Supplementary Materials: Deep Instruction Tuning for Segment Anything Model

1 QUANTITATIVE ANALYSIS

1.1 Training Memory Consumption

We compare the memory overhead of L-DIT and existing methods in Tab. 1. From this table, we can see that the additional cost of backbone frozen is close to linear probe tuning and smaller than the full finetuning. In addition, the GPU overhead of full finetuning is even much smaller than LAVT [8] in referring image segmentation, since L-DIT does not require another deep fusion model. Compared with LAVT, our L-DIT is simpler and more intuitive, which only requires linear projections to insert text words into each SAM's layer. These results further validate the efficiency of L-DIT.

Table 1: The memory consumption of L-DIT and existing methods. Here, ✓ means that the encoder is frozen, while ✗ is updated.

Model	Visual	Text	Batch Size	GPU memory
L-DIT	✓	✓	32	11.9G
L-DIT	✓	✗	32	12.4G
L-DIT	✗	✗	32	17.1G
LAVT [8]	✗	✗	8	24G
ReLA [2]	✗	✗	32	12G

1.2 The impact of shared parameters for the injection of text features into the visual backbone

In the visual backbone, the shallow and deep layers typically extract low- and high-level features, respectively. To validate this intuition, we experiment on the progressive cascade structure in Tab. 2. From this table, we see that there is no obvious difference between *layer-wise* and *cascade* structures. To explain, DIT aims to project the text words onto the visual semantic space of SAM, and simple linear projections may be enough. Similar findings can be also found on recent MLLMs [3], where the bridge part is often a stack of linear layers.

Table 2: The impact of shared parameters for the injection of text features into the visual backbone. *Cascade* means linear layers replace with progressive cascade structure.

Structure	val		testA		testB	
	mIoU	oIoU	mIoU	oIoU	mIoU	oIoU
Layer-wise DIT	69.97	67.50	72.44	69.55	68.12	64.77
Cascade DIT	69.94	66.76	72.20	69.72	66.93	63.38

Table 3: Comparison of methods using additional data.

Setting	Data	Backbone	val		testA		testB	
			oIoU	mIoU	oIoU	mIoU	oIoU	mIoU
Default-SAM	3.88M	ViT-B	67.17	69.85	69.99	72.33	65.62	68.08
LISA-7B [1]	18.16M	ViT-H	74.90	-	79.10	-	72.30	-
PolyFormer [4]	5.7M	Swin-L	75.96	-	**78.29**	-	73.25	-
Layer-wise DIT	3.88M	ViT-B	**76.20**	**77.15**	77.85	**79.03**	**73.53**	**75.01**

1.3 Speed comparison and using more training data

When training from scratch, L-DIT with ViT-B performs slightly worse than methods using Swin-B. However, it significantly outpaces these state-of-the-art (SOTA) methods in terms of speed, achieving 123 ms compared to 320 ms for ReLA, 350 ms for CARIS, and 530 ms for PolyFormer. Furthermore, as the amount of training data increases, L-DIT demonstrates its potential by surpassing SOTA methods like PolyFormer, as illustrated in Tab. 3. This highlights L-DIT's promising capabilities in text-guided segmentation.

Table 4: Comparison on the multi-target GRES dataset.

Method	Backbone	val		testA		testB	
		oIoU	mIoU	oIoU	mIoU	oIoU	mIoU
CRIS [6]	CLIP-R101	55.34	56.27	63.82	63.42	51.04	51.79
LAVT [8]	Swin-B	57.64	58.40	65.32	65.90	55.04	55.83
ReLA [2]	Swin-B	62.42	63.60	**69.26**	**70.03**	59.88	61.02
L-DIT	ViT-B	**63.76**	**65.87**	67.19	68.25	**61.85**	**64.52**

1.4 Comparison with the state-of-the-art methods on GRES.

In Tab.4, we present a comparison of our L-DIT against the state-of-the-art (SOTA) methods for multi-target referring image segmentation on the GRES[2] dataset, utilizing the *oIoU* metric. Notably, L-DIT demonstrates outstanding performance on the GRES dataset, showcasing capabilities that exceed those of existing SOTA methods [2, 8].

1.5 The effect of freezing encoders

In Tab. 5, we also ablate DITs under the parameter efficient setting [5, 7]. It can be seen that when freezing the SAM's image backbone and the directly plugged-in BERT, L-DIT can still achieve 67.16% *oIoU* on RefCOCO *val*, which is even much better than fully tuning the default SAM, as shown in Tab. 5. Unfreezing both encoders can greatly improve performance. But compared with the pre-trained BERT, the significance of updating image encoder is much more obvious, *e.g.*, +4.88% *oIoU* on *val*. This result suggests that in DIT, the image encoder is capable of visual feature learning

Table 5: The impact of freezing the visual and text encoders. Here, ✓means that the encoder is frozen, while ✗is updated.

Visual	Text	Method	val					testA					testB				
			P@0.5	P@0.7	P@0.9	oIoU	mIoU	P@0.5	P@0.7	P@0.9	oIoU	mIoU	P@0.5	P@0.7	P@0.9	oIoU	mIoU
✓	✓	End-to-end	71.83	57.95	18.83	60.76	62.87	75.41	62.49	19.05	63.61	65.67	68.15	52.43	19.54	57.97	60.23
✓	✓	Layer-wise	80.93	69.26	25.00	67.16	69.56	81.64	71.33	23.00	68.24	70.60	73.98	61.01	25.29	62.70	65.59
✓	✗	End-to-end	76.51	66.00	23.19	62.96	66.87	78.89	68.41	21.31	63.74	67.83	72.90	60.01	25.39	60.43	64.63
✓	✗	Layer-wise	81.22	70.21	24.31	67.50	69.97	83.89	74.66	24.36	69.55	72.44	77.26	65.44	28.26	64.77	68.12
✗	✗	End-to-end	81.78	73.07	28.27	68.07	71.46	85.28	77.17	**27.56**	70.81	73.90	78.02	67.60	30.20	65.58	69.36
✗	✗	Layer-wise	**85.68**	**76.61**	**29.89**	**71.98**	**74.73**	**88.06**	**79.17**	25.94	**74.51**	**75.62**	**81.28**	**69.11**	**30.95**	**68.77**	**71.21**

Table 6: The impact of different text encoders for L-DIT.

Text encoder	val		testA		testB	
	oIoU	mIoU	oIoU	mIoU	oIoU	mIoU
BERT	67.50	69.97	69.55	72.44	64.77	68.12
CLIP	67.19	69.82	68.99	71.60	64.88	67.67

and cross-modal interactions, well confirming our intuition about DIT.

1.6 The impact of different text encoder on Layer-wise DIT

Tab. 6 examines the impact of different language encoders. It can be seen that BERT outperforms CLIP in most cases, though the gap between the two encoders is not significant. For example, CLIP performs slightly better than BERT in terms of *oIoU* on *testB*. We believe this is due to the typically concise descriptions in the dataset, resulting in a relatively limited impact of the text encoder on DIT-SAM.

2 QUALITATIVE ANALYSIS

The Deep Instruction Tuning (DIT) approach can be effectively adapted to tackle multi-objective task in GRES. To illustrate the effectiveness of this extension, we present a visualization of the results achieved in multi-object segmentation, as shown in Fig. 1. This figure highlights the performance improvements and the ability of DIT to handle multiple objectives simultaneously, demonstrating its versatility and efficiency in complex segmentation scenarios.

To gain deep insights into DIT, we visualize the attention maps of our DIT tuning methods and the default SAM in Fig. 2. We observe some interesting findings about the self-attention of SAM backbone after inserting text tokens.

For instance, under the frozen backbone setting, the inserted text tokens mainly interact with the visual ones on the higher layers of SAM's encoder, *e.g.*, the 8-*th* or 11-*th* layers. This suggests that the text mainly interacts with the high-level visual semantics of SAM when only tuning the DIT projections.

In contrast, under the full tuning setting, the interactions mainly happen at the shallow layers, *e.g.*, 2-*nd* and 4-*th* layers. It might indicate that the text information is early embedded into the visual tokens and then for the deep multi-modal fusion in the image encoder of SAM.

Moreover, it can be observed from the visual attention map that for each layer of the visual backbone, the attentive area of the text instruction is different. For default SAM, it has no obvious interested area, which mainly divides the image into foreground and background. However, for E-DIT and L-DIT, it can be clearly seen that each layer has its attention tendency for different text instructions. The initial layers are interested in entity nouns in text instructions. The 4-*th* layer is more sensitive to comparative words. At 11-*th* layer, it mainly focuses on digital information. The 12-*th* layer is more inclined to color information.

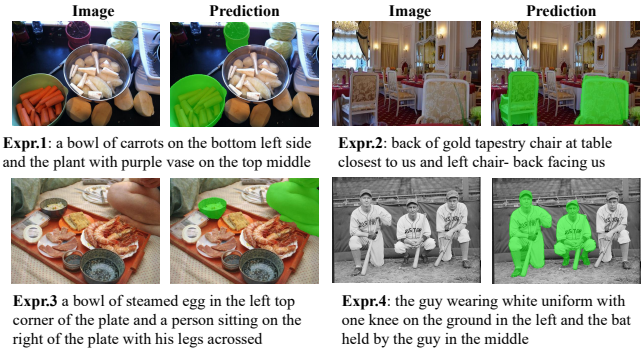

Figure 1: Visualized multi-target segmentations by DIT-SAM.

REFERENCES

[1] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. 2023. Lisa: Reasoning segmentation via large language model. *arXiv* (2023).

[2] Chang Liu, Henghui Ding, and Xudong Jiang. 2023. GRES: Generalized Referring Expression Segmentation. In *CVPR*.

[3] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2024. Visual instruction tuning. *NeurIPS* (2024).

[4] Jiang Liu, Hui Ding, Zhaowei Cai, Yuting Zhang, Ravi Kumar Satzoda, Vijay Mahadevan, and R. Manmatha. 2023. PolyFormer: Referring Image Segmentation As Sequential Polygon Generation. In *CVPR*.

[5] Gen Luo, Minglang Huang, Yiyi Zhou, Xiaoshuai Sun, Guannan Jiang, Zhiyu Wang, and Rongrong Ji. 2023. Towards efficient visual adaption via structural re-parameterization. *arXiv* (2023).

[6] Zhaoqing Wang, Yu Lu, Qiang Li, Xunqiang Tao, Yandong Guo, Mingming Gong, and Tongliang Liu. 2022. Cris: Clip-driven referring image segmentation. In *CVPR*.

[7] Qiong Wu, Wei Yu, Yiyi Zhou, Shubin Huang, Xiaoshuai Sun, and Rongrong Ji. 2023. Parameter and Computation Efficient Transfer Learning for Vision-Language Pre-trained Models. *NeurIPS* (2023).

[8] Zhao Yang, Jiaqi Wang, Yansong Tang, Kai Chen, Hengshuang Zhao, and Philip HS Torr. 2022. Lavt: Language-aware vision transformer for referring image segmentation. In *CVPR*.

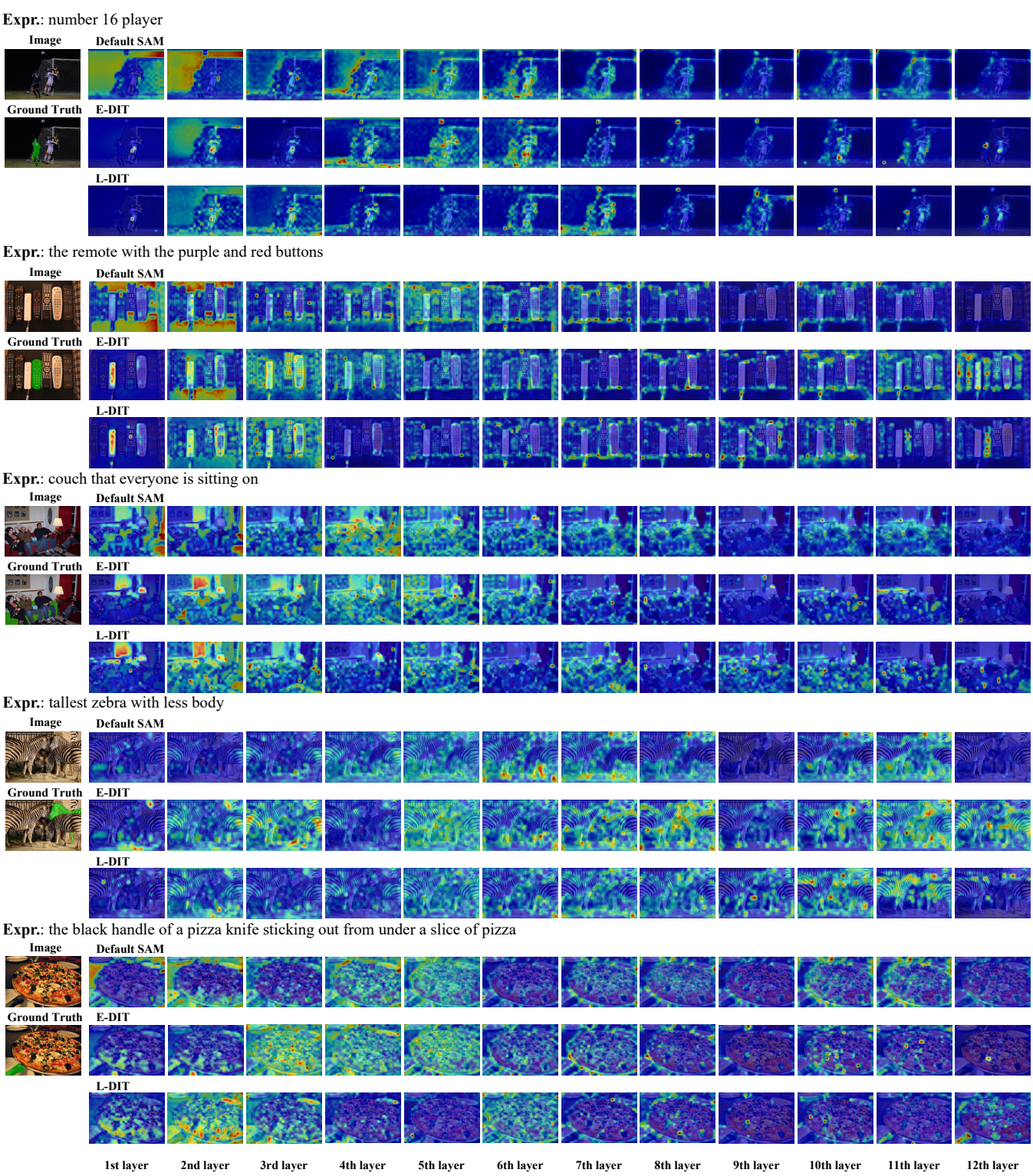

Figure 2: The attention maps of the default SAM and our end-to-end and layer-wise DITs, *i.e.*, E-DIT and L-DIT. Both methods can follow text instructions better than the default SAM and adjust the area of interest according to text instructions.