# SUPPLEMENT OF LEARNING TO FOCUS ON TARGET FOR WEAKLY SUPERVISED VISUAL GROUNDING

**Anonymous authors**
Paper under double-blind review

## 1 MORE DETAILS ABOUT THE LOSS FUNCTION

In this section, we will introduce more details about the self-taught regression loss and phrase reconstruction loss which following the RIF (Liu et al. (2021))

### 1.1 SELF-TAUGHT REGRESSION LOSS

The purpose of the self-taught regression loss is that the location for each phrase is not annotated, we need a loss function to help the model to find a location for each phrase. Here, we will use confident proposals from partially-trained models to supervise the location refinement. Specifically, given phrase $q_i$, we denote $\delta^{c*} = \{\delta i, m^{c*}\}$ where $\{\delta i, m^{c*}\}$ is the offset between proposal $o_m$ and the most confident proposal if their overlaps are larger than a threshold, otherwise we called is $\{\delta i, m^c\}$. Then the loss function is

$$L_{reg} = \sum_{i=1}^{N} (L_{sm}(\{\delta i, m^{c*}\}, \{\delta i, m^c\}) \tag{1}$$

where $L_{sm}$ is the smooth-L1 loss.

### 1.2 PHRASE RECONSTRUCTION LOSS

Given a noun phrase, we use a phrase reconstruction loss to provide model supervision. To use phrase reconstruction loss, we will first calculate a visual representation $z_i^c$ for each phrase $q_i$, then we can calculate a sequence of word distribution $y_i^c$ as below

$$y_i^c = \mathrm{LSTM}_{dec}([z_i^c, q_i]) \tag{2}$$

Then we use a standard sequence log loss $L_{log}$ to calculate the final loss function which can be written as

$$L_{rec} = \sum_{i=1}^{N} (L_{log}(y_i^c, q_i)) \tag{3}$$

## 2 MORE ABLATION STUDIES

### 2.1 INFLUENCE OF THE NUMBER OF TRANSFORMER ENCODER LAYERS

In this section, we conduct an ablation study to evaluate the influence of the number of transformer encoder layers in our visual and language encoders. Table.1 shows the results for different numbers of encoder layers. We observe that if the number of encoder layers is too small, it is insufficient to extract all of the information in the given image and language query. Once the number of layers is sufficient to extract the information, further increasing the number of encoder layers does not improve performance.

## 3 MORE DETAILS ABOUT THE ARCHITECTURE

In this section, we provide more details about our architecture

### 3.1 RESOLUTION ABOUT MULTI-LAYER FEATURES

We provide multi-layer features to our visual encoder, here, we provide the resolution for different layers' features

## 4 TRAINING DETAILS

In our experiments, we utilized two NVIDIA Tesla V100-sxm2 GPUs, each with 32GB of memory, for a total of 64GB of memory. All of the modules are end-to-end trained. We used the AdamW optimizer to optimize our architecture with an initial learning rate of 1e-5 and a weight decay of 1e-5. We used a batch size of 64 for all experiments. We applied the cosine learning rate schedule for all datasets. For data augmentation, we followed the same procedure as TransVG (Deng et al. (2021)), which included RandomBrightness, RandomContrast, RandomSaturation, ColorJitter, RandomResizeCrop, and RandomHorizonFlip. We trained our model for a total of 10 epochs for RefCOCO, RefCOCO+, and RefCOCOg datasets, and for 20 epochs for the ReferItGame and Flickr30K Entities datasets. These training parameters were chosen through experimentation to ensure that our model was optimized for performance on each dataset.

## 5 MORE VISUALIZATIONS

Here, we provide more visualizations of our model in Fig. 1

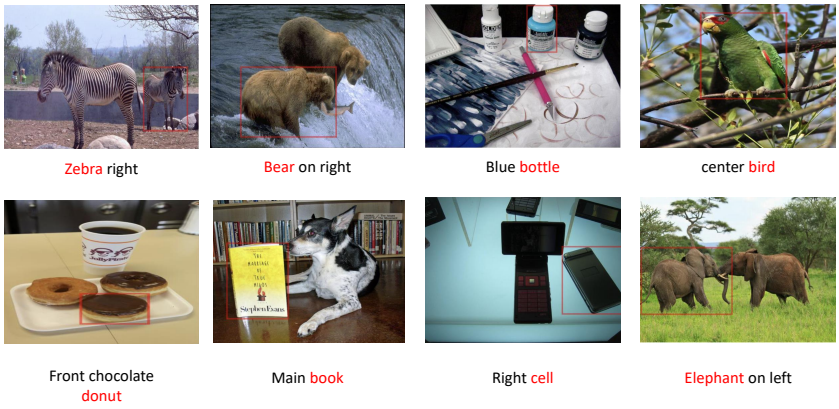


Figure 1: More visualizations of our model

Table 1: Ablation for the number of encoder layers in the visual and language encoder. The results show that increasing the encoder layer when the layer is not deep can improve the performance, and if we continue to increase the number of the encoder layers, the performance is not further improved; the best results are highlighted in bold

| visual encoder | language encoder | RefCOCO | | | RefCOCO+ | | | RefCOCOg | | | ReferItGame | Flicker30K |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | val | testA | testB | val | testA | testB | val-g | val-u | test-u | | |
| 3 | 3 | 50.13 | 51.11 | 48.68 | 38.16 | 41.34 | 36.90 | 45.94 | 43.30 | 47.65 | 39.71 | 56.34 |
| 3 | 6 | 52.68 | 53.91 | 51.35 | 40.95 | 42.78 | 38.47 | 47.06 | 45.18 | 49.76 | 42.29 | 59.82 |
| 6 | 6 | 53.86 | 55.07 | 52.67 | 41.29 | 44.88 | 38.23 | 49.18 | 47.33 | 51.11 | 44.39 | 60.37 |
| 6 | 12 | **54.78** | 55.71 | **53.15** | 42.25 | 45.88 | **39.18** | **50.48** | 48.54 | **51.25** | 45.27 | **61.78** |
| 12 | 12 | 54.73 | **55.82** | 53.01 | 41.28 | 44.79 | 38.11 | 49.58 | **48.55** | 50.19 | **45.39** | 61.25 |
| 12 | 24 | 53.27 | 54.49 | 51.13 | **42.46** | **46.98** | 38.77 | 50.19 | 48.30 | 50.79 | 44.95 | 60.98 |

## REFERENCES

Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. Transvg: End-to-end visual grounding with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1769–1779, 2021.

Yongfei Liu, Bo Wan, Lin Ma, and Xuming He. Relation-aware instance refinement for weakly supervised visual grounding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5612–5621, 2021.

Table 2: Input and output shape of different layers' features

| Layer | Input Shape |
|---|---|
| $\text{Layer}_5$ | $\frac{H \times W}{32}$ |
| $\text{Layer}_4$ | $\frac{H \times W}{16}$ |
| $\text{Layer}_3$ | $\frac{H \times W}{8}$ |