Instance-aware Dynamic Prompt Tuning for Pre-trained Point Cloud Models Supplementary File

Yaohua Zha[1*] Jinpeng Wang[1*] Tao Dai[2†] Bin Chen[3] Zhi Wang[1] Shu-Tao Xia[4]

Tsinghua Shenzhen International Graduate School, Tsinghua University[1]
College of Computer Science and Software Engineering, Shenzhen University[2]
Harbin Institute of Technology, Shenzhen[3]
Research Center of Artificial Intelligence, Peng Cheng Laboratory[4]
{chayh21,wjp20}@mails.tsinghua.edu.cn

1. More Experimental Analysis

1.1. Number of Prompt Tokens

In this section, we investigate the impact of prompt numbers in IDPT on classification tasks. By default, we use three layers of EdgeConv [2] and one layer of MLP to extract the semantic information $\boldsymbol{E_P} \in \mathbb{R}^{\boldsymbol{m}\times\boldsymbol{d}}$ from all patches and then use max pooling along the feature dimension to aggregate the semantic information of all patches to generate prompt $\boldsymbol{P_{N-1}} \in \mathbb{R}^{1\times\boldsymbol{d}}$.

To generate multiple representative prompts, we replace the max pooling operation along the feature dimension with a top-K operation, resulting in K prompts $\boldsymbol{P'_{N-1}} \in \mathbb{R}^{\boldsymbol{K}\times\boldsymbol{d}}$. We then aggregate $\boldsymbol{P'_{N-1}}$, $\boldsymbol{c}_{N-1}$, and $\boldsymbol{E}_{N-1}$ and feed them to the last transformer layer $\boldsymbol{f}_N$.

$$[\boldsymbol{c}_N; \boldsymbol{P}'_N; \boldsymbol{E}_N] = f_N([\boldsymbol{c}_{N-1}; \boldsymbol{P}'_{N-1}; \boldsymbol{E}_{N-1}]). \quad (1)$$

For the classification head, we perform max pooling along the feature dimension of $\boldsymbol{P}'_N$ to obtain $\boldsymbol{P}_N \in \mathbb{R}^{1\times d}$ as prompt-related input.

We analyze the impact of different prompt numbers on classification tasks. Figure 1 presents the experimental results on two variants of ScanObjectNN. The results indicate that simply increasing the prompt number does not contribute to performance gain. Therefore, we only set a single prompt in IDPT to improve efficiency.

1.2. Insert Independent Prompt Generation Modules to All Layers

In Figure 4 of the main paper, we have demonstrated the effect of inserting prompts into multiple layers of the pre-trained point cloud model. Note that we share the prompt generation module among multiple layers in Figure 4 in the spirit of parameter-efficient tuning. Nevertheless, it would be interesting to see the results of inserting independent prompt generation modules into different layers. In particular, here we provide the results of all-layer insertion, as shown in Table 1. The results indicate that incorporating a parameter-independent prompt generation module at every layer only brings marginal improvement with a significant increase of trainable parameters, deviating from the goal of parameter-efficient tuning. Regarding the empirical observations in Figure 4 of the main paper and Table 1, we only insert the dynamic prompt generation module into the last layer of the pre-trained model.

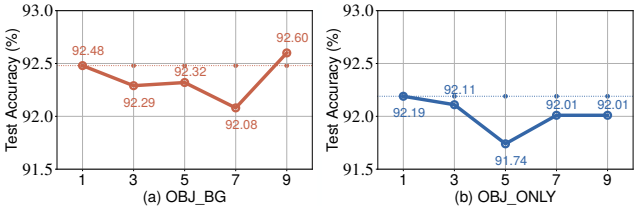

Figure 1. Effect of different numbers of prompts.

Trainable Parameters Type	#TP (M)	OBJ_BG	OBJ_ONLY
1 PM + Head	1.70	92.48	92.19
12 PM + Head	16.34	**92.60**	**92.22**

Table 1. Effect of inserting independent prompt modules to all layers. PM indicates the dynamic prompt generation module.

1.3. Input of Downstream Task Head

We conducted an investigation on the impact of the downstream task head's input features, which include the prompt token, the CLS token, and the max pooling of point

*These authors contributed equally to this work.
†Corresponding author. (✉ daitao.edu@gmail.com)

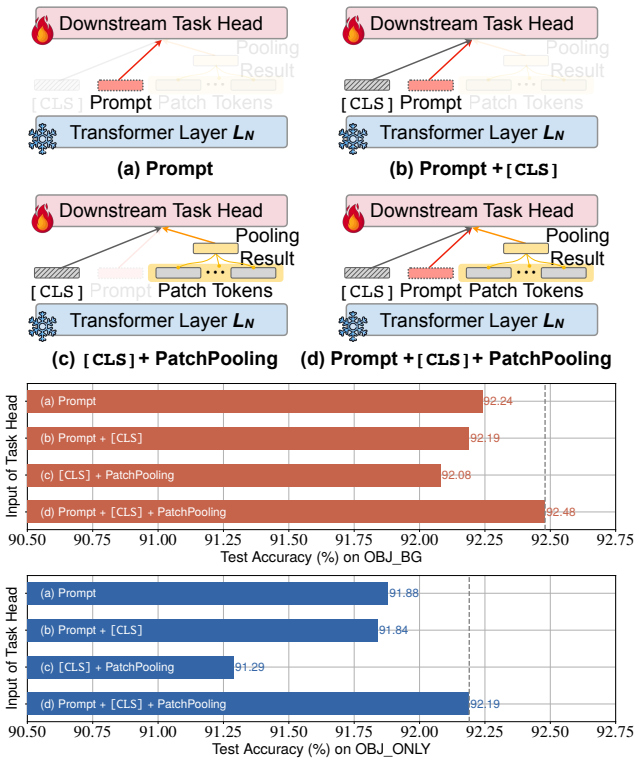

Figure 2. Effect of different head inputs.

patch tokens. The results are shown in Figure 2. We found that the highest performance was achieved when all three features were included (d): prompt token, CLS, and point patch tokens. When using only the dynamic prompt token (b), the performance was still strong and only second to the previous case. However, removing our prompt token (c) resulted in a slight decline in performance. These results indicate that the dynamic prompt token plays a critical role in guiding the downstream task fitting, as it contains specific and semantic information about the task data.

Although omitting the prompt feature in the head results in a performance decline, there is still a significant improvement when compared to traditional static prompts, as shown in Table 6. This suggests that our dynamic prompt is effective in aligning with different distributional data.

1.4. Compare IDPT with Full Fine-tuning

Although fine-tuning with full trainable parameters enables more flexible adaptation, it is not always an optimal solution for downstream transfer. One possible issue is over-fitting the training set, which harms the generalizability. In particular, we investigate the accuracy dynamics of different tuning strategies with Point-MAE throughout training, as shown in Figure 3. We can see IDPT outperforms fine-tuning on test sets, despite slightly inferior performance on train sets. This phenomenon demonstrates the structural flexibility of IDPT in domain adaptation and also indicates that it regularizes the model against over-fitting better. Besides, the efficacy of IDPT is not simply owed to the parameter-efficient setting, because VPT-Deep with a similar proportion of trainable parameters does not show satisfactory performance as IDPT does. Instead, the results provide evidence for the superiority of the proposed instance-aware dynamic prompt design.

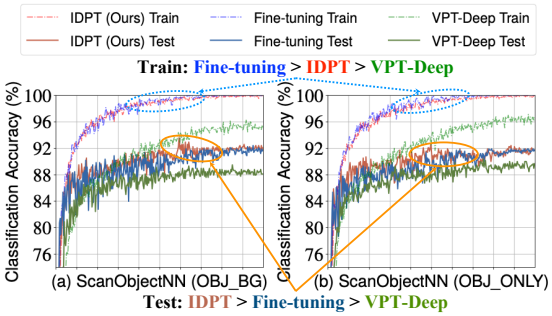

Figure 3. The convergence curves of training and testing.

1.5. Convergence of Different Tuning Strategies

In this section, we study how the performances of different tuning strategies change in the whole training process. The accuracy curves of fine-tuning, VPT, and IDPT on two datasets (*i.e.*, ModelNet40 and ScanObjectNN) are illustrated in Figure 4.

As shown in Figure 4, our IDPT strategy achieves significant improvements upon VPT. The performance of IDPT is competitive with fine-tuning on most datasets. Moreover, we can learn that IDPT yields faster convergence by incorporating prior semantic information of instances, revealing the merit of instance-aware dynamics for model adaptation.

1.6. Demonstration of Sub-modes in Real-world Point Cloud Data

Due to the limitations of scanning techniques, it is prevailing to see various kinds of missing or noisy points in real-world point clouds, corresponding to different sub-modes in the data distribution. Such inconsistent noises will threaten the robustness of prompt-based adaptation, especially for static prompt strategies like VPT [1]. Here we would like to give some point cloud samples to facilitate an intuitive understanding about *how different sub-modes look like*. Specifically, Figure 5 presents different missing types w.r.t. different categories in the ScanObjectNN dataset. We use sub_mode1 and sub_mode2 to differentiate missing types. For each scanned object, we show its projection images from three different viewpoints (*i.e.* view1, view2, and view3) to simulate stereoscopy.

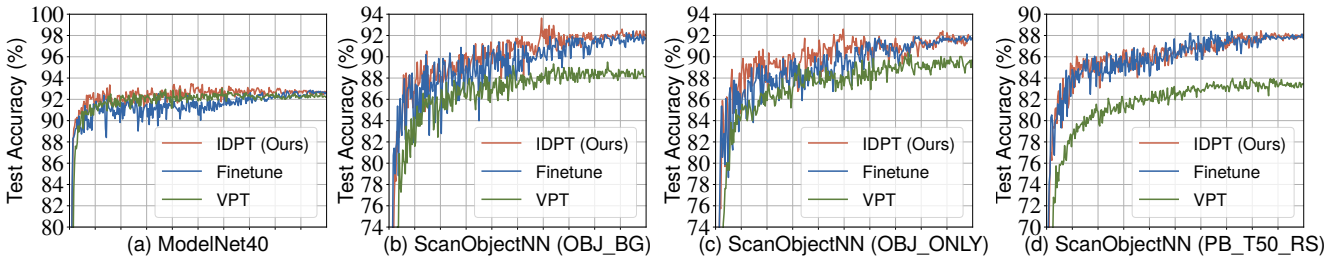

Figure 4. The classification accuracy curves of fine-tuning, VPT, and our IDPT strategy on two datasets.

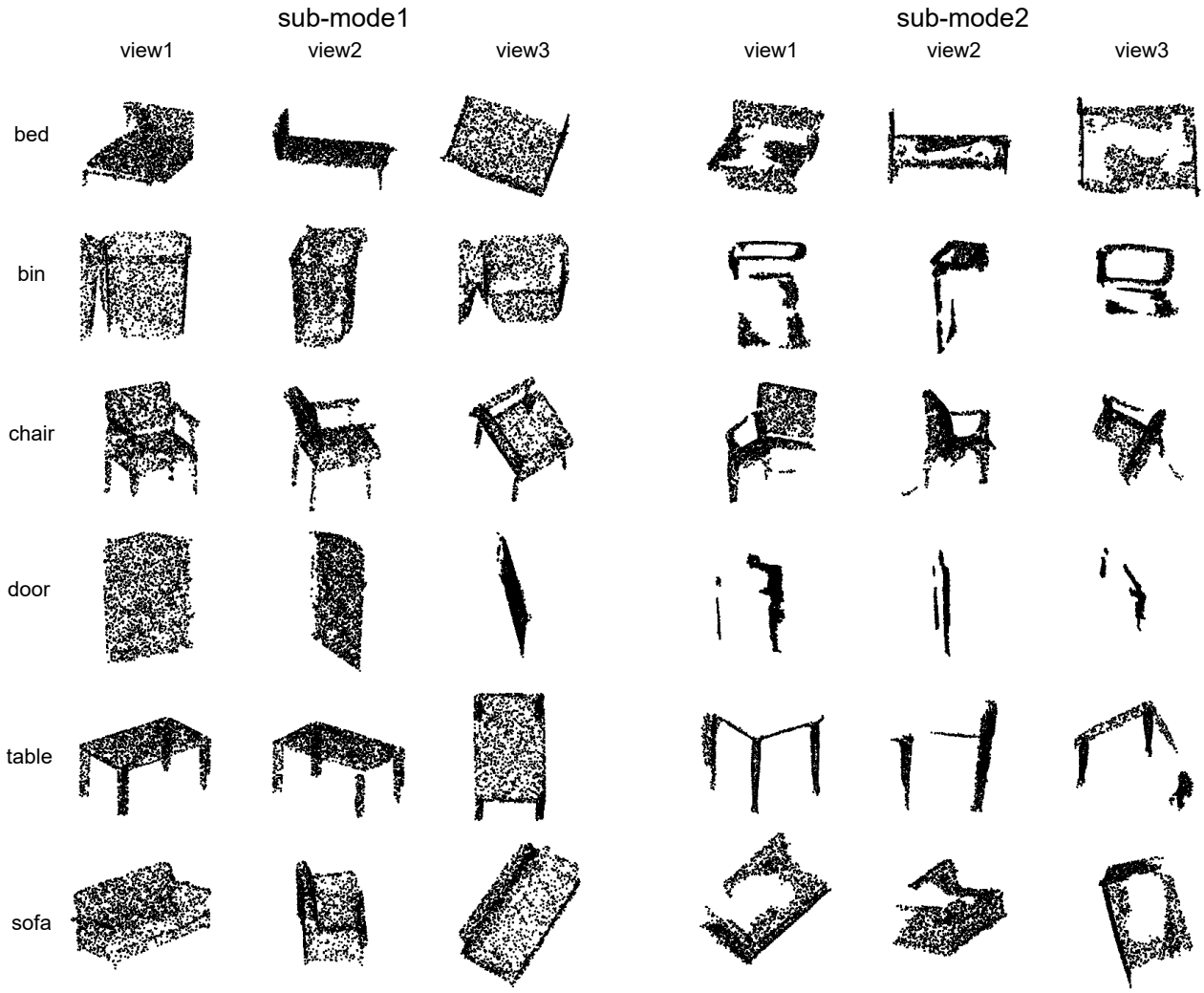

Figure 5. Different sub-modes in each category of ScanObjectNN

References

[1] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *ECCV*, 2022. 2

[2] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM ToG*, 38(5), 2019. 1