Feature Design for Bridging SAM and CLIP toward Referring Image Segmentation –Supplementary Material–

Koichiro Ito
Hitachi, Ltd. R&D Group
koichiro.ito.tm@hitachi.com

Abstract

In this material, we provide supplementary explanations of the existing methods utilized in our paper and the details of the experiments necessary for the better understanding of our paper.

1. Prompt encoder and mask decoder in SAM

SAM [2] accepts a diverse range of prompts for the segmentation. In our paper, we design a dense feature $\mathbf{p}_{\text{dense}}$ and a sparse feature $\mathbf{p}_{\text{sparse}}$ for the mask estimation. Note that in our method, we set the SAM decoder trainable, thus we feed each feature to the decoder not as the prompt in original usage. We first describe how to feed these features into the SAM prompt encoder.

Sparse embedding: SAM originally contains query features for the mask prediction and the iou prediction denoted as $\mathbf{p}_{\text{query}} \in \mathbb{R}^{N_m \times C}$ and $\mathbf{p}_{\text{iou}} \in \mathbb{R}^{C}$ respectively, where $N_m = 3$. In basic usage, prompts such as a bounding box or a point information are encoded into the vector-level features and concatenated with these queries. In our paper, our sparse feature $\mathbf{p}_{\text{sparse}} \in \mathbb{R}^{C}$ is also concatenated with the queries and fed into the SAM mask decoder.

Dense embedding: SAM also accepts a dense embedding for its prompt, such as the binary mask information. The dense feature $\mathbf{p}_{\text{dense}} \in \mathbb{R}^{P^2_{\text{CLIP}} \times C}$ extracted through the pixel decoder from $\mathcal{F}_{\text{fused}}$ obtained from our DFEM is added to the SAM image feature $\mathbf{F}_{\text{sam}} \in \mathbb{R}^{P^2 \times C}$. Before this addition, the feature size of $\mathbf{p}_{\text{dense}}$ is resized from P^2_{CLIP} to P^2 through a 2D interpolation operation.

Next, in SAM mask decoder, cross attention between these sparse embeddings and dense embeddings are conducted to produce the output queries, from which the masks are obtained.

Predicted Mask: SAM predicts multiple masks simultaneously, each correspond to its query feature $\mathbf{p}_{\text{query}}$ (with $N_m = 3$ in default). We utilize the first index mask as $\mathbf{M}_{\text{SAM}}$ for the training and the evaluation.

2. GRES setting

Non-target branch: In the GRES setting [1, 4], the model is required to conduct the multi-target or the non-target prediction, which differs from RefCOCO/+/g problem setting. In our proposal, our model can adapt to the multi-target setting because the model is not on the singular target assumption, but to handle non-target prediction, we introduce an additional branch. Following the work [4], we introduce an additional linear layer for predicting non-target denoted as $\mathbf{y}_{\text{nt}} \in \mathbb{R}^2$ to the output $\hat{\mathbf{X}}_{N_l}$ of DFAM as follows:

$$\mathbf{y}_{\text{nt}} = \omega_{\text{nt}}(\hat{\mathbf{X}}_{N_l}), \tag{1}$$

where ω_{nt} is projection function. We also introduce the loss function to $\mathbf{y}_{\text{nt}}$ as follows:

$$\mathcal{L}_{\text{nt}} = \lambda_{\text{nt}} \text{CE}(\mathbf{y}_{\text{nt}}, \mathbf{y}^*_{\text{nt}}), \tag{2}$$

where $\text{CE}(\cdot)$ is the cross entropy loss and $\mathbf{y}^*_{\text{nt}}$ is the ground truth for the non-target prediction provided in the dataset. We set $\lambda_{\text{nt}} = 0.1$ and added the loss to the total loss $\mathcal{L}_{\text{total}}$ for the training. In the evaluation, we use the output $\mathbf{y}_{\text{nt}}$ following the works. The evaluation protocols between the works [1, 4] are slightly different. We follow each work on our comparison.

Sparse feature design: In the GRES setting, the coco category to the expression in non-target situation is not provided. Thus, we utilize $\mathbf{L}_{\text{BERT},\texttt{[CLS]}} \in \mathbb{R}^C$ instead of $\mathbf{L}_{\text{cat}} \in \mathbb{R}^C$ for the design of the sparse feature $\mathbf{p}'_{\text{sparse}}$ as follows:

$$\mathbf{p}'_{\text{sparse}} = \text{LayerNorm}(\mathbf{L}_{\text{BERT},\texttt{[CLS]}} \tag{3}$$
$$+ \text{CrossAttn}\left(\mathbf{L}_{\text{BERT},\texttt{[CLS]}}, \mathbf{L}_{\text{BERT}}\right)). \tag{4}$$

The sparse feature design above is different from the original one described in Sec. 3.4. We conduct experiment

Table 1. The effect of $\mathbf{p}'_{\text{sparse}}$ on RefCOCO+ dataset.

Sparse feature	Val		TestA		TestB	
	oIoU	mIoU	oIoU	mIoU	oIoU	mIoU
$\mathbf{p}_{\text{sparse}}$	66.19	70.56	72.37	75.87	56.75	62.08
$\mathbf{p}'_{\text{sparse}}$	66.08	70.08	71.02	75.41	56.70	61.89

Table 2. Parameter sizes of each component (in millions of parameters [M]).

	CLIP ViT	SAM ViT	SAM Decoder	BERT	DFAM	PWAM	Total
Frozen	304.3	89.67	–	–	–	–	394.0
Trainable	–	–	4.06	108.9	6.90	7.88	130.0

on RefCOCO+ using $\mathbf{p}'_{\text{sparse}}$ and shows result on Tab. 1. As can be seen in the table, this did not cause a big difference in the RIS result.

Implementations: The computational cost of SAM's ViT is expensive, so to enhance experimental efficiency, we pre-extracted the image feature from the encoder as pickle files before the training process. We used BERT through the Hugging Face library. The feature dimensions output by each encoder were $C_{\text{SAM}} = 1024$, $C_{\text{CLIP-V}} = 1024$, and $C_t = 768$. To handle these features in a unified dimension, we applied a linear layer to project them to $C = 256$. This mapping aligned the feature dimensions of $\mathcal{D}_{\text{SAM}}$ accordingly.

Both SAM and CLIP require a predetermined resolution of images, so we resized the images to 1024^2 and 336^2 pixels, respectively, before inputting them. This resizing to square inputs allows for the alignment of $\mathbf{F}_{\text{SAM}}$ and $\mathbf{p}_{\text{dense}}$ when fed into SAM decoder $\mathcal{D}_{\text{SAM}}$. Since RIS involves language queries that inquire about the spatial relationships between objects, we did not apply any augmentation to the input images. We set $N_q = 100$ in DFAM and $N_l = 3$ for number of layers. The hyperparameters used in the loss function described in Sec. 3.5 were set as follows: $\lambda_{\text{Dice}} = 2$, $\lambda_{\text{BCE}} = 0.5$, $\lambda_{\text{DFAM}} = 1.0$, $\lambda_{\text{Dice},l} = 5$, $\lambda_{\text{BCE},l} = 5$, and $\lambda_{\text{VL},l} = 1$. The batch size was set to 32, and the initial learning rate was $5e-5$.

3. Additional Evaluations

In this section, we provide additional evaluations to further validate the effectiveness of our proposed method.

3.1. The module sizes on our model

In the main paper, we discussed the overall computational cost of the proposed method. Here, we provide additional details regarding the parameter sizes of each component of the model, as shown in Tab. 2.

As seen in Tab. 2, the frozen components (CLIP ViT and SAM ViT) account for the majority of the model's parameter size, while the trainable components (SAM Decoder, BERT, DFAM, and PWAM) add up to a smaller portion.

Table 3. Reproduced ReLA on RefCOCO/g in terms of oIoU and mIoU. U: UMD split.: † indicates the model reproduced by us.

Methods	RefCOCO			G-Ref	
	Val	Test A	Test B	Val$_{(\text{U})}$	Test$_{(\text{U})}$
oIoU					
ReLA [4]	73.82	76.48	70.18	65.00	65.97
ReLA † [4]	73.73	76.71	69.98	64.40	65.63
Ours	**74.80**	**78.02**	69.96	**65.34**	**67.08**
mIoU					
ReLA † [4]	76.06	78.19	73.05	67.87	68.45
Ours	**76.75**	**79.55**	72.74	**69.56**	**70.34**

Table 4. The evaluation on Precision@X on G-ref umd.: † indicates the model reproduced by us.

	Pr@0.5	Pr@0.6	Pr@0.7	Pr@0.8	Pr@0.9
G-ref val					
ReLA † [4]	76.92	72.90	66.79	56.70	29.68
ours	78.16	74.75	69.85	60.04	33.61
G-Ref test					
ReLA † [4]	77.55	73.50	67.70	57.08	30.65
ours	79.34	75.52	70.38	60.95	34.57

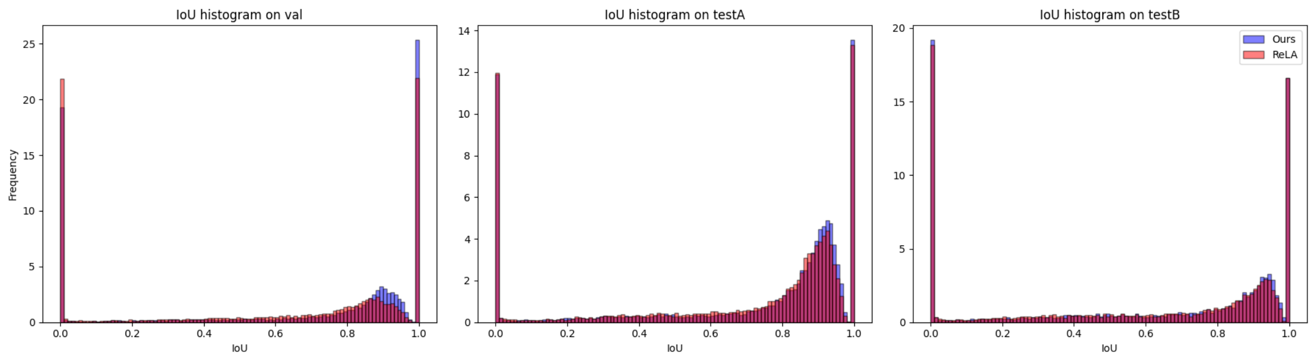
The trainable parameter contains other components such as projection layers, convolution and deconvolution in SFP etc. This division highlights the efficiency of our method in terms of trainable parameters, as the majority of the model remains frozen, contributing to the training efficacy. We did not count the parameters on CLIP-T here, since we can extract the class category feature $\mathbf{L}_{\text{cat}}$ in advance before the actual inference.

3.2. Analysis on IoU distribution

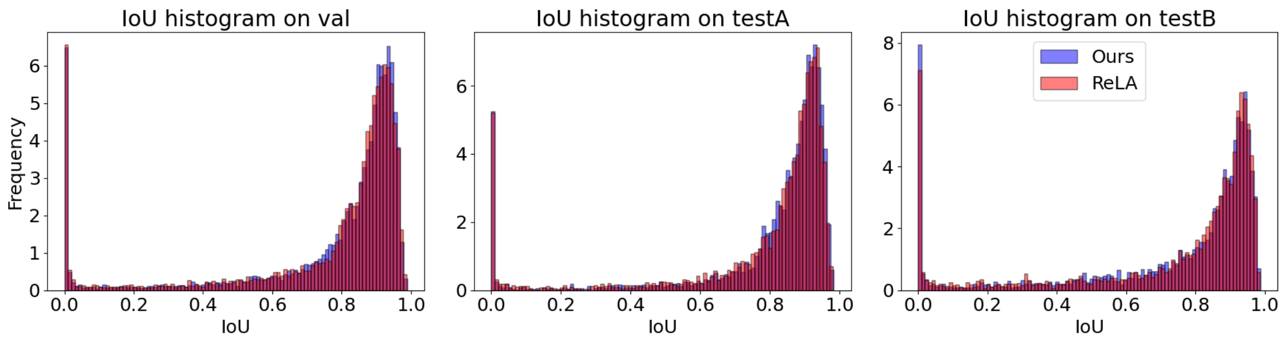
We further evaluate the model performance regarding the IoU. We utilize the ReLA model parameters trained on gRefCOCO which is publicly available. On the other hand, we trained the ReLA model on RefCOCO and G-Ref for comparison since the trained models for these datasets are not released. The reproduced result is also shown in Tab. 3.

Fig. 1 shows the histograms of the IoU distribution on gRefCOCO, RefCOCO and G-ref umd split. Fig. 1(a) shows that the distribution of the histogram is skewed towards the higher values, demonstrating the superiority of our proposed method on the gRefCOCO. When evaluating on gRefCOCO, we use the gIoU metric, which results in a high frequency of IoU = 1.0 on each method. Fig. 1(b) shows the competitive performance of our method since it is not suited to handling simple language expressions in RefCOCO as discussed in the paper. However, Fig. 1(c) shows our superiority on G-ref.

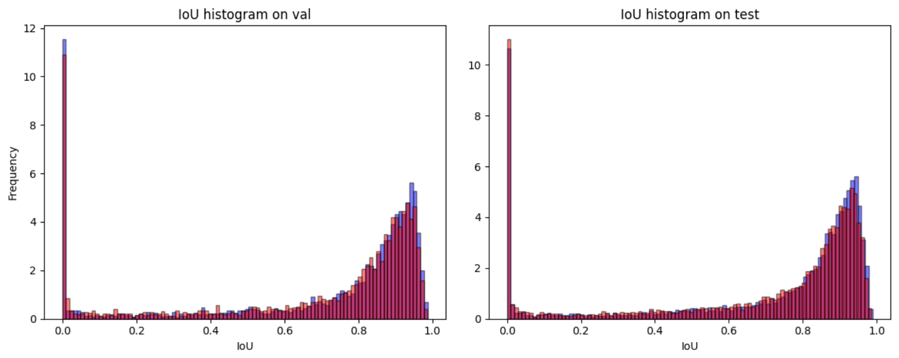
Since ReLA uses a learnable encoder, it is likely to be adaptable to the dataset producing high IoU value. Regarding our method, our training method can effectively specify the target object when the language expressions in the training data are precise enough to narrow down the target in cases such as gRefCOCO and G-ref dataset.



(a) Comparison on grefCOCO



(b) Comparison on RefCOCO



(c) Comparison on G-Ref umd split

Figure 1. Comparative IoU histograms between Ours and ReLA on G-Ref umd and grefCOCO

Tab. 4 evaluate the methods using Precision@X (Pr@X). Pr@X counts the percentage of samples with IoU higher than the threshold X. The table is almost identical to Fig. 1(c) and easy for visualizing the performance. The table shows our method shows superiority regarding the precision increased as the IoU threshold become higher.

3.3. Comparison with LLaVA based method

Here, we compare our method with the LLaVA-based method in Tab. 5. LISA [3] is composed of a 7B LLM, CLIP, SAM and a LoRA adapter. The model component is similar to us expept the use of LLM. The model is trained on VQA and segmentation datasets in addition to RefCOCO/+/g. Therefore, comparing with LISA puts our method at a disadvantage. Even so, our method remains competitive on RefCOCO/+ despite LISA utilizing an LLM and having a significantly larger model size. However, LISA excels on G-ref, where language expressions are more complex, thanks to the LLM's extensive vocabulary capacity. The work of [5] further improves performance thanks to their pre-training method, but the increase in training data in their work makes direct comparison difficult.

Table 5. Comparative results on RefCOCO/+/g in terms of oIoU.

Methods	Size	RefCOCO			RefCOCO+			G-Ref	
		Val	Test A	Test B	Val	Test A	Test B	$\text{Val}_{(U)}$	$\text{Test}_{(U)}$
ReLA [4]	225M	73.82	76.48	70.18	66.04	71.02	57.65	65.00	65.97
CGFormer [6]	251M	74.75	77.37	70.64	64.54	71.00	57.14	64.68	65.09
LISA [3]	7.3B	74.9	79.1	72.3	65.1	70.8	58.1	67.9	70.6
Ours	524M	74.80	78.02	69.96	**66.19**	**72.38**	56.75	65.34	67.08

References

[1] Yutao Hu, Qixiong Wang, Wenqi Shao, Enze Xie, Zhenguo Li, Jungong Han, and Ping Luo. Beyond one-to-one: Rethinking the referring image segmentation. In *Proc. of ICCV*, pages 4067–4077, 2023. 1

[2] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 1

[3] Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. Lisa: Reasoning segmentation via large language model. In *Proc. of CVPR*, pages 9579–9589, 2024. 3, 4

[4] Chang Liu, Henghui Ding, and Xudong Jiang. Gres: Generalized referring expression segmentation. In *Proc. of CVPR*, pages 23592–23601, 2023. 1, 2, 4

[5] Hanoona Rasheed, Muhammad Maaz, Sahal Shaji, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M Anwer, Eric Xing, Ming-Hsuan Yang, and Fahad S Khan. Glamm: Pixel grounding large multimodal model. In *Proc. of CVPR*, pages 13009–13018, 2024. 3

[6] Jiajin Tang, Ge Zheng, Cheng Shi, and Sibei Yang. Contrastive grouping with transformer for referring image segmentation. In *Proc. of CVPR*, pages 23570–23580, June 2023. 4