Corresponding authors.
[1]Also affiliated with Cisco - La Trobe Centre for AI and IoT.
[2]College of Computing and Data Science, NTU, Singapore 639798.

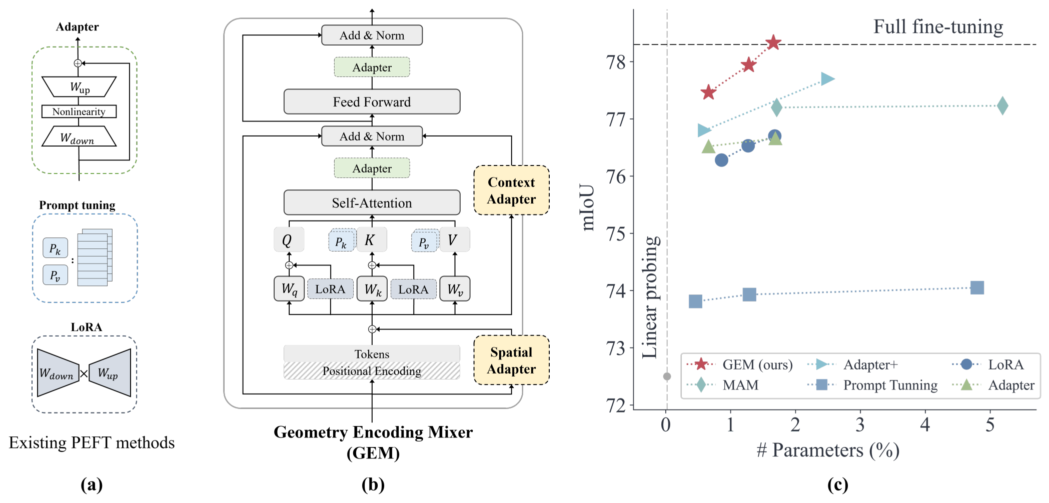

Figure 1. **(a)** Existing PEFT methods, such as adapters, prompt tuning, and LoRA, focus on adaptations inside attention and feed-forward layers. **(b)** In contrast, Geometry Encoding Mixer (GEM) explicitly encodes the geometric cues and mixes them into the pre-trained model, by the Spatial Adapter refining the pre-trained positional encoding and the Context Adapter complementing the local attention. **(c)** By capturing 3D spatial details and scene-wide geometry context, GEM reaches full fine-tuning performance while tuning $< 2\%$ parameters, outperforming existing PEFT methods.

While PEFT has been extensively validated in language processing and 2D vision tasks, its effectiveness on large-scale 3D point clouds remains largely unexplored and inadequately addressed. As demonstrated empirically in Fig. 1, existing PEFT methods only yield limited performance gains if directly applied to a 3D point cloud backbone. We tend to attribute this limitation to a key challenge that is often overlooked by current PEFT methods when using for 3D point cloud: unlike structured data such as text or images, point clouds are inherently unordered sets of coordinates in $\mathbb{R}^3$. They exhibit strong irregularity, sparsity, and structural variability, shaped by different sensing protocols and scene geometries. These factors lead to significant geometric and spatial distribution shifts between large-scale pre-training datasets and downstream domains, which existing PEFT methods fail to account for. In particular, representative PEFT methods, such as LoRA [26], adapters [25] and prompt tuning [37, 36], either adapt models at an isolated, per-point level or insert fixed external tokens at the global level, thus failing to adequately capture important geometric and spatial contexts inherent in 3D scenes. Moreover, to avoid the prohibitive computational cost of global attention over a large number of points, current 3D transformers predominantly employ local attention mechanisms without explicitly modeling global contexts, which further restricts the potential of current PEFT approaches in performance.

To address the above challenges, we re-examine PEFT for 3D scene understanding and hypothesize that effective PEFT on 3D scenes could take advantage of explicitly modeling both fine-grained local spatial patterns and global geometric contexts. We propose that neglecting either aspect would degrade fine-tuning performance: local-only adaptations may be prone to noise due to the lack of broader context consideration, whereas global-only methods would miss local details and lose precision. This motivates a novel PEFT adaptation framework tailored specifically for 3D scenes to bridge these local and global scales.

We introduce Geometry Encoding Mixer (GEM), a geometry-aware module for parameter-efficient fine-tuning of point cloud transformer on 3D scenes. It comprises two complementary components: a spatial adapter at local neighborhood and a context adapter capturing the scene context. In the spatial adapter, we employ a lightweight 3D convolutional bottleneck that operates on points neighborhood, enriching the pre-trained positional encoding by learning fine-grained local spatial details at target domains. In the context adapter, we introduce a set of learned latent tokens to serve as global context vectors. These latent tokens interact with the full point cloud through efficient attention, forming a bottleneck at the token dimension and bypassing the constraints of local attention, and thus aggregating scene-specific context from across the entire point cloud. By fusing these two paths, the Geometry Encoding Mixer effectively bridges local and global representations, providing the fine-tuned model with a richer understanding of the 3D scene than either component alone.

Empirically, we validate our approach on large-scale 3D scene datasets, including both indoor [12, 61, 2, 84] and outdoor [5] scenes. Our results indicate that models equipped with GEM consistently achieve performance matching or sometimes surpassing full fine-tuning methods, updating merely ~1.6% of model parameters, while being fewer than existing PEFT alternatives. These findings underscore the importance of explicitly modeling geometric and spatial contexts for efficient and effective adaptation in 3D scene understanding. To the best of our knowledge, our work represents the first exploration and validation of PEFT approaches tailored explicitly for large point cloud transformer under large-scale 3D scenes, hoping to establish a foundation for future research and practical deployments.

2 Related Work

Point cloud segmentation. Point clouds serve as an efficient representation for large-scale 3D scenes. Consequently, point cloud segmentation becomes a fundamental task that drives the design of 3D backbone architectures. Since the seminal work PointNet [53, 54], numerous 3D backbone architectures have been proposed for this task. These backbones either project the points onto a grid-like structure, *e.g.* 3D voxel grid, to exploit 3D convolutional networks [10, 70, 65, 21, 19, 13], or directly process the raw unordered point sets [73, 38, 75, 27, 66, 45]. Recently, inspired by the success of large transformer models in natural language and 2D vision [68, 16, 7, 9], researchers have started training increasingly large and powerful point cloud backbones [56, 91, 69, 57, 34, 78, 77]. Although these models achieve strong performance, they typically need to be trained from scratch for each new dataset and often require specialized training recipes to reach optimal results.

Following the paradigm of large-scale pre-training that has proven successful in vision and language domains [59, 33, 50, 14, 49, 23], researchers have begun exploring similar strategies for 3D point clouds. In particular, self-supervised learning (SSL) on large 3D scene datasets has demonstrated promising results [82, 44, 24, 80]. However, most such efforts focus on modest convolutional backbones like SparseUNet [10] rather than transformer-based architectures. Only recently has an SSL approach been applied to a transformer-based 3D backbone: Sonata [76] introduced SSL for a large point transformer encoder [77], but even this method still requires full fine-tuning on each downstream dataset to achieve optimal performance.

In this work, we explore effective PEFT methods for large pre-trained point cloud backbones, with the goal of reducing the computational and storage overhead when adapting them to downstream datasets, while matching the performance of full fine-tuning.

Parameter-efficient fine-tuning (PEFT). The ever-growing size of transformer-based foundation models[16, 33, 67] makes full fine-tuning for downstream task prohibitively expensive in both memory and computation. PEFT methods address this challenge by adapting large models while updating only a small fraction of their parameters. Existing approaches fall into four broad categories.

Selective fine-tuning methods update a carefully chosen subset of the original weights. One simple variant is linear probing [50, 15] that trains only the classification head. Other strategies restrict training to specific parts of the network, *e.g.*, tuning only bias terms [6] and selecting a subset of parameters based on gradient magnitude criteria [90].

Adapter-based tuning inserts lightweight bottleneck modules into an otherwise frozen backbone. First explored for CNNs [60] and later extended to transformers, adapters may be placed sequentially [25, 62] or in parallel [22, 93, 17, 8] to the original modules. Beyond single task, adapters can be composed or fused, promoting knowledge sharing without catastrophic forgetting [72, 51].

Prompt-based tuning techniques, including prompt tuning [36] and prefix tuning [37], extend to the prompting paradigm in large language models [41]. Instead of hand-crafted prompts, these methods learn task-specific vectors that are prepended to the input sequence [3, 36] or injected as extra tokens in transformer layers [37, 43, 31], thereby steering the model without modifying its original weights [36, 41].

Low-rank adaptation (LoRA) constrains the weight updates to a low-dimensional subspace by learning a pair of low-rank matrices that are added to each pre-trained weight, typically in the attention layers [26]. This design approximates the effect of full fine-tuning while introducing only a small number of additional parameters. Successors further enhance the approximation ability, robustness, or rank allocation [32, 42, 88, 81, 40].

Recent works also propose hybrid schemes that highlight common design principles [28], for instance, integrating their respective strengths under a unified adapter framework [22].

Although PEFT has proved effective in language and 2D vision, a direct transfer to 3D scene understanding is often sub-optimal, largely due to the unordered nature of point clouds and the prominence of geometric cues. Our work thus investigates PEFT strategies tailored to large-scale 3D scenes.

PEFT for 3D point cloud. In the context of 3D point clouds, PEFT remains relatively underexplored. Existing methods thus far have focused mostly on object-level inputs with limited spatial scale. Some approaches introduce prompt tokens that adapt to 3D data, where the prefix tokens are dynamically generated from intermediate features [87, 92, 1] or from the spatial centers of local patches [89]. Several works also introduce auxiliary side networks, operating either in the spectral domain [39] or the spatial domain [18, 71], to provide geometric context. Other methods construct banks of prompt vectors using domain-specific dataset to inject 3D prior knowledge [64], or combine language-side prompt tuning with point-cloud adapters to handle open-vocabulary recognition [63].

Compared to object-level datasets [85, 48], 3D scene understanding involves much larger inputs containing on the order of millions of points [12, 2]. This scale amplifies computational challenges and underscores the need for PEFT techniques expressly designed for large 3D scene models, which is however a research direction that remains largely unexplored.

3 Methodology

We propose the *Geometry Encoding Mixer* (GEM) as a lightweight parameter-efficient module for fine-tuning point-cloud transformer. Fig. 2 summarises the overall architecture.

In particular, GEM consists of two complementary adaptations: a *Spatial Adapter* that refines the positional encoding of each point, injecting local geometry information overlooked by generic adapters; and a *Context Adapter* that distills a compact set of latent tokens that broadcast global scene-level cues. Both components follow the residual, bottleneck design of classic adapters, yet emphasize and operate on spatial rather than channel space.

3.1 Preliminaries

We first consider the direct application of existing PEFT methods on current point-cloud transformers.

Challenges under 3D. Following transformer [68, 16], point-cloud transformers also build upon self-attention layer (Attn) followed by feed-forward network (FFN). However, due to the large scale input points at *million* scale, global self-attention over all points would demand prohibitively large GPU memory and compute resource, if ever possible, due to the quadratic complexity of the attention operations: $\mathcal{O}(\text{Attn}) = n^2$, where n denotes the number of input points. In this regard, state-of-the-art point-cloud transformers propose various designs of local attentions [91, 78, 77, 34, 69] to cap the complexity, where each point can only attend to points within the same patch. With a patch size of p, the complexity of local attention is then $\mathcal{O}(\text{Attn}_{\text{loc}}) = np$. For example, it is set to $p = 16$ in PT [91] and $p = 1024$ in the larger PTv3 [77].

The irregular and sparse nature of 3D point cloud, together with these architecture designs, greatly impede the application of standard PEFT methods such as adapter, LoRA, and prompt tuning. As shown in Fig. 1, directly applying existing methods yields only marginal gains. We attribute this to their lack of spatial structure awareness of 3D geometry and the inability to capture scene-wide context under the local attention regimes of modern point cloud backbones.

Adapters. The adapter-based methods [25, 62] insert small modules after the attention or FFN layers:

$$f_{\text{adapter}}(\boldsymbol{x}) = \boldsymbol{x} + \sigma(\boldsymbol{x}\boldsymbol{W}_{\text{down}})\boldsymbol{W}_{\text{up}}, \tag{1}$$

where $\boldsymbol{W}_{\text{down}} \in \mathbb{R}^{d\times r}$ and $\boldsymbol{W}_{\text{up}} \in \mathbb{R}^{r\times d}$ are down and up projections that form a bottleneck structure with $r \ll d$ and activation σ in the mid, and are also surrounded by residual connection. As generic

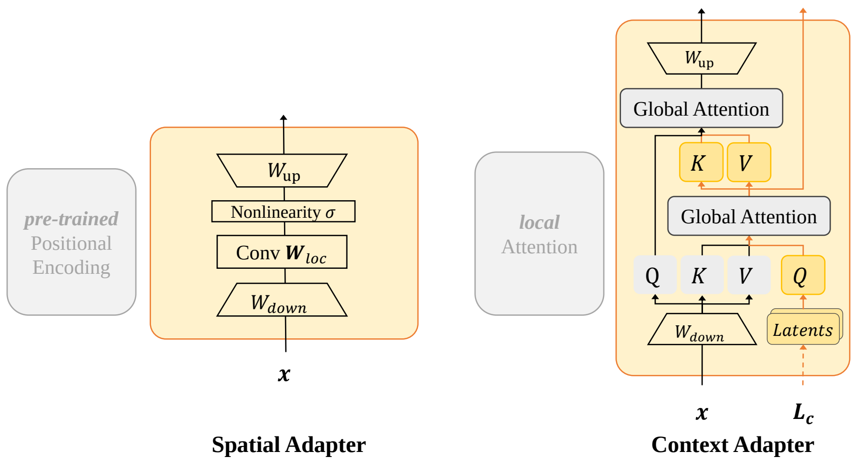

Figure 2. **Geometry Encoding Mixed.** We propose the spatial adapter to enhance the pre-trained positional encoding, and the context adapter to overcome the local attention mechanism, thus enhancing the efficient adaptation on large-scale 3D scenes with explicit geometry encoding.

MLP bottlenecks, adapters adapt the model on a per-point basis, overlooking the important spatial cues in 3D point cloud.

LoRA. LoRA methods [26] typically adapt the attention projection layers and approximate the full fine-tuning with a bottleneck in weight space, transforming the attention into:

$$\text{Attn}_{\text{LoRA}} = \text{Attn}(\boldsymbol{Q} + \Delta\boldsymbol{Q}, \boldsymbol{K} + \Delta\boldsymbol{K}, \boldsymbol{V}), \tag{2}$$

where $\Delta\boldsymbol{Q} = \boldsymbol{x}\boldsymbol{W}_{\text{down}}\boldsymbol{W}_{\text{up}}$, with $\boldsymbol{W}_{\text{down}} \in \mathbb{R}^{d\times r}$ and $\boldsymbol{W}_{\text{up}} \in \mathbb{R}^{r\times d}$, and similarly for $\Delta\boldsymbol{K}$ with a different pair of weights. While LoRA can leverage the strong ability of the attention layers, we note that all attentions in point cloud transformer are constrained within local patches. LoRA methods are thus inherently limited by the prevalent design of local attention. With only limited rank $r << d$, it could fail to provide global context of the target dataset.

Pompt tuning. Prompt tuning methods [37, 36] explicitly introduce global tokens into the attention:

$$\text{Attn}_{\text{prompt}} = \text{Attn}(\boldsymbol{Q}, [\boldsymbol{P}_K; \boldsymbol{K}], [\boldsymbol{P}_V; \boldsymbol{V}]), \tag{3}$$

where $\boldsymbol{P}_K, \boldsymbol{P}_V \in \mathbb{R}^{m\times d}$ are two sets of tokens prepending to the original key-value pair. In spite of the constraint of local patches, prompt tokens can be duplicated into each patch. However, with $m << n$, it can be hard for a few static external tokens to capture scene-specific context that can vary across point clouds within the same dataset. In addition, it also overlooks the spatial pattern of the point cloud, lacking the adaptation to local point features.

Others. There are more methods that explore other aspects. Selective tunings propose to tune a small subset of the frozen model parameters, such as BitFit [6] that tunes only bias terms. Hybrid methods are also explored, such as MAM [22] that couples FFN adapter with prompt tuning for attention layer. These methods still inherit the constraints from those representative PEFT methods, which are hindered by the local attention and fail to recognize the spatial structure.

In summary, existing methods overlook 3D geometry and remain confined to local patches, motivating a PEFT design that incorporates spatial awareness with scene-wide context, introduced next.

3.2 Our Methodology: Geometry Encoding Mixer (GEM)

To address these limitations, we present the Geometry Encoding Mixer (GEM). It refines positional encodings and injects scene context for efficient adaptation on large-scale 3D scenes:

$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \text{pos}(\boldsymbol{x}) + f_{\text{spatial}}(\boldsymbol{x}), \tag{4}$$
$$\boldsymbol{x} \leftarrow \boldsymbol{x} + \text{Attn}_{\text{loc}}(\boldsymbol{x}) + f_{\text{context}}(\boldsymbol{x}), \tag{5}$$

where $\boldsymbol{x}$ denotes input points and $\text{pos}(\cdot)$ is the pre-trained positional embedding. Fig. 2 illustrats the overall structure.

Spatial Adapter (SA). Point clouds are sparse, irregular samples in 3D. Adapting to their fine-grained geometry requires explicitly modeling local neighborhoods, which prior PEFT methods fail to address.

To this end, the spatial adapter refines per-point positional encoding through a lightweight 3D convolutional bottleneck. Concretely, for each point $\boldsymbol{x}$, we consider a 3D grid and gather points in the vicinity voxels as neighbors, denoted by $\mathcal{N}$:

$$f_{\text{spatial}}(\boldsymbol{x}) = \boldsymbol{x} + \sigma\Big(\sum_{i \in \mathcal{N}} \boldsymbol{W}^{i}_{\text{loc}}(\boldsymbol{x}\boldsymbol{W}_{\text{down}})\Big)\boldsymbol{W}_{\text{up}}, \tag{6}$$

where $\boldsymbol{W}_{\text{down}} \in \mathbb{R}^{d\times r}$ and $\boldsymbol{W}_{\text{up}} \in \mathbb{R}^{r\times d}$ are down and up projectsion with $r \ll d$. $\boldsymbol{W}^{i}_{\text{loc}} \in \mathbb{R}^{r\times r}$ composes the spatial kernel weights for i-th neighboring voxel and $\sigma(\cdot)$ is the nonlinearity, such as ReLU.

With a common kernel dimension $k = 3$, the spatial adapter touches at most k^3 neighbours per point and adds $2rd + k^3r^2$ parameters, rendering a complexity of $\mathcal{O}(2ndr + nk^3r^2) = \mathcal{O}(nd)$ given $k, r \ll n, d$. It thus functions as an efficient convolution-based positional encoding [11, 29] in parallel to the pre-trained positional encoding, thereby capturing fine spatial layout to match the target distribution.

Context Adapter (CA). Due to the local attention, any local and channel-wise adaptation would be constrained from acquiring scene-wide context, hindering the adaptation performance on downstream tasks like semantic segmentation.

To inject global context without breaking the $\mathcal{O}(n^2)$ barrier, we introduce m latent tokens $\boldsymbol{L} \in \mathbb{R}^{m\times r}$, where $m \ll n$, that attend to all points once:

$$\boldsymbol{L}_c = \text{Attn}\big(\boldsymbol{L}\boldsymbol{W}^{Q},\, \boldsymbol{K},\, \boldsymbol{V}\big), \tag{7}$$

$$f_{\text{context}}(\boldsymbol{x}) = \boldsymbol{x} \;+\; \text{Attn}\big(\boldsymbol{Q},\, \boldsymbol{L}_c\boldsymbol{W}^{K},\, \boldsymbol{L}_c\boldsymbol{W}^{V}\big)\boldsymbol{W}_{\text{up}}, \tag{8}$$

where $\boldsymbol{L} \in \mathbb{R}^{m\times r}$ and $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} = \boldsymbol{x}\boldsymbol{W}^{Q}_{\text{down}}, \boldsymbol{x}\boldsymbol{W}^{K}_{\text{down}}, \boldsymbol{x}\boldsymbol{W}^{V}_{\text{down}}$ are down-projected point features.

As both attention costs $O(nm)$ with $m \ll n$, global aggregation is affordable and at the same level of complexity as the conventional prompt tuning [37, 36]. Furthermore, we update $\boldsymbol{L} \leftarrow \boldsymbol{L} + \boldsymbol{L}_c$ at each adapter insertion to share across layers, yielding dynamic prompts that capture the context of current scene, unlike static prefixes in prompt tuning.

Discussion. In comparison to existing PEFT methods, GEM explicitly biases the adaptation to 3D geometry and context. The spatial adapter captures localized variations that generic adapters, such as LoRA and basic adapters, would miss, since those ignore spatial structure and use the positional encodings from pre-training. The context adapter circumvents the local attention design to provide scene-wide context for efficient adaptation with few parameters. By combining these two components, our approach enables scene-aware adaptation of 3D transformers, where each point is tuned with respect to both its neighbors and the entire point cloud. It thus results in significantly improved performance on 3D scene datasets with only a lightweight adaptation overhead[1].

4 Experiments

We present the results of our proposed GEM on semantic segmentation of large-scale 3D scene datasets and experiment with both self-supervised pre-trained backbone, Sonata [76], and supervised pretrained one, PTv3-PPT [79], for investigation[2]. We also provide ablation studies to reveal the detailed effects of different components better.

4.1 Experimental Setup

We primarily follow the comprehensive protocols proposed in Sonata [76], covering ScanNet [12], ScanNet200 [61], ScanNet++ [84] and S3DIS [2].

[1]We study the empirical overheads in the supplementary Sec. A.

[2]For more generalization and comparisons in other settings, such as convolutional model and 3D shape analysis, please refer to the supplementary Sec. B.

Semantic Seg.	Params		ScanNet Val [12]			ScanNet200 Val [61]			ScanNet++ Val [84]			S3DIS Area 5 [2]			S3DIS 6-fold [2]		
Methods	Learn.	Pct.	mIoU	mAcc	allAcc	mIoU	mAcc	allAcc	mIoU	mAcc	allAcc	mIoU	mAcc	allAcc	mIoU	mAcc	allAcc
Training from scratch																	
SparseUNet [10]	39.2M	100%	72.3	80.2	90.0	25.0	32.9	80.4	28.8	38.4	80.1	66.3	72.5	89.8	72.4	80.9	89.9
PTv3 [77]	124.8M	100%	77.6	85.0	92.0	35.3	46.0	83.4	42.1	53.4	85.6	73.4	78.9	91.7	77.7	85.3	91.5
Full fine-tuning																	
Sonata (full.) [76]	124.8M	100%	79.4	86.1	92.5	36.8	46.5	84.4	43.7	55.8	86.6	76.0	81.6	93.0	82.3	89.9	93.3
Sonata (ft.)	108.5M	100%	78.3	85.9	92.3	37.3	47.8	83.7	49.8	61.2	87.6	72.4	79.0	92.2	79.5	87.3	92.3
PEFT methods																	
Sonata (lin.)	0.02M	0.02%	72.5	83.1	89.7	29.3	41.6	81.2	37.3	50.9	84.3	72.3	81.2	90.9	76.5	87.4	90.8
+BitFit [6]	0.2M	0.2%	74.7	84.7	90.8	32.5	45.5	82.0	42.4	56.5	85.7	73.9	82.0	91.5	76.1	87.1	91.0
+Adapter [25]	2.8M	2.5%	77.0	85.4	91.8	33.6	45.7	82.5	42.6	57.5	85.6	73.8	82.9	91.5	76.4	87.5	91.4
+LoRA [26]	1.9M	1.7%	76.7	85.7	91.7	33.6	45.5	82.7	44.2	58.0	**86.5**	74.5	**83.2**	91.5	77.4	87.8	91.2
+Prompt Tunning [37]	5.5M	4.8%	74.3	84.1	90.5	31.4	44.4	81.6	41.2	56.2	84.8	73.4	82.5	91.0	73.7	86.5	90.5
+**GEM (ours)**	1.8M	1.6%	**78.3**	**86.6**	**92.3**	**35.6**	**46.9**	**83.3**	**46.6**	**60.3**	86.3	**75.1**	83.0	**92.1**	**77.9**	**88.2**	**92.1**

Table 1. **Semantic segmentation.**

We adopt two representative pre-trained backbones, the Sonata model [76] with large-scale self-supervised training and the PTv3-PPT [79] with supervised pre-training on multiple large-scale curated datasets. We compare GEM with the existing popular PEFT methods, including bias-tuning from BitFit [6], Adapter [25], LoRA [26], and Prompt Tuning [37]. We consider linear probing as a baseline and the full fine-tuned model as our target strong reference. In addition, we note that the Sonata (full.) specifically introduces a task-specific decoder for semantic segmentation during the fine-tuning, which may not reflect the fair comparison. Therefore, we revive to the standard full fine-tuning without additional decoder network, which is denoted by Sonata (ft.).

For training, we follow the widely accepted fine-tuning setups to update only the inserted or selected weights with the pre-trained backbone weights remaining frozen. All PEFT baselines follow the implementations from released code, adopt the suggested common practice [47, 52], and are tuned to their best validation setting in Fig. 1(c). Specifically, we set the default rank to be $r = 32$ and the number of learnable tokens to be $m = 4$. More details are given in the supplementary.

4.2 Performance Comparison

Main results. Tab. 1 shows that GEM consistently surpasses all representative PEFT methods across datasets. It matches the performance of full fine-tuning on most datasets and even exceeds it on ScanNet++[84]. ScanNet++ comprises large and diverse scenes captured at sub-millimeter resolution, diverging greatly from the spatial distribution of common pre-training datasets such as ScanNet [12]. Under this domain gap, the joint modeling of local spatial cues and global scene context introduced by GEM proves especially beneficial.

Interestingly, LoRA [26] and adapter [25] deliver similar scores, despite acting on different transformer components. This observation implies that, when updates remain local and geometric is not modeled explicitly, the precise choice of adaptation target can be secondary. In addition, prompt tuning [37] underperforms even the linear probing baseline in the S3DIS 6-fold evaluation, revealing the penalty of ignoring spatial structure during fine-tuning.

Data efficiency. We assess PEFT methods under the scenarios of limited data and annotations in Tab. 2. The results demonstrate the exceptional data efficiency of GEM, which outperforms both other PEFT methods and the full fine-tuning counterparts. Notably, under extreme data scarcity, such as 1% of the labeled scenes and limited annotations (20 points per scene), GEM surpasses both Sonata (full.) and Sonata (ft.), highlighting its superiority in low-data regimes.

Supervised pre-training. While self-supervised methods dominate the language domain, supervised pre-training remains competitive for vision [33]. To probe the limits of PEFT under large-scale 3D supervision, we adopt recent advances in 3D supervised pre-training [79], which achieve leading performance by training on a large collection of curated, labeled scene datasets. As shown in Tab. 3, existing PEFT methods can even degrade performance, likely suffering from negative transfer [86, 74]. In contrast, GEM improves upon supervised backbone, further outperforming the larger PTv3 [77] with only 1.6% additional parameters.

PEFT with decoder. A dedicated segmentation decoder (13% of the total parameters) is known to lift fully fine-tuned baselines. We therefore ask whether PEFT can still add value in the presence of

Data Efficiency	Limited Scenes (Pct.)					Limited Annotation (Pts.)				
Methods	1%	5%	10%	20%	Full	20	50	100	200	Full
Training from scratch										
SparseUNet [10]	26.0	47.8	56.7	62.9	72.2	41.9	53.9	62.2	65.5	72.2
PTv3 [77]	25.8	48.9	61.0	67.0	77.2	60.1	67.9	71.4	72.7	77.2
Full fine-tuning										
Sonata (full) [76]	45.3	65.7	72.4	72.8	79.4	70.5	73.6	76.0	77.0	79.4
Sonata (ft.)	44.4	63.2	71.3	72.3	78.3	69.6	72.6	75.3	76.2	78.3
PEFT methods										
Sonata (lin.)	43.6	62.5	68.6	69.8	72.5	69.0	70.5	71.1	71.5	72.5
+BitFit [6]	46.5	64.9	69.9	71.7	74.7	71.0	72.3	72.9	73.6	74.7
+Adapter [25]	46.4	64.2	70.1	72.2	77.0	71.8	73.4	74.7	75.0	77.0
+LoRA [26]	46.6	63.0	70.1	72.6	76.7	72.1	73.6	75.2	75.5	76.7
+Prompt Tunning [37]	45.5	62.6	68.9	71.1	74.3	70.2	71.5	72.5	72.8	74.3
+GEM (ours)	**47.5**	**65.6**	**71.0**	**73.3**	78.3	**72.3**	**74.7**	**76.2**	**76.6**	78.3

Table 2. **Data efficiency.**

Supervised Pre-train.	Params		ScanNet Val [12]		
Methods	Learn.	Pct.	mIoU	mAcc	allAcc
Training from scratch					
SparseUNet [10]	39.2M	100%	72.3	80.2	90.0
PTv3 [77]	124.8M	100%	77.6	85.0	92.0
Full fine-tuning					
PTv3-PPT (ft.) [79]	97.4M	100%	78.6	86.0	92.5
PEFT methods					
PTv3-PPT (lin.)	0.1M	0.1%	78.6	85.9	92.5
+BitFit [6]	0.2M	0.2%	78.2	85.6	92.3
+Adapter [25]	1.8M	1.8%	78.5	85.9	92.4
+LoRA [26]	1.4M	1.7%	78.4	86.0	92.4
+Prompt Tunning [37]	4.8M	4.8%	78.3	85.9	92.4
+GEM (ours)	1.8M	1.6%	**79.1**	**86.6**	**92.6**

Table 3. **Supervised pre-training.**

with Decoder	Params		ScanNet Val [12]		
Methods	Learn.	Pct.	mIoU	mAcc	allAcc
Training from scratch					
PTv3 [77]	124.8M	100%	77.6	85.0	92.0
Full fine-tuning					
Sonata (full.) [76]	124.8M	100%	79.4	86.1	92.5
Sonata (full.)*	124.8M	100%	78.5	86.3	92.4
PEFT methods with decoder					
Sonata (dec.)	16.3M	13.1%	79.1	86.6	92.7
Sonata (dec.)*	16.3M	13.1%	77.2	85.9	91.9
+BitFit [6]	16.4M	13.2%	78.1	86.1	92.3
+Adapter [25]	19.1M	15.0%	78.2	86.5	92.3
+LoRA [26]	18.2M	14.3%	78.9	87.0	92.5
+Prompt Tunning [37]	22.6M	17.2%	77.4	86.1	92.1
+GEM (ours)	18.1M	14.3%	**79.5**	**87.3**	**92.7**

Table 4. **PEFT with segmentation decoder.** Methods with * reports our re-produced results.

Outdoor Seg.	Params		Sem.KITTI Val [5]		
Methods	Learn.	Pct.	mIoU	mAcc	allAcc
Training from scratch					
PTv3 [77]	124.8M	100%	69.1	76.1	92.6
Full fine-tuning					
Sonata (full.) [76]†	124.8M	100%	72.6	77.9	93.4
Sonata (ft.)	108.5M	100%	68.8	75.2	92.6
PEFT methods					
Sonata (lin.)†	0.02M	0.02%	62.0	72.5	91.0
Sonata (lin.)	0.02M	0.02%	52.0	63.3	87.1
+BitFit [6]	0.2M	0.2%	59.9	70.6	91.0
+Adapter [25]	2.8M	2.5%	62.5	72.2	92.0
+LoRA [26]	1.9M	1.7%	63.8	74.0	92.0
+Prompt Tunning [37]	5.5M	4.8%	59.1	71.0	90.8
+GEM (ours)	1.8M	1.6%	**67.7**	**75.5**	**93.1**

Table 5. **Outdoor semantic segmentation.** Methods with † use outdoor datasets for pre-training.

such a task-specific head. Tab. 4 answers in the affirmative: GEM outperforms all competing PEFT variants and even surpasses its own fully fine-tuned counterpart, setting a new state-of-the-art on ScanNet. The result underscores the merit of jointly modeling local geometry and scene-level context, even when the decoder already provides task-specific capacity.

Outdoor segmentations. During the original evaluation of Sonata [76], it employs separate pre-trained models for indoor and outdoor scenarios, due to the significant domain gaps between indoor and outdoor scenes. Here, we test a stronger setting: transferring an indoor pre-trained backbone to an outdoor autonomous-driving benchmark. We replace the input layer with a randomly initialized counterpart to match dimensionality and freeze all remaining weights.

Tab. 5 shows that GEM narrows the gap to a model trained from scratch on outdoor data, yet a noticeable performance gap persists. Closing this gap remains an interesting direction for future work.

4.3 Ablations and Analsys

We mainly consider the linear probing, Sonata (lin.), as baseline and ablate on ScanNet [12] to better investigate the key factors for effective PEFT in 3D scenes, shown in Tab. 6. For more studies on the effect of available parameters, please refer to the supplement.

Individual effectiveness of SA and CA. To validate our initial hypothesis, we study the individual effectiveness of the SA and CA in Tab. 6a. We find that both adapters can already be superior to the baseline and competitive to the existing PEFT methods. We notice that the SA takes the majority part of the introduced parameters, which is partially due to the inherently expensive 3D convolution with k^3 kernels. Such challenge is also 3D specific and urges us to develop the CA, rather than solely

	SA	CA	Params	mIoU
Sonata (lin.)			0.02%	72.5
	✓		1.0%	77.2
+ GEM		✓	0.6%	77.3
	✓	✓	1.6%	78.3

(a) **SA** and **CA** compose of effective PEFT for 3D scenes, individually and jointly.

m	mIoU	mAcc	allAcc
1	78.1	86.4	92.1
4	78.3	86.6	92.3
8	78.2	86.8	92.3

(b) **GEM** is robust to available latent tokens.

strategy	mIoU	mAcc	allAcc
N/A.	77.8	86.1	92.1
per-stage.	78.0	86.1	92.0
global	78.3	86.6	92.3

(c) **GEM** can provide better capabilities when sharing latent tokens.

Table 6. Ablations on GEM. If not specified, the backbone is the self-supervised pre-trained Sonata [76] with no additional task-specific decoder, evaluated on ScanNet [12]. Default settings are marked in gray.

relying on the positional encoding to capture 3D geometry. By combining the two form of spatial adapters, we obtain GEM that reaches the best performance by comprehensively exploring the 3D scene while remaining parameter-efficient.

Understanding the latent tokens. In Tab. 6b, we are motivated to further investigate how efficient and effective the proposed CA could be. We thus study how few tokens it could afford before failing to capture the global context. Surprisingly, we find that CA can use as few as one single token to effectively express the global scene context, obtaining clear improvement over SA-only approach. We consider this to be the effectiveness of weight decomposition, where we are akin to approximating the n^2 global self-attention weights with two $n \times 1$ matrices. In comparison to LoRA that decomposes in the weight space, we decompose the spatial attention weight matrices and form a spatial bottleneck at token dimension. In addition, while we are aware that related topics have been similarly explored for efficient attentions [30, 35], we are, to the best of our knowledge, the first to motivate their effective application as PEFT methods for 3D scenes understanding.

Sharing the latent tokens. The last experiment, Tab. 6c probes whether the latent tokens learned by CA should remain layer-local or be shared across the hierarchy, assessing its ability to capture dynamic and scene-specific context. When each layer keeps its own bank of tokens (*N/A*), GEM already surpasses SA-only, confirming that CA supplies complementary global cues. Re-using the same tokens within each encoder stage (*per-stage*) yields a further yet modest gain, suggesting that a coherent context inside each resolution level helps the network reconcile local geometry with mid-range structure.

The strongest result emerges when a single set of latent tokens is shared by *all* transformer blocks. We attribute this improvement to its consistency with the fact that 3D point clouds represent the same underlying 3D geometry in spite of their irregularity and sparsity. From this perspective, such globally shared latent tokens confine the model to adapt to the actual underlying scene, rather than overfitting on irrelevant patterns due to the noisy sampled point cloud.

Revealing the geometry cues in latent tokens. Under the above assumption that latent tokens regularize the model to capture the underlying 3D scene, we visualize the attention weights and compare them to those produced by other PEFT methods that also inject global context, such as prompt tuning.

As shown in Fig. 3, starting from the first stage, our latent tokens learn to produce crisper attention weights that follow geometric cues, such as the simple geometric primitives of floor surface. In deeper stages, the tokens progressively attend to more salient geometric objects, including clutter on the table and sofa. In contrast, the static tokens introduced by prompt tuning produce yield blurrier attention patterns that often span across object and region boundaries.

These observations suggest that the proposed latent tokens effectively capture and broadcast scene-wide geometric context. Combined with the local spatial structure extracted by 3D convolutions, this leads to superior performance, establishing our method as an effective PEFT approach for 3D scene understanding.

5 Conclusion

This study presents GEM, a geometry-aware fine-tuning module tailored for large-scale 3D scene segmentation. By combining local spatial refinement and global context modeling, GEM addresses key challenges inherent in adapting pre-trained 3D models to new domains. Experimental results across multiple benchmarks confirm its ability to achieve competitive performance with minimal

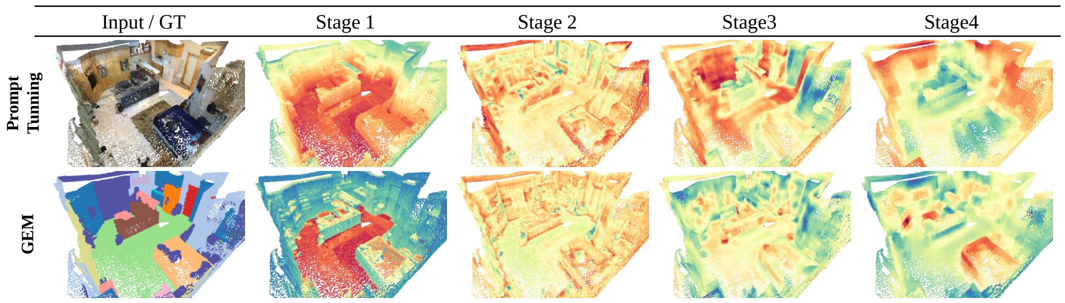

Figure 3. We visualize the attention weights of our latent tokens, comparing to the attentional weights with the prompt tuning, showing the enhanced geometry cues produced by GEM.

parameter updates. These findings highlight the value of incorporating geometric priors into efficient model adaptation and offer a promising direction for scalable deployment in 3D scene understanding tasks.

Limitation and future work. While GEM achieves promising performance with little overhead in complexity, we notice it relies on the assumption that the model can be fully parallelized. For example, it would incur noticeable overhead if using large batch sizes, as the device is forced to process the input sequentially, in spite of the parallel design in GEM. In addition, we are able to successfully train our PEFT methods with a fraction of epochs than the full fine-tuning, *e.g.* 100 rather than 800 epochs, we realize that the gradients need to be back-propagated into early layers and could hinder the training efficiency, especially with dense point clouds like S3DIS [2]. We thus suggest that it is required a systematic examination on the additional overhead that PEFT methods would experience in 3D scene understanding.

Acknowledgement. Dr. Tao's research is partially supported by NTU RSR and Start Up Grants.

References

[1] Zixiang Ai, Zichen Liu, Yuanhang Lei, Zhenyu Cui, Xu Zou, and Jiahuan Zhou. Gaprompt: Geometry-aware point cloud prompt for 3d vision model, 2025. 4

[2] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017. 3, 4, 6, 7, 10

[3] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. Exploring visual prompts for adapting large-scale models, 2022. 3

[4] Ankur Bapna and Orhan Firat. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 2019. 1

[5] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019. 3, 8

[6] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. BitFit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9, Dublin, Ireland, May 2022. Association for Computational Linguistics. 3, 5, 7, 8, 23

[7] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 3

[8] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *arXiv preprint arXiv:2205.13535*, 2022. 3

[9] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022. 3

[10] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 3, 7, 8, 22, 23

[11] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, and Chunhua Shen. Conditional positional encodings for vision transformers. In *The Eleventh International Conference on Learning Representations*, 2023. 6

[12] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern*

Recognition (CVPR), IEEE, 2017. 3, 4, 6, 7, 8, 9, 22, 23

[13] Angela Dai, Daniel Ritchie, Martin Bokeloh, Scott Reed, Jürgen Sturm, and Matthias Nießner. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans. *CoRR*, abs/1712.10215, 2017. 3

[14] DeepSeek-AI. Deepseek-v3 technical report, 2025. 1, 3

[15] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 647–655, Bejing, China, 22–24 Jun 2014. PMLR. 3

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 1, 3, 4

[17] Chin-Lun Fu, Zih-Ching Chen, Yun-Ru Lee, and Hung-yi Lee. AdapterBias: Parameter-efficient token-dependent representation shift for adapters in NLP tasks. In Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, editors, *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2608–2621, Seattle, United States, July 2022. Association for Computational Linguistics. 3

[18] Takahiko Furuya. Token adaptation via side graph convolution for efficient fine-tuning of 3d point cloud transformers, 2025. 4

[19] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun 2018. 3

[20] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1

[21] Lei Han, Tian Zheng, Lan Xu, and Lu Fang. Occuseg: Occupancy-aware 3d instance segmentation. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2937–2946, 2020. 3

[22] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*, 2022. 3, 4, 5

[23] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15979–15988, 2022. 3

[24] Ji Hou, Benjamin Graham, Matthias Nießner, and Saining Xie. Exploring data-efficient 3d scene understanding with contrastive scene contexts. *CoRR*, abs/2012.09165, 2020. 3

[25] 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 *Proceedings of the 36th International Conference on Machine Learning*, 2019. 2, 3, 4, 7, 8, 23

[26] 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. In *International Conference on Learning Representations*, 2022. 2, 4, 5, 7, 8, 23

[27] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. *CoRR*, abs/1911.11236, 2019. 3

[28] Zhiqiang Hu, Yihuai Lan, Lei Wang, Wanyu Xu, Ee-Peng Lim, Roy Ka-Wei Lee, Lidong Bing, and Soujanya Poria. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. *arXiv preprint arXiv:2304.01933*, 2023. 4

[29] Md Amirul Islam*, Sen Jia*, and Neil D. B. Bruce. How much position information do convolutional neural networks encode? In *International Conference on Learning Representations*, 2020. 6

[30] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention, 2021. 9

[31] 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 (ECCV)*, 2022. 3

[32] Damjan Kalajdzievski. A rank stabilization scaling factor for fine-tuning with lora, 2023. 4

[33] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2023. 3, 7

[34] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. Stratified transformer for 3d point cloud segmentation. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8490–8499, 2022. 3, 4

[35] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3744–3753. PMLR, 09–15 Jun 2019. 9

[36] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021. 2, 3, 5, 6

[37] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2021. 2, 3, 5, 6, 7, 8, 23

[38] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 3

[39] Dingkang Liang, Tianrui Feng, Xin Zhou, Yumeng Zhang, Zhikang Zou, and Xiang Bai. Parameter-efficient fine-tuning in spectral domain for point cloud learning. *arXiv preprint arXiv:2410.08114*, 2024. 4, 23

[40] Haokun Liu, Derek Tam, Muqeeth Mohammed, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. 4

[41] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, Jan. 2023. 3

[42] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. Dora: weight-decomposed low-rank adaptation. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024. 4

[43] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland, May 2022. Association for Computational Linguistics. 3

[44] Yunze Liu, Li Yi, Shanghang Zhang, Qingnan Fan, Thomas A. Funkhouser, and Hao Dong. P4contrast: Contrastive learning with pairs of point-pixel pairs for RGB-D scene understanding. *CoRR*, abs/2012.13089, 2020. 3

[45] Ze Liu, Han Hu, Yue Cao, Zheng Zhang, and Xin Tong. A closer look at local aggregation operators in point cloud analysis. *ECCV*, 2020. 3

[46] Dening Lu, Qian Xie, Mingqiang Wei, Kyle Gao, Linlin Xu, and Jonathan Li. Transformers in 3d point clouds: A survey, 2022. 1

[47] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`, 2022. 7

[48] Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4

[49] OpenAI. Hello gpt-4o, 2024. 1, 3

[50] Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023. 1, 3

[51] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics, 2021. 1, 3

[52] Clifton Poth, Hannah Sterz, Indraneil Paul, Sukannya Purkayastha, Leon Engländer, Timo Imhof, Ivan Vulić, Sebastian Ruder, Iryna Gurevych, and Jonas Pfeiffer. Adapters: A unified library for parameter-efficient and modular transfer learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 149–160, Singapore, Dec. 2023. Association for Computational Linguistics. 7

[53] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016. 3

[54] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *CoRR*, abs/1706.02413, 2017. 3

[55] Zekun Qi, Runpei Dong, Guofan Fan, Zheng Ge, Xiangyu Zhang, Kaisheng Ma, and Li Yi. Contrast with reconstruct: Contrastive 3d representation learning guided by generative pretraining. In *International Conference on Machine Learning (ICML)*, 2023. 22, 23

[56] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 3

[57] Haibo Qiu, Baosheng Yu, and Dacheng Tao. Collect-and-distribute transformer for 3d point cloud analysis. *arXiv preprint arXiv:2306.01257*, 2023. 3

[58] Qwen. Qwen2.5 technical report, 2025. 1

[59] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 1, 3

[60] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 506–516, Red Hook, NY, USA, 2017. Curran Associates Inc. 3

[61] David Rozenberszki, Or Litany, and Angela Dai. Language-grounded indoor 3d semantic segmentation in the wild. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 3, 6, 7, 22, 23

[62] Jan-Martin O. Steitz and Stefan Roth. Adapters strike back. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 23449–23459. IEEE, June 2024. 3, 4

[63] Hongyu Sun, Yongcai Wang, Wang Chen, Haoran Deng, and Deying Li. Parameter-efficient prompt learning for 3d point cloud understanding. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. 4

[64] Yiwen Tang, Ray Zhang, Zoey Guo, Xianzheng Ma, Bin Zhao, Zhigang Wang, Dong Wang, and Xuelong Li. Point-peft: Parameter-efficient fine-tuning for 3d pre-trained models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 5171–5179, 2024. 4

[65] Lyne Tchapmi, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. *2017 International Conference on 3D Vision (3DV)*, Oct 2017. 3

[66] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J. Guibas. Kpconv: Flexible and deformable convolution for point clouds. *CoRR*, abs/1904.08889, 2019. 3

[67] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. 3

[68] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 3, 4

[69] Peng-Shuai Wang. Octformer: Octree-based transformers for 3d point clouds. *ACM Transactions on Graphics*, 42(4):1–11, July 2023. 3, 4

[70] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-CNN: Octree-based Convolutional Neural Networks for 3D Shape Analysis. *ACM Transactions on Graphics (SIGGRAPH)*, 36(4), 2017. 3

[71] Song Wang, Xiaolu Liu, Lingdong Kong, Jianyun Xu, Chunyong Hu, Gongfan Fang, Wentong Li, Jianke Zhu, and Xinchao Wang. Pointlora: Low-rank adaptation with token selection for point cloud learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6605–6615, 2025. 4, 23

[72] Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. Adamix: Mixture-of-adaptations for parameter-efficient model tuning. *arXiv preprint arXiv:2205.12410*, 2022. 3

[73] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph CNN for learning on point clouds. *CoRR*, abs/1801.07829, 2018. 3

[74] Zirui Wang, Zihang Dai, Barnabas Poczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 11285–11294. IEEE, June 2019. 7

[75] Wenxuan Wu, Zhongang Qi, and Fuxin Li. Pointconv: Deep convolutional networks on 3d point clouds. *CoRR*, abs/1811.07246, 2018. 3

[76] Xiaoyang Wu, Daniel DeTone, Duncan Frost, Tianwei Shen, Chris Xie, Nan Yang, Jakob Engel, Richard Newcombe, Hengshuang Zhao, and Julian Straub. Sonata: Self-supervised learning of reliable point representations. In *CVPR*, 2025. 1, 3, 6, 7, 8, 9, 22, 23

[77] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. In *CVPR*, 2024. 1, 3, 4, 7, 8, 22, 23

[78] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. In *NeurIPS*, 2022. 3, 4

[79] Xiaoyang Wu, Zhuotao Tian, Xin Wen, Bohao Peng, Xihui Liu, Kaicheng Yu, and Hengshuang Zhao. Towards large-scale 3d representation learning with multi-dataset point prompt training. In *CVPR*, 2024. 1, 6, 7, 8, 22

[80] Xiaoyang Wu, Xin Wen, Xihui Liu, and Hengshuang Zhao. Masked scene contrast: A scalable framework for unsupervised 3d representation learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9415–9424. IEEE, June 2023. 3, 22, 23

[81] Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. ReFT: Representation finetuning for language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 4

[82] Saining Xie, Jiatao Gu, Demi Guo, Charles R. Qi, Leonidas J. Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. *CoRR*, abs/2007.10985, 2020. 3

[83] Y. Xie, J. Tian, and X. X. Zhu. Linking points with labels in 3d: A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*, 8(4):38–59, 2020. 1

[84] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023. 3, 6, 7

[85] 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 Trans. Graph.*, 35(6), Nov. 2016. 4, 22, 23

[86] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 3320–3328, Cambridge, MA, USA, 2014. MIT Press. 7

[87] Yaohua Zha, Jinpeng Wang, Tao Dai, Bin Chen, Zhi Wang, and Shu-Tao Xia. Instance-aware dynamic prompt tuning for pre-trained point cloud models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 4

[88] Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*, 2023. 4

[89] Shaochen Zhang, Zekun Qi, Runpei Dong, Xiuxiu Bai, and Xing Wei. Positional prompt tuning for efficient 3d representation learning, 2024. 4

[90] Zhi Zhang, Qizhe Zhang, Zijun Gao, Renrui Zhang, Ekaterina Shutova, Shiji Zhou, and Shanghang Zhang. Gradient-based parameter selection for efficient fine-tuning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 28566–28577. IEEE, June 2024. 3

[91] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer, 2021. 3, 4

[92] Xin Zhou, Dingkang Liang, Wei Xu, Xingkui Zhu, Yihan Xu, Zhikang Zou, and Xiang Bai. Dynamic adapter meets prompt tuning: Parameter-efficient transfer learning for point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14707–14717, 2024. 4

[93] Yaoming Zhu, Jiangtao Feng, Chengqi Zhao, Mingxuan Wang, and Lei Li. Counter-interference adapter for multilingual machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, page 2812–2823. Association for Computational Linguistics, 2021. 3

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: no theoretical proof needed

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

Justification: all publicly available

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: will open-source for reproduction

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: publicly available

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: following the established protocols

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: as indicated by the public codebase

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: follow

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: not related

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: not related

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All are open-source and are properly cited

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: will be integrated into open-sourced codebase

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: not related

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:not related

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: not related

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.

Supplementary

In this supplementary, we provide more materials as follows,
Sec. A details the implementation, training, and inference;
Sec. B presents additional experiments and analyses with tight parameter budgets
Sec. C offers more qualitative visualizations.

A Details of Implementation, Training, and Inference

Our implementation is based on the open-source codebase Pointcept (here) and follows the official implementations for Sonata [76], PPT [79], as well as PTv3 [77].

Leveraging the parameter efficiency of our method, we train on a single 4090 GPU for much fewer epochs to obtain the reported performance. For example, we train on ScanNet for 100 epochs, in contrast to the 800 epochs in the released Sonata configuration. Our code is available at https://github.com/LiyaoTang/GEM.

Tab. 7 summarizes empirical latency and memory usage. We intentionally refrain from deployment-specific optimizations such as LoRA weight merging. Although the global attention and local convolution modules introduce extra cost, the runtime and memory footprint stay within the same order of magnitude as other PEFT baselines.

B More Experiments and Analysis

In spite of the promising results shown in Sec. 3, we suggest that it can better demonstrate the full potential of PEFT methods under broader and more challenging settings.

PEFT for shape analysis. While GEM is motivated by PEFT for large-scale 3D scenes, most existing 3D PEFT methods target object-level understanding, as discussed in Sec. 2. To assess cross-domain generalization and compare against these existing 3D-specific PEFT methods, we further evaluate GEM on 3D shape datasets, ShapeNetPart [85]. As shown in Tab. 8, GEM remains competitive and achieves the best performance. We also observe that common backbones for shape analysis, such as ReCon [55], attach large MLP heads that account for approximately 20% of all parameters, rendering fine-tuning inefficient. To broaden the applicability of backbone models for 3D shapes, an interesting direction for future work is to develop architectures with more parameter-efficient heads that are better suited to fine-tuning.

PEFT for 3D convolutional networks. While our primary experiments apply GEM to state-of-the-art backbones, mainly transformer models, we also verify its generality on convolutional architectures. In particular, we integrate GEM into SparseUNet [10], a 3D convolutional model, following the MSC [80] protocol to pre-train on ScanNet [12] and fine-tune on ScanNet200 [61]. As reported in Tab. 9, GEM improves performance by +4.6 mIoU over linear probing and outperforms other PEFT baselines, demonstrating robustness beyond transformer models. We notice the gains are narrower than those on transformer backbones, largely due to the limited capacity of convolutions, as also indicated by a near-collapse linear probing performance in this setting.

PEFT under tight budgets. To stress parameter efficiency, we constrain all methods to the same limited learnable-parameter budgets, including enforcing rank $r = 1$, allowing 0.1% parameters, and 1% parameters. As reported in Tab. 10, GEM delivers consistently higher accuracy, whereas several baselines fail when the budget becomes extremely stringent.

C More Visualizations

We provide more qualitative results regarding the attention map generated by our latent tokens in Fig. 4. Despite shared across the stages, the latent tokens appear to focus on different parts of the scenes, providing different scene context to the backbone models at different stages.

Methods	Params.	Memory	Latency
Base model			
Sonata (full.) [76]	124.8M	8.2G	61.8ms
Sonata (ft.)	108.5M	6.4G	50.2ms
PEFT methods			
Sonata (lin.)	0.02M	**6.4G**	**50.2ms**
+Adapter [25]	2.8M	6.7G	50.6ms
+LoRA [26]	1.9M	7.9G	52.8ms
+Prompt Tunning [37]	5.5M	7.3G	51.5ms
+**GEM (ours)**	1.8M	7.1G	57.8ms

Table 7. **Inference efficiency of PEFT methods.** We benchmark with the batch size fixed to 1.

Shape-Part Seg.	Params	ShapeNetPart [85]	
Methods	Learn.	Cls. mIoU	Inst. mIoU
Full fine-tuning			
ReCon (ft.) [55]	27.06	84.52	86.1
PEFT methods			
ReCon (lin.) [55]	5.23M	83.06	85.2
+PointLoRA [71]	5.63M	83.98	85.4
+PointGST [39]	5.59M	83.98	85.8
+**GEM (ours)**	5.58M	**84.02**	**85.8**

Table 8. **Generalizing to 3D shapes.**

Tight Budgets	Params		ScanNet200 Val [61]		
Methods	Learn.	Pct.	mIoU	mAcc	allAcc
Training from scratch					
SparseUnet [10]	39.2M	100%	25.0	32.9	80.4
Full fine-tuning					
SparseUnet (ft.) [80]	0.02M	0.05%	32.0	41.6	82.3
PEFT methods with rank=1					
SparseUnet (lin.)	0.02M	0.05%	1.5	2.5	53.6
+BitFit [6]	0.03M	0.07%	4.8	6.9	62.7
+Adapter [25]	0.6M	1.5%	9.5	13.4	69.5
+LoRA [26]	0.8M	2.0%	13.2	17.6	74.7
+**GEM (ours)**	0.6M	1.5%	**15.2**	**22.9**	**75.6**

Table 9. **Generalizing to convolutional networks.**

Tight Budgets	Params		ScanNet Val [12]		
Methods	Learn.	Pct.	mIoU	mAcc	allAcc
Training from scratch					
PTv3 [77]	124.8M	100%	77.6	85.0	92.0
Full fine-tuning					
Sonata (full.) [76]	124.8M	100%	79.4	86.1	92.5
Sonata (ft.)	108.5M	100%	78.3	85.9	92.3
PEFT methods with rank=1					
Sonata (lin.)	0.02M	0.02%	72.5	83.1	89.7
+Adapter [25]	0.05M	0.04%	74.0	84.1	90.5
+LoRA [26]	0.05M	0.05%	42.5	55.4	75.1
+Prompt Tunning [37]	0.05M	0.05%	72.9	83.5	90.0
+**GEM (ours)**	0.07M	0.06%	**75.2**	**84.8**	**91.1**
PEFT methods with 0.1% params.					
Sonata (lin.)					
+Adapter [25]	0.1M	0.1%	75.1	84.7	91.1
+LoRA [26]	0.1M	0.1%	75.0	84.4	91.1
+Prompt Tunning [37]	0.1M	0.1%	73.5	83.9	90.3
+**GEM (ours)**	0.1M	0.1%	**76.5**	**85.5**	**91.6**
PEFT methods with 1% params.					
Sonata (lin.)					
+Adapter [25]	1.1M	1.0%	76.6	85.3	91.7
+LoRA [26]	1.1M	1.0%	76.7	85.6	91.7
+Prompt Tunning [37]	1.1M	1.0%	73.8	84.2	90.4
+**GEM (ours)**	1.1M	1.0%	**78.2**	**86.3**	**92.2**

Table 10. **Performance with tight budgets.**

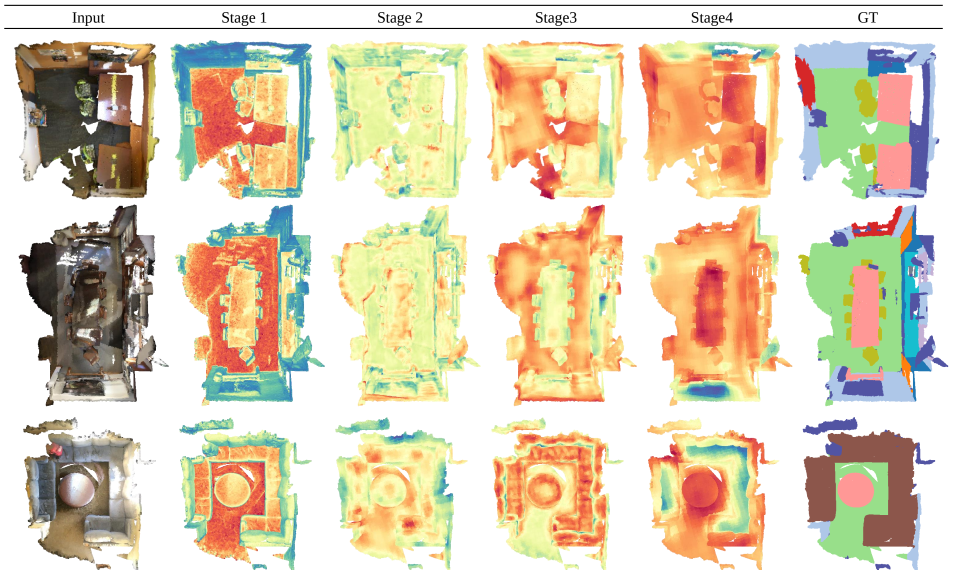

Figure 4. Attention maps of our latent tokens in context adapter.