GraphAdapter: Tuning Vision-Language Models With Dual Knowledge Graph (Supplementary Materials)

Anonymous Author(s)
Affiliation
Address
email

Sec. 1 validates the applicability of our GraphAdapter by introducing it to the state-of-the-art adapter-style tuning methods, including CaFo [15] and TaskRes* [14].

Sec. 2 provides more experimental details on Few-shot Learning for our GraphAdapter.

Sec. 3 describes more details about datasets and implementation.

Sec. 4 visualizes the textual graph nodes used for classification before and after utilizing our GraphAdapter.

Sec. 5 makes a comprehensive analysis of the possible broader impacts.

1 Applicability

To validate the applicability of our GraphAdapter, we select two state-of-the-art adapter-style works, including CaFo [15] and TaskRes* [14]. Here, CaFo [15] incorporates diverse prior knowledge from large pre-trained vision and language models, including DINO's vision-contrastive knowledge, GPT-3's language-generative knowledge, and DALLE's generative capability. The adapting strategy of CaFo [15] is from the Tip-Adapter [16]. The TaskRes* denotes the enhanced version of TaskRes [14], which exploits the enhanced base classifier instead of the original classifier from CLIP [11].

For CaFo [15], we directly incorporate our GraphAdapter into the textual classifier. For TaskRes* [14], we replace the task residual with our proposed GraphAdapter and maintain its enhanced textual branch from CLIP. The experimental results on ImageNet [3] are shown in Table 1. We can observe that our GraphAdapter can consistently increase the performance of CaFo [15] and TaskRes* [14] on few-shot learning with all 1-/2-/4-/8-/16-shots settings. Particularly, on the 16-shot setting, ours improves CaFo [15] by 0.51%, and TaskRes* by 1.15%, which validates the powerful applicability of our GraphAdapter. *Overall, our GraphAdapter is complementary to these prior-augmented methods, and can obtain better performance by integrating ours into them.*

Table 1: The experiments for the applicability of our GraphAdapter. For Cafo [15], we incorporate our GraphAdapter into the textual classifier. Notably, the TaskRes* exploits the enhanced base classifier. Therefore, TaskRes* + Ours denotes that TaskRes* replace the task residual with our proposed GraphAdapter.

Methods	1-shot	2-shot	4-shot	8-shot	16-shot
CaFo [15]	63.80	64.34	65.64	66.86	68.79
+Ours	**63.81**	**64.97**	**66.17**	**67.68**	**69.30**
TaskRes*[14]	61.43	62.17	62.93	64.03	64.75
+Ours	**61.73**	**62.53**	**63.47**	**64.57**	**65.80**

2 More Experimental Results

We present the numerical results of "Figure 3 in the main text" as Table 2. We compare our GraphAdapter with the state-of-the-art works, including the prompt-based method CoOp [17], and adapter-style methods, *i.e.*, CLIP-Adapter [5], Tip-Adapter-F [16], and TaskRes [14]. Here, the performance of Tip-Adapter-F is reproduced by [14], which aims to ensure a fair comparison with CoOp [17]. From the table, we can find that on the 16-shot few-shot learning, our GraphAdapter outperforms all previous works except for UCF101 [12] where its performance is comparable. Depart from that, for the average accuracy of 11 benchmark datasets in the 1-/2-/4-/8-/16-shot few-shot learning, our GraphAdapter surpasses previous works with a consistent improvement of 0.57% to 0.76%. We also make the analysis for the Error Bars by providing the standard deviation (Std) of our experimental results in Table 2.

Table 2: A numerical comparison between our GraphAdapter and the state-of-the-art methods.

Methods	Setting	Caltech101	DTD	EuroSAT	FGVCAircraft	Flowers102	Food101	ImageNet	OxfordPets	StanfordCars	SUN397	UCF101	Avg.
Zero-shot CLIP [11]	1-shot	86.29	42.32	37.56	17.28	66.14	77.31	58.18	85.77	55.61	58.52	61.46	58.77
CoOp [17]		87.53	44.39	50.63	9.64	68.12	74.32	57.15	85.89	55.59	60.29	61.92	59.59
CLIP-Adapter [5]		88.60	45.80	61.40	17.49	73.49	**76.82**	61.20	85.99	55.13	61.30	62.20	62.67
Tip-Adapter-F [16]		88.80	50.49	50.34	19.01	**81.17**	76.22	60.88	**86.04**	56.78	61.23	**66.19**	63.38
TaskRes [14]		88.80	50.17	61.27	**21.20**	78.77	74.03	61.43	83.50	58.77	**61.93**	64.57	64.04
Ours (w/ Std)		**88.90**	**51.77**	**63.30**	20.93	79.98	75.43	**61.50**	84.40	**59.70**	**61.93**	64.93	**64.80**
		(±0.22)	(±1.48)	(±1.96)	(±0.25)	(±0.90)	(±0.14)	(±0.09)	(±1.02)	(±0.45)	(±0.26)	(±0.59)	(±0.34)
Zero-shot CLIP [11]	2-shot	86.29	42.32	37.56	17.28	66.14	**77.31**	58.18	85.77	55.61	58.52	61.46	58.77
CoOp [17]		87.93	45.15	61.50	18.68	77.51	72.49	57.81	82.64	58.28	59.48	64.09	62.32
CLIP-Adapter [5]		89.37	51.48	63.90	20.10	81.61	77.22	61.52	**86.73**	58.74	63.29	67.12	65.55
Tip-Adapter-F [16]		89.61	55.32	64.76	21.76	85.40	77.05	61.57	86.06	61.13	63.19	68.99	66.80
TaskRes [14]		90.13	54.53	65.77	23.07	**85.63**	75.30	62.17	84.43	62.77	64.33	69.10	67.02
Ours (w/ Std)		**90.20**	**55.75**	**67.27**	**23.80**	**85.63**	76.27	**62.32**	86.30	**63.23**	**64.60**	**69.47**	**67.71**
		(±0.22)	(±1.56)	(±1.57)	(±0.65)	(±0.25)	(±0.12)	(±0.17)	(±0.99)	(±0.12)	(±0.33)	(±0.42)	(±0.31)
Zero-shot CLIP [11]	4-shot	86.29	42.32	37.56	17.28	66.14	77.31	58.18	85.77	55.61	58.52	61.46	58.77
CoOp [17]		89.55	53.49	70.18	21.87	86.20	73.33	59.99	86.70	62.62	63.47	67.03	66.77
CLIP-Adapter [5]		89.98	56.86	73.38	22.59	87.17	**77.92**	61.84	**87.46**	62.45	65.96	69.05	68.61
Tip-Adapter-F [16]		90.87	**60.25**	69.66	26.39	89.53	77.46	62.62	86.46	64.86	65.88	**72.71**	69.70
TaskRes [14]		90.63	59.50	72.97	24.83	89.50	76.23	62.93	86.27	66.50	66.67	69.70	69.61
Ours (w/ Std)		**90.97**	59.63	**75.20**	**26.97**	**89.90**	76.77	**63.12**	86.57	**66.53**	**66.70**	71.47	**70.35**
		(±0.05)	(±0.39)	(±1.37)	(±0.29)	(±0.19)	(±0.26)	(±0.19)	(±1.47)	(±0.29)	(±0.28)	(±0.16)	(±0.27)
Zero-shot CLIP [11]	8-shot	86.29	42.32	37.56	17.28	66.14	77.31	58.18	85.77	55.61	58.52	61.46	58.77
CoOp [17]		90.21	59.97	76.73	26.13	91.18	71.82	61.56	85.32	68.43	65.52	71.94	69.89
CLIP-Adapter [5]		91.40	61.00	77.93	26.25	91.72	**78.04**	62.68	87.65	67.89	67.50	73.30	71.40
Tip-Adapter-F [16]		91.70	62.93	79.33	30.62	91.00	77.90	64.15	**88.28**	69.51	**69.23**	74.76	72.67
TaskRes [14]		92.23	64.23	78.07	29.50	**94.30**	76.90	64.03	87.07	**70.57**	68.70	74.77	72.76
Ours (w/ Std)		**92.45**	**64.50**	**80.17**	**31.37**	94.07	77.73	**64.23**	87.63	70.53	68.97	**75.73**	**73.40**
		(±0.38)	(±0.34)	(±1.87)	(±0.40)	(±0.12)	(±0.19)	(±0.08)	(±0.26)	(±0.12)	(±0.12)	(±0.45)	(±0.29)
Zero-shot CLIP [11]	16-shot	86.29	42.32	37.56	17.28	66.14	77.31	58.18	85.77	55.61	58.52	61.46	58.77
CoOp [17]		91.83	63.58	83.53	31.26	94.51	74.67	62.95	87.01	73.36	69.26	75.71	73.42
CLIP-Adapter [5]		92.49	65.96	84.43	32.10	93.90	78.25	63.59	87.84	74.01	69.55	76.76	74.44
Tip-Adapter-F [16]		92.63	66.94	84.94	35.86	94.23	78.11	65.44	88.18	75.75	71.00	**79.03**	75.65
TaskRes [14]		92.90	**67.57**	82.57	33.73	96.10	78.23	64.75	88.10	74.93	70.30	76.87	75.10
Ours (w/ Std)		**93.33**	**67.57**	**85.27**	**36.87**	**96.23**	**78.63**	**65.70**	**88.57**	**76.23**	**71.20**	78.80	**76.22**
		(±0.08)	(±0.09)	(±0.29)	(±0.50)	(±0.16)	(±0.08)	(±0.08)	(±0.51)	(±0.17)	(±0.08)	(±0.26)	(±0.11)

3 More Dataset and Implementation Details

More Dataset Details. In this paper, we follow previous works, *e.g.*, CoOp [17], CLIP-Adapter [5], TaskRes [14], and Tip-Adapter [16], and exploit the prompts in Table 3 for the tuning and testing.

More Implementation Details. Our experimental results are achieved by running the algorithm three times with different seeds for each setting. The training and inference are implemented with a single NVIDIA GeForce RTX 3090. In the implementation of GraphAdapter for the ImageNet [3], we decouple the sub-graph with 1000 nodes for each modality into four graphs with 256 nodes to alleviate the computational cost.

4 Visualization of Graph Nodes

To demonstrate how our GraphAdapter works for the adapter-style tuning for VLMs, we visualize the graph nodes for textual features before and after the GraphAdapter. As shown in Figure 1, we

Table 3: The number of classes and the used prompt temple for each dataset.

Datasets	# Classes	Prompt Templet
Caltech101 [4]	100	"a photo of a [class]."
DTD [2]	47	"[class] texture."
EuroSAT [6]	10	"a centered satellite photo of [class]."
FGVCAircraft [8]	100	"a photo of a [class], a type of aircraft."
Flowers102 [9]	102	"a photo of a [class], a type of flower."
Food101 [1]	101	"a photo of a [class], a type of food."
OxfordPets [10]	37	"a photo of a [class], a type of pet."
StanfordCars [7]	196	"a photo of a [class]."
SUN397 [13]	397	"a photo of a [class]."
UCF101 [12]	101	"a photo of a person doing [class]."
ImageNet [3]	1000	Ensemble of 7 selected templates, including "itap of a [class].", "a bad photo of the [class].", "a origami [class].", "a photo of the large [class].", "a [class] in a video game.", "art of the [class]." and "a photo of the small [class]." .

randomly sampled 20 classes from ImageNet [3] and utilize the t-SNE to visualize the distribution of each node corresponding to the textual fracture for classification. We can observe that with our GraphAdapter, the nodes of different classes move in directions that lead to much larger inter-class distances, thereby improving the performance of adapter-style tuning for VLMs.

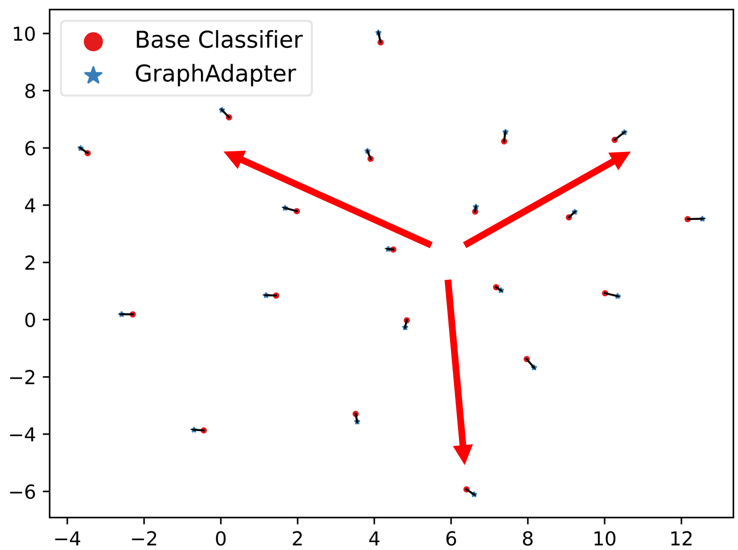

Figure 1: Visualization of the variance of the graph nodes before and after GraphAdapter. Each node represents the representation of one class. We randomly sampled 20 classes from ImageNet for better visualization. The nodes move toward the direction that leads to much larger inter-class distances after GraphAdapter. The red arrows denote the directions.

5 Broader Impacts

The adapter-style tuning of VLMs aims to efficiently finetune the VLMs for downstream tasks by optimizing a few parameters in the low-data regime. The possible broader impact of our GraphAdapter stems from the tuning of VLMs itself, which has a heavy dependency on the pre-trained VLMs. The utilization of our GraphAdapter should follow the privacy and safety of datasets and pre-trained models.

References

[1] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101–mining discriminative components with random forests. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI 13*, pages 446–461. Springer, 2014.

[2] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[4] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *2004 conference on computer vision and pattern recognition workshop*, pages 178–178. IEEE, 2004.

[5] P. Gao, S. Geng, R. Zhang, T. Ma, R. Fang, Y. Zhang, H. Li, and Y. Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021.

[6] P. Helber, B. Bischke, A. Dengel, and D. Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[7] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[8] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013.

[9] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[10] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[11] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[12] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[13] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3485–3492. IEEE, 2010.

[14] T. Yu, Z. Lu, X. Jin, Z. Chen, and X. Wang. Task residual for tuning vision-language models. *arXiv preprint arXiv:2211.10277*, 2022.

[15] R. Zhang, X. Hu, B. Li, S. Huang, H. Deng, H. Li, Y. Qiao, and P. Gao. Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners. *arXiv preprint arXiv:2303.02151*, 2023.

[16] R. Zhang, W. Zhang, R. Fang, P. Gao, K. Li, J. Dai, Y. Qiao, and H. Li. Tip-adapter: Training-free adaption of clip for few-shot classification. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXV*, pages 493–510. Springer, 2022.

[17] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022.