PMA: Towards Parameter-Efficient Point Cloud Understanding via Point Mamba Adapter (Supplementary Material)

1. Implementation Details

We utilized the officially provided pre-trained Point-BERT [6], Point-MAE [3], and PointGPT-L [1] models as our point cloud pre-training models. For a fair comparison, in line with previous studies [3, 6–9], we evaluated the effectiveness of our Point Mamba Adapter (PMA) on tasks such as classification, part segmentation, and few-shot learning. The training configurations for all downstream tasks, including learning rate, optimizer, loss function, and data augmentation strategies, were kept consistent with the default settings of the original works. All experiments involving Point-BERT and Point-MAE were conducted on a single RTX 3090 GPU (24GB), while experiments with PointGPT-L were conducted on a single A800 GPU.

2. Additional Ablation Study

2.1. Layer-Specific G2PG

In our experiments, the Geometry-constrained Gate Prompt Generator (G2PG) produces gate prompts to adjust the output matrix of the Mamba Adapter and generates index sequences for reordering layer tokens. This facilitates more effective integration of spatial information, thereby enhancing the model's ability to process point cloud data.

By default, we use a shared G2PG across different layers to enable efficient fine-tuning, thus minimizing the introduction of additional parameters. In fact, we can also use a layer-specific G2PG, which would increase the parameter count to some extent. In this section, we conduct further experiments to investigate the impact of this approach on performance.

	# TP	OBJ-BG	OBJ-ONLY
layer-shared	1.05	91.05	90.89
layer-specific	1.71	91.48	91.05

Table 1. The effect of layer-specific G2PG.

Table 1 reports our experimental results. By introducing the layer-specific G2PG, we observed a certain performance improvement on both the OBJ-BG and OBJ-ONLY variants of the ScanObjectNN [4] dataset. However, the improvement was not substantial, and this layer-specific approach also led to an increase in both parameter count and computational complexity. Therefore, in our experiments, we default to using the layer-shared G2PG to improve efficiency. Of course, the layer-specific G2PG serves as a variant of our method.

2.2. The Effect of Geometric Constraints in G2PG

	# TP	OBJ-BG	OBJ-ONLY
MLP	1.03	90.36	90.19
Attention	1.62	90.53	90.19
Mamba	1.64	90.71	90.53
Geometry-constrained MLP	1.05	91.05	90.89

Table 2. The Effect of Geometric Constraints in G2PG

We further analyzed the importance of applying geometric constraints in G2PG. By default, we utilized a geometry-constrained MLP architecture to generate prompts. Additionally, we examined the impact of generating prompts using only MLP, a Self-Attention-based architecture [5], and the Mamba architecture [2] on the final performance. When using only MLP, no token interaction occurs, providing no guidance for the generated prompts. In contrast, the Self-Attention mechanism enforces feature constraints through attention-based interactions, while the Mamba architecture relies on sequential dependencies to impose constraints.

Table 2 reports our experimental results and the experimental results demonstrate that the combination of geometric constraints and an MLP architecture achieves the best performance, followed by the Mamba-based architecture, the Self-Attention-based architecture, and finally the plain MLP. This can be attributed to that the generated prompts influence the Mamba Adapter's output gates, guiding spatial feature perception. Geometric constraints + MLP yield the best results by embedding explicit spatial guidance, enabling precise gating adjustments. Mamba-based prompts leverage sequential dependencies for hierarchical modeling but are less effective in fine-grained geometry. Self-Attention-based prompts emphasize global patterns but lack spatial specificity, leading to suboptimal gating. Plain MLP

prompts lack interaction and spatial awareness, offering minimal guidance. Overall, geometric constraint integration proves most effective for adaptive and precise gating.

2.3. The Effect of Different Intermediate Layer Features

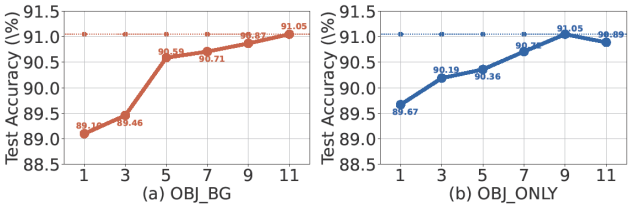

Figure 1. The effect of different intermediate layer features.

We further analyzed the impact of using intermediate features from different layers on the final performance. In our experiments, we defaulted to using the intermediate features from the first N-1 layers (N=12) as input to the downstream task head. We then investigated the performance when using the features from the first 1, 3, 5, up to 11 layers to explore how the number of intermediate features influences the final performance.

Figure 1 shows our experimental results, which indicate that as the number of intermediate layers increases, performance consistently improves in both the OBJ-BG and OBJ-ONLY variants. This trend suggests that the interaction of more intermediate layer features contributes to further gains in performance. The only exception is the performance at the 9-th layer of OBJ-ONLY, which surpasses the performance of the 11-th layer. This anomaly reinforces our hypothesis that intermediate layer features may contain richer information than the final layer features, further supporting our findings.

2.4. Comparison with Scale-up Segmentation Head

We compare full fine-tuning (FFT), Vanilla Head Tuning (HT), scale up Head Tuning (Scale HT), and our PMA with different heads. As shown in Table 3, FFT significantly outperforms HT in mIoU_I by 1.1%. Further scaling up the head parameters also improves performance, highlighting the effectiveness of more trainable parameters. However, compared to Scale HT (only scaled Head), PMA (PMA+G2PG+Vanilla Head) achieves a more substantial improvement (0.9% in mIoU_I) with fewer parameters by leveraging intermediate feature fusion, demonstrating the superiority of our approach. Additionally, PMA combined with the scaled head achieves further improvements.

2.5. The Effect of Different Fusion Methods

As shown in Table 4, we compare different PEFT methods (IDPT, DAPT) and various order-agnostic fusion methods (Pooling, MLPs, GCN) with our PMA in terms of efficiency and performance. Our PMA achieves the optimal efficiency-performance trade-off, benefiting from the linear complexity of Mamba during inference.

Tuning Strategy	#TP (M)	mIoU_C	mIoU_I
FFT (baseline)	27.06	84.2	86.1
Head Tune (HT)	5.24	83.1	85.0
Scale HT	5.71	83.3	85.2
IDPT	5.69	83.8	85.7
DAPT	5.65	84.0	85.7
PMA	5.64	84.0	86.1
PMA+Scale HT	6.10	84.4	86.1

Table 3. The results of scale-up segmentation head.

Fusion Methods	#TP (M)	GFLOPs	ScanObjectNN
IDPT	1.7	7.3	84.94
DAPT	1.1	5.2	85.08
Pooling	0.4	4.9	79.39
MLPs	1.0	5.8	83.78
GCN	1.3	11.7	84.87
PMA	1.1	6.8	86.43

Table 4. The effect of different fusion methods.

2.6. The Effect of Layer-wise Reorder

Gate Prompt	Order	OBJ-BG	OBJ-ONLY
✘	✘	89.69	88.57
✘	fixed	89.82	88.74
✘	layer-wise	90.08	89.26
✔	✘	90.62	89.69
✔	fixed	90.87	90.17
✔	layer-wise	91.05	90.89

Table 5. The effect of layer-wise reorder.

The ordered point cloud patches generated by G2PG follow spatial locality to some extent. The different orderings at each layer can be regarded as order of the complete 3D geometry from different direction. By integrating multiple direction, we achieve a more comprehensive understanding of the 3D structure. Therefore, our layer-wise sorting is to enhance the model's overall perception of 3D point cloud spatial relationships by incorporating the orderings from different directions across multiple feature layers, thereby addressing the isotropic nature of 3D space. We also provide a more detailed comparative experiment on different orderings on Table 5. "fixed" refers to using only the first-layer index generated by G2PG across all layers.

References

[1] Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. 2024. 1

[2] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. 1

[3] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Tel Aviv, Israel, 2022. 1

[4] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1588–1597, Seoul, Korea, 2019. 1

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, page 30, Long Beach, CA, USA, 2017. 1

[6] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19313–19322, New Orleans, Louisiana, USA, 2022. 1

[7] 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 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14161–14170, Paris, France, 2023.

[8] Yaohua Zha, Huizhen Ji, Jinmin Li, Rongsheng Li, Tao Dai, Bin Chen, Zhi Wang, and Shu-Tao Xia. Towards compact 3d representations via point feature enhancement masked autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, VANCOUVER, CANADA, 2024.

[9] Yaohua Zha, Naiqi Li, Yanzi Wang, Tao Dai, Hang Guo, Bin Chen, Zhi Wang, Zhihao Ouyang, and Shu-Tao Xia. Lcm: Locally constrained compact point cloud model for masked point modeling. *arXiv preprint arXiv:2405.17149*, 2024. 1