Parameter-Efficient and Memory-Efficient Tuning for Vision Transformer: A Disentangled Approach (Appendix)

Taolin Zhang[1]*, Jiawang Bai[2,3]*, Zhihe Lu[3], Dongze Lian[3], Genping Wang[4,✉], Xinchao Wang[3,✉], and Shu-Tao Xia[1,5]

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]Tencent [3]National University of Singapore
[4]Shenzhen Qiji Technology Co., Ltd.
[5]Research Center of Artificial Intelligence, Peng Cheng Laboratory
ztl23@mails.tsinghua.edu.cn; jiawangbai@tencent.com;
zhihelu.academic@gmail.com; dzlianx@gmail.com;
genpingwang@163.com; xinchao@nus.edu.sg; xiast@sz.tsinghua.edu.cn

1 Analysis on Memory Cost and Entanglement

In this section, we conduct an additional experiment to investigate the connection between memory cost and entanglement. To achieve varying degrees of entanglement, we fine-tune **only one layer** of the pre-trained model using LoRA and provide memory usage at **different positions** in Fig. 1.

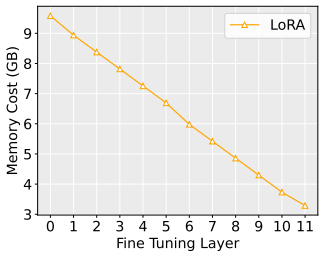


Fig. 1: Memory usage with different fine-tuning layer.

When tuning the highest layer, the degree of entanglement is minimal and the features of the lower layers remain unchanged, not engaging in gradient descent, resulting in a small memory cost. As the layer being fine-tuned becomes lower, the degree of entanglement increases and the intermediate features that participate in gradient descent become heavier, leading to a larger memory cost.

2 Analysis on Inference Cost

The inference cost of the model is important in some real-world applications. Compared to the original pre-trained model, SynQT, with the lightweight architectures QSM and KEM, could potentially introduce additional inference costs. Therefore, we evaluate the inference time, memory usage, and FLOPS in comparison to traditional PETL methods, and present the results along with the optimal number of tokens in Table 1.

Notably, the low inference speed and high FLOPS of VPT-Shallow [2] are attributed to its optimal number of tokens on CIFAR100 reaching up to 100. In

contrast, the inference speeds and FLOPS of BitFit [7], VPT-Deep [2], VQT [6], and SynQT are similar without any significant differences. It suggests that the important factor affecting inference speed is the number of prompt tokens rather than the additional lightweight architectures. Given that we use an extremely small number of tokens for the lightweight design of QSM, which is chosen from $\{1, 4, 16\}$, the overall inference speed of SynQT is comparable to VQT and the additional inference cost is affordable.

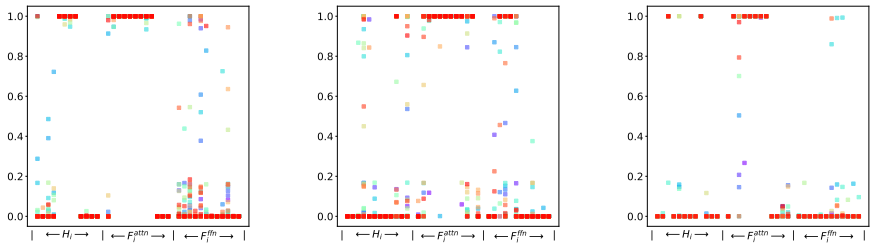
Table 1: Inference speed analysis on CIFAR100. We evaluate different methods with their optimal number of tokens on a single NVIDIA V100 32GB GPU and report the frames per second (fps).

Methods	Optimal #Tokens	Inference Speed (fps)	Memory (G)	FLOPS (G)
BitFit [7]	-	249.94	2.8	16.9
VPT-Shallow [2]	100	172.69	2.9	25.3
VPT-Deep [2]	10	245.76	2.9	17.7
VQT [6]	1	246.95	2.8	17.2
SynQT (Ours)	4	244.91	2.9	17.2

3 Analysis on Feature Importance

The classification head in SynQT is instance-aware, by adopting the feature weights conditioned on the output of the last block. To gain deeper insight into the feature selection within the classification head, we investigate the generated feature weights for different input samples. Specifically, we randomly select 50 samples and visualize the importance across 36 distinct features in Figure 2. The features are divided into 3 groups: the output of KEM H_i, the attention features F_i^{attn} , and the FFN features F_i^{ffn}, where $i \in \{1, 2, ..., 12\}$. The features within the groups are sorted by the layer index and their importance is determined by their absolute values of the feature weight. The experiments are conducted on EuroSAT, SVHN, and DMLab.

As illustrated in Figure 2, the feature selection in the classification head assigns varying weights to different samples, thereby enabling SynQT to be instance-aware and enhancing the model's performance. An interesting observation is that the importance of most features is either greater than 0.8 or less than 0.2, suggesting that the classification head distinguishes between useful and non-useful features by assigning relatively extreme values within the 0∼1 range for a given sample. Specifically, the number of features with a weight greater than 0.8 is noticeably fewer than those assigned a weight less than 0.2, thereby demonstrating the effectiveness of the feature selection in reducing information redundancy.



(a) Eurosat **(b)** SVHN **(c)** DMLab

Fig. 2: Feature importance visualization of 50 samples on different datasets. The feature weights of different samples are marked with different colors. For a given sample, SynQT tends to assign relatively extreme values within the 0∼1 range to distinguish useful and un-useful features.

Table 2: Comparison with versions of VPT-Deep and VQT that employ a larger number of tokens (VPT-Deep† and VQT†) on the VTAB-1K benchmark, with ViT-B/16 pre-trained on ImageNet-21K. "# Param" specifies the number of trainable parameters in backbones. Average accuracy are averaged over group-wise mean values.

		Natural							Specialized				Structured								
	# Param (M)	CIFAR100	Caltech101	DTD	Flower102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	Average
VPT-Deep [2]	0.17	**78.8**	**90.8**	65.8	98.0	88.3	**78.1**	49.6	81.8	96.1	83.4	68.4	68.5	60.0	46.5	72.8	73.6	47.9	**32.9**	**37.8**	72.0
VPT-Deep† [2]	2.84	26.6	69.5	50.9	82	63.6	27.4	18.7	77.6	92.1	72.1	74.3	53.7	39.3	32.9	70.5	14.0	11.0	9.9	25.9	53.2
VQT [6]	0.09	66.3	89.9	67.8	97.9	84.7	79.9	45.5	79.0	95.2	80.9	74.7	46.7	61.6	45.1	63.6	62.9	32.1	30.0	28.8	68.3
VQT† [6]	2.84	66.5	88.3	68.1	98.2	88.4	58.8	50.4	78.3	95.5	79.3	74.2	53.7	58.1	39.0	67.5	30.8	15.1	14.5	26.9	64.7
SynQT (Ours)	2.73	70.9	89.7	**68.8**	**98.5**	**89.6**	77.8	**50.6**	**82.3**	**96.7**	**83.5**	**75.2**	**71.8**	**62.7**	**48.5**	**75.4**	**74.1**	**49.0**	31.7	36.1	**72.9**

4 Discussion on Trainable Parameters

SynQT outperforms previous PETL methods while requiring slightly more trainable parameters due to a few projection layers and FFN operations. To validate that the superior performance of our method does not come from the additional parameters, we match the numbers of trainable parameters used in VPT-Deep [2] and VQT [6] with ours, by increasing their number of tokens to 300. We term the VPT-Deep and VQT with more trainable parameters as VPT-Deep† and VQT†, respectively. The comparison results are provided in Table 2.

We can see that increasing the number of tokens does not consistently yield a performance improvement across different datasets but may result in a significant performance drop. The observation is consistent with that in their original papers. That is, the optimal prompt length should not be too large, as the increased model complexity may cause overfitting due to the limited training data on VTAB-1K. Our results suggest that the key to the success of our SynQT is not the increased trainable parameters but the appropriate designs of our QSM and KEM.

5 Comparison among SynQT, Head2Toe, and LST

We further present a comparison among SynQT, Head2Toe [1], and LST [5] on the VTAB-1K benchmark, using ViT-B/16 pre-trained on ImageNet-1K, as shown in Table 3. Head2Toe is another method that utilizes intermediate representations for classification and simply applies an averaging over these features to reduce dimension. The input of Head2Toe consists of features from four distinct stages: after the layer normalization, after the Multi-head Attention block, inside, and after the MLP block. LST proposes a side network to sequentially take intermediate features as inputs for memory saving. LST also initializes the ladder-side network based on structural pruning to enhance performance.

Compared to Head2Toe and LST from the results, SynQT achieves a significant performance improvement. Specifically, SynQT outperforms Head2Toe on 16 out of 19 tasks and surpasses LST by 3.3% on average, indicating that the intermediate representations extracted by the task-specific synthesized query are more powerful than features derived from the original pre-trained model.

6 Discussion on Scaling Factors

In SynQT, we introduce two scaling factors s' and s'' to control the information flow in the QSM. A recent work [3] indicates that shifting the intermediate features helps close the gap between pre-training and the downstream task. Inspired by [3], in our case, we make use of these scaling factors to cope with the significant gaps between pre-training and various downstream tasks, e.g., 19 datasets in VTAB-1K.

To have a better understanding of these scaling factors, we further set them as 1 and report the results in Table 4. The results indicate that the scaling operation significantly improves the performance across different datasets. In particular, SynQT shows a 5.9% improvement in the Structured category and an average improvement of 2.4% due to these scaling factors.

7 Discussion on Frozen KEM

In SynQT, we reuse the original model's weights in the KEM due to their ability to effectively extract features from the intermediate features. In our case, we keep them frozen to leverage their abilities and preserve the feature interaction in the original space. Additionally, the frozen KEM is beneficial for PETL methods, effectively reducing the number of trainable parameters during training and preventing overfitting on downstream small datasets.

We also provide the comparison with a variant incorporating trainable KEM, whose training complexity would be similar to full-tuning. As shown in Table 5, a trainable KEM would not benefit SynQT due to the large number of trainable parameters and the limited training samples available in the downstream tasks.

Table 3: Comparison among SynQT, LST and Head2Toe on VTAB-1K benchmark with ViT-B/16 pre-trained on ImageNet-1K. Average accuracy are averaged over group-wise mean values.

	Natural							Specialized				Structured								
	CIFAR100	Caltech101	DTD	Flower102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele	Average
Head2Toe [1]	58.2	87.3	64.5	85.9	85.4	**82.9**	35.1	81.2	95.0	79.9	74.1	49.3	58.4	41.6	64.4	53.3	32.9	**33.5**	**39.4**	63.3
LST [5]	51.8	83.7	62.0	**93.2**	78.9	77.1	28.7	80.5	**96.6**	79.6	75.1	**76.3**	61.0	44.1	73.4	**73.9**	35.6	27.1	35.9	68.1
SynQT (Ours)	**59.2**	**89.7**	**66.2**	91.9	**88.9**	77.4	**39.0**	**84.0**	**96.6**	82.9	**75.4**	68.1	**60.2**	**47.9**	76.9	73.8	**52.4**	32.4	37.7	**71.4**

Table 4: Comparison with variant having a scale equal to 1 (SynQT†).

Methods	Natural	Specialized	Structured	Average
SynQT†	77.7	83.6	50.3	70.5
SynQT	**78.0**	**84.4**	**56.2**	**72.9**

Table 5: Comparison with variants having trainable KEM (SynQT‡).

Methods	Natural	Specialized	Structured	Average
SynQT‡	60.5	77.7	36.6	58.3
SynQT	**78.0**	**84.4**	**56.2**	**72.9**

8 t-SNE Visualization on More Datasets

We provide t-SNE [4] visualization on more datasets in Figure 3. Similar to the visualization in the main manuscript, it further shows that the features obtained by SynQT are more separable compared to the original CLS Token and VQT.

References

1. Evci, U., Dumoulin, V., Larochelle, H., Mozer, M.C.: Head2toe: Utilizing intermediate representations for better transfer learning. In: International Conference on Machine Learning. pp. 6009–6033. PMLR (2022)
2. Jia, M., Tang, L., Chen, B.C., Cardie, C., Belongie, S., Hariharan, B., Lim, S.N.: Visual prompt tuning. In: European Conference on Computer Vision. pp. 709–727. Springer (2022)
3. Lian, D., Zhou, D., Feng, J., Wang, X.: Scaling & shifting your features: A new baseline for efficient model tuning. Advances in Neural Information Processing Systems **35**, 109–123 (2022)
4. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
5. Sung, Y.L., Cho, J., Bansal, M.: Lst: Ladder side-tuning for parameter and memory efficient transfer learning. Advances in Neural Information Processing Systems **35**, 12991–13005 (2022)
6. Tu, C.H., Mai, Z., Chao, W.L.: Visual query tuning: Towards effective usage of intermediate representations for parameter and memory efficient transfer learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7725–7735 (2023)
7. Zaken, E.B., Ravfogel, S., Goldberg, Y.: Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. arXiv preprint arXiv:2106.10199 (2021)

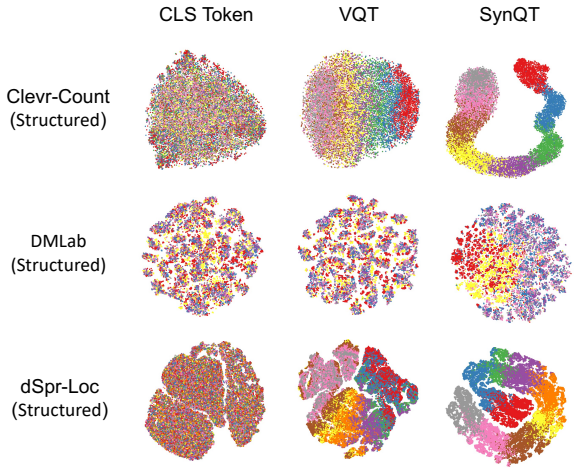


Fig. 3: t-SNE visualization on more datasets including Clevr-Count, DMLab, and dSpr-Loc.