GFT: Graph Feature Tuning for Efficient Point Cloud Analysis

Supplementary Material

Learnable Prompt Length	#TP (M)	OA(%)
10	0.72	91.91
25	0.72	91.74
50	0.73	92.56
100	0.75	92.05

Table 1. Analysis of prompt length for GFT analysis. Prompt length beyond 50 hurts the performance.

EdegeConv KNN	Training Complexity		OA(%)
	Epoch Time (secs)	Memory (GB)	
5	15.5	6.9	91.77
10	16.5	7.1	91.67
20	18.0	8.7	92.56
40	23.5	10.4	92.22

Table 2. Analysis of different dimensions of the EdgeConv feature pyramid. Increasing the neighbours in KNN imposes higher training time and memory complexities.

8. Additional Ablation Study

Similar to the ablation study in the main paper, we use Point-MAE as the pretrained backbone and OBJ_BG as the downstream classification task; each experimentation is subjected to five different seed values.

Prompt Length vs. Performance. As shown in Table 1, we evaluate the sensitivity of our method to prompt length by experimenting with four different values: $\{10, 25, 50, 100\}$. The optimal performance is achieved with a prompt length of 50, which we use for our reported results. Increasing the prompt length does not necessarily yield better performance. Unlike patch tokens, these prompts lack locality information but offer greater degrees of freedom in graph construction. However, an excessive number of prompts may introduce unnecessary edges, which could negatively impact downstream tasks.

EdgeConv KNN The performance of GFT is also influenced by the number of neighbors used for message passing. Changing the number of neighbors does not affect the trainable parameters but changes the time and computational complexities. To analyze this, we conduct an ablation study with different values of $k \in \{5, 10, 20, 40\}$ in KNN, as shown in Table 2. Given 50 prompt tokens and 128 patch tokens, we have a total of 178 tokens and message passing with the 20 nearest neighbors yields the best results. While increasing the number of neighbors raises both time and memory complexity, it does not lead to further improvements in performance.

Feature Pooling Techniques			#TP (M)	OA(%)
T_{cls}	Patch Pooling	Prompt Pooling		
✓			0.54	90.29
✓	✓		0.64	91.98
✓	✓	✓	0.73	92.56

Table 3. Analysis of token projections for classification task. The combinationion of all of the feature pooling gives the best result.

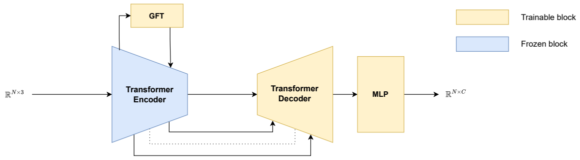

Figure 1. **Overall architecture of the segmentation model.** The segmentation head utilizes intermediate features from the encoder layers, processes them through transformer decoders, and finally passes them to an MLP layer to predict the class labels.

Tokens Projections for Task Head We explore combinations of three options for projecting features from Point Transformers for classification: T_{cls}, patch pooling, and prompt pooling. Previous fine-tuning methods primarily relied on T_{cls} and patch pooling as projections for the task head [1–5]. However, GFT has learnable prompts where task-specific features are accumulated. We achieve the best results by leveraging all three projection options, as shown in Table 3.

9. Training Details

Table 4 provides the comprehensive training details for our method across different classification and segmentation tasks. Classification models have been optimized using cross-entropy loss, while segmentation models utilize negative log-likelihood loss. We follow a similar training approach to earlier PEFT methods [4, 5].

10. Architecture for Segmentation Head

Our part segmentation task head consists of two main components: (1) a transformer decoder, composed of stacked

Configuration	Classification			Segmentation (ShapeNetPart)
	ScanObjectNN	ModelNet	ModelNet Few-shot	-
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning rate (LR)	5e-4	5e-4	1e-3	2e-4
LR scheduler	Cosine	Cosine	Cosine	Cosine
Warmup LR	1e-6	1e-6	1e-6	1e-6
Warmup epochs	10	10	10	10
Minimum LR	1e-6	1e-6	1e-6	1e-6
Weight decay	5e-2	5e-2	1e-4	5e-2
Training epochs	300	300	150	300
Batch size	32	32	32	32
Number of points	2048	1024	1024	2048
Number of patches/tokens	128	64	64	128
Point patch size	32	32	32	32
GFT Configurations				
Learnable prompt length	50	50	50	50
EdgeConv KNN size	20	20	20	20
EdgeConv feat dims.	$[64, 64, 64, 64]$	$[64, 64, 64, 64]$	$[64, 64, 64, 64]$	$[64, 64, 64, 64]$
EdgeConv FFN dim	256	256	256	256
Cross-attention dim	32	32	32	32
Cross-attention heads	2	2	1	2

Table 4. Training details for the fine-tuning tasks.

self-attention blocks, which receives input from the encoder representations through skip connections and down-projection layers; and (2) a multi-layer perceptron (MLP) that outputs class predictions for each point, producing a tensor of shape $\mathbb{R}^{N\times C}$, where N is the number of points and C is the number of classes as given in Figure 1.

11. Vizualization of Attention Maps

As shown in the Figure 2, during fine-tuning of the pre-trained backbones, we analyze the attention maps from patch tokens to the CLS token (T_{cls}). When the pretrained weights are kept frozen, fine-tuning the task-level graph features guides the attention to consistently focus on discriminative regions. For example, attention maps for cones highlight the apex, while those for chairs emphasize the backrest. In contrast, the pretrained model often struggles to identify a consistent region of interest—sometimes focusing on one part of the object, and other times on another—indicating a lack of clear task-specific understanding.

References

[1] Runpei Dong, Zekun Qi, Linfeng Zhang, Junbo Zhang, Jianjian Sun, Zheng Ge, Li Yi, and Kaisheng Ma. Autoencoders as cross-modal teachers: Can pretrained 2d image transformers help 3d representation learning? In *The Eleventh International Conference on Learning Representations*, 2023. 1

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

[3] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19313–19322, 2022.

[4] 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 *ICCV*, 2023. 1

[5] 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 *CVPR*, 2024. 1

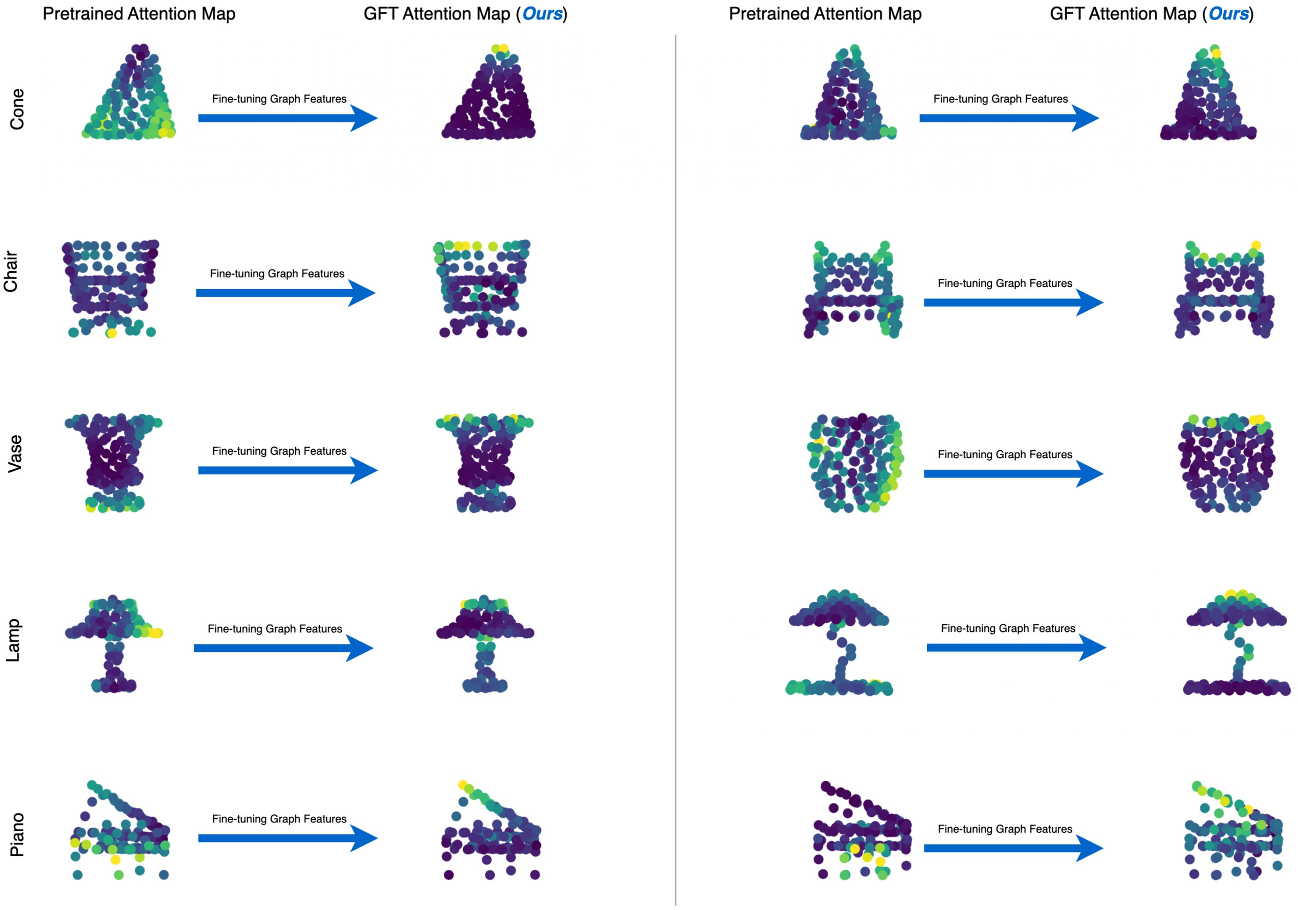

Figure 2. Fine-tuning leads to a shift in attention, redirecting focus from arbitrary regions to task-relevant, discriminative parts of the objects—such as the backrest for chairs, apex for cones, mouth for vases, and fallboard for pianos. In contrast, the pretrained model exhibits uncertainty, with attention scattered across different parts.