SGTR: End-to-end Scene Graph Generation with Transformer Supplementary Material

Overview of Appendixes

In this supplementary material, we present implementation details and more experiments results. First, more experiments and analysis (*e.g.*, analysis of overall recall performance, experiments using stronger long-tail learning strategy, and extra qualitative results) are described in Sec. 1. We provides the implementation details in Sec. 2. Moreover, we also present the details of graph assembling mechanism and loss function in Sec. 3 and Sec. 4.

In this supplementary material, we present implementation details and more experimental results. First, more experiments and analysis (*e.g.*, analysis of overall recall performance, experiments using stronger long-tail learning strategies, and extra qualitative results) are described in Sec. 1. We provide the implementation details in Sec. 2. Moreover, we also present the details of the graph assembling mechanism and loss function in Sec. 3 and Sec. 4.

1. More Experimental Results

1.1. Overall Recall Analysis

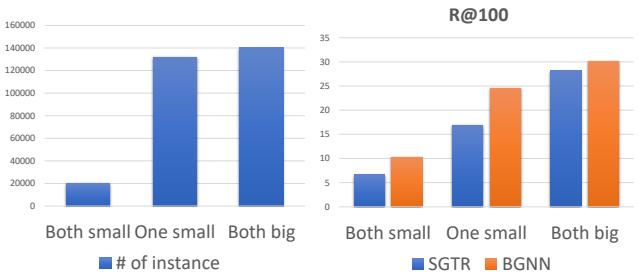

Figure 1. **Left)** The statistics of entity sizes in visual relationship based on the training set of Visual Genome. **Right)** The performance (Recall@100) of relationship detection with respect to different entity sizes. We define an entity as "small" if the area of its box is smaller than 64 $\times$ 64 pixels.

First, we analyze why our SGTR achieves lower overall recall performance than the traditional two-stage design. One potential reason is that the SGTR uses an ResNet-101 backbone (like DETR) for entity detection rather than the Faster-RCNN with an ResNet-101 FPN backbone. According to the experimental results of DETR [1], using the ResNet-101 backbone achieves lower performance than Faster-RCNN with the ResNet-101 FPN on small objects (21.9 versus 27.2 APs on the COCO dataset).

To confirm that, we further study how small entities influence visual relationship detection. We categorize the relationship instances in the Visual Genome dataset into three disjoint sets according to their entity sizes, and plot the statistics of the sets in Fig. 1. The result shows that more than half of the relationships consist of small entities. We also compare the performance of our method (R@100) on three relationship sub-sets with the two-stage approach BGNN [6] and report the performance in Fig. 1. The BGNN with the two-stage design outperforms the SGTR on the relationships with small entities by a large margin due to the inherent limitation of DETR on small entity detection.

The issue of recognizing small objects efficiently and effectively is still under active study [8, 11, 12, 16]. With more sophisticated transformer-based detectors, the overall recall of our method can be further improved.

1.2. Influence of Object Detector

We first compare the performances of DETR and faster RCNN on the head (H), body (B) and tail (T) predicates in Tab.1, which also groups the detector results into H/B/T object classes. The result indicates that DETR achieves a similar detection performance as Faster R-CNN for tail predicates. Moreover, we equip BGNN with DETR and as shown in Tab.2, its performance is on par with its Faster R-CNN counterparts. Finally, we also combine DETR optimized in SGTR with a finetuned BGNN. The results in Tab.2 clearly show the SGTR outperforms this baseline, demonstrating the benefit of our design.

	Recall (S-DETR / RCNN)		
	Rel Head	Rel Body	Rel Tail
Ent Head	49.3 / **50.5**	49.2 / **50.3**	**59.7** / 57.1
Ent Body	39.5 / **41.3**	**46.2** / 43.1	**44.9** / 43.7
Ent Tail	**44.3** / 41.7	**47.6** / 44.5	**45.1** / 38.5

Table 1. The entity detection performances of DETR and faster RCNN on the head (H), body (B) and tail (T) predicates.

M	mR@50/100	R@50/100	H	B	T	AP@det	mR@det
BGNN	8.4 / 9.8	**29.0 / 33.7**	**30.0**	11.2	2.2	29.9	41.1
$BGNN_D$	6.3 / 7.7	25.4 / 29.5	28.5	7.2	1.5	30.7	**41.2**
$BGNN_S$	6.7 / 8.3	25.6 / 29.7	28.5	7.2	3.0	**31.2**	41.0
SGTR†	**14.4 / 17.7**	24.8 / 28.5	21.7	**21.6**	**17.1**	**31.2**	41.0

Table 2. 'D' means DETR. 'S' means DETR optimized in SGTR. † means bi-level resampling, 'AP': mAP50, and 'mR': mRecall

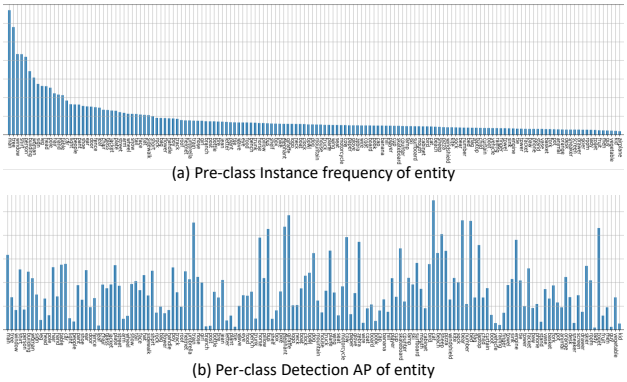

Figure 2. **The frequency statistics and per-class detection performance of entities on the VG dataset**. **(a)**: The frequency of each entity category; **(b)** The per-class entity detection performance (AP).

1.3. Experiments with Long-tail Learning Strategy

Method	mR@50/100	R@50/100	Head	Body	Tail
Ours	12.0 / 15.2	**24.6 / 28.4**	**28.2**	18.6	7.1
Ours*	**15.8 / 20.1**	20.6 / 25.0	21.7	**21.6**	**17.1**
Ours-P	18.9 / 22.0	**22.1 / 24.8**	**26.0**	20.9	**15.2**
$\textbf{Ours}^{\text{DisAlign}}$	13.7 / 16.8	**24.1 / 28.0**	**26.8**	21.7	8.9
$\textbf{Ours}^{\text{ACBS}}$	16.5 / 19.8	20.8 / 23.6	23.4	21.6	17.5
$\textbf{Ours}^{\text{cRT}}$	**18.8 / 21.6**	22.0 / 24.8	24.1	**22.1**	**18.1**

Table 3. **The performance of SGTR by adopting the advanced long-tail learning strategy on the VG dataset.** "*" denotes the bi-level sampling proposed in [6]; "P" denotes the modified bi-level sampling; The "cRT" denotes the decoupled retraining strategy on predicate classifier proposed by [4]; the "DisAlign" denotes the retraining strategy for logits adjustment proposed by [14]; the "ACBS" refers to the alternative class balanced retraining strategy proposed by [3].

Long-tail data distribution is a challenging issue in the SGG. To achieve better performance on the SGG task, we further apply several recent long-tail learning strategies in our model. [3, 4, 14], and report the performance in Tab. 3. We find that there exists a trade-off between overall and mean recall in the comprehensive experimental results. The advanced learning strategies enable our model to either achieve a higher mean recall or maintain a better trade-off between overall and mean recall.

• We adopt the retraining strategy, DisAlign [14], to adjust the predicate prediction logits via loss re-weighting with respect to the instance distribution of relationships. This method improves the performance trade-off between mR@100 and R@100 by increasing mean recall by **1.6** and achieving only a 0.4 performance drop on R@100.

• Moreover, we re-implement the alternative class-balanced retraining strategy (ACBS) [3], which achieves the SOTA on mean recall with the two-stage SGG model. The ACBS retrains both entity and predicate classifiers using class-based sampling. This method achieves high mean recall performance while sacrificing the performance of the overall performance.

• Finally, we apply the decoupled retraining strategy [4] on the predicate classifier of SGTR. We observe that using additional balanced-sampling retraining results in **6.4** performance gain on mR@100 with a 3.6 drop on R@100. This strategy outperforms the aforementioned methods in terms of mean recall performance.

We also observe that *using the re-balance idea on entity classifier does not bring too much performance benefit.* To investigate this phenomenon, we report the relationship between the per-class instance frequency and the performance of entity detection in Fig. 2. Despite the fact that the distribution of entity instances obeys the long-tail distribution, SGTR's entity detection performance is quite balanced, which means that the transformer-based detector is capable of tackling the data imbalanced scenario to some degree, and the additional re-balancing strategy is unnecessary.

1.4. Zero-shot Recall

We compare our method with VCTree-TDE in Tab.4, achieving a gain of 2.6 on zR@100. The result shows the generalization capability of SGTR to unseen relationships. We greatly appreciate R2's suggestions and will leave the further exploration to future work.

zR	BGNN†	$BGNN_D^\dagger$	DT2-ACBS†	VCTree-TDE	**SGTR**
@50	0.4	0.5	0.3	2.6	**2.4**
@100	0.9	0.7	0.5	3.2	**5.8**

Table 4. The performance of zero-shot relationship retrieval.

1.5. Model Selection

We present experiments for selecting the hyper-parameters (N_r, K) on validation set of VG in Tab.5. The performance saturates at $N_r = 160$ and $K = 3$.

N_r	mR@100	R@100	K	mR@100	R@100
100	16.1	27.5	1	16.4	26.2
130	16.3	27.3	2	17.3	28.2
160	**17.7**	**28.5**	3	**17.7**	**28.5**
190	16.1	27.0	4	17.5	28.2

Table 5. The performance of different choices of hyper-parameters N_r.

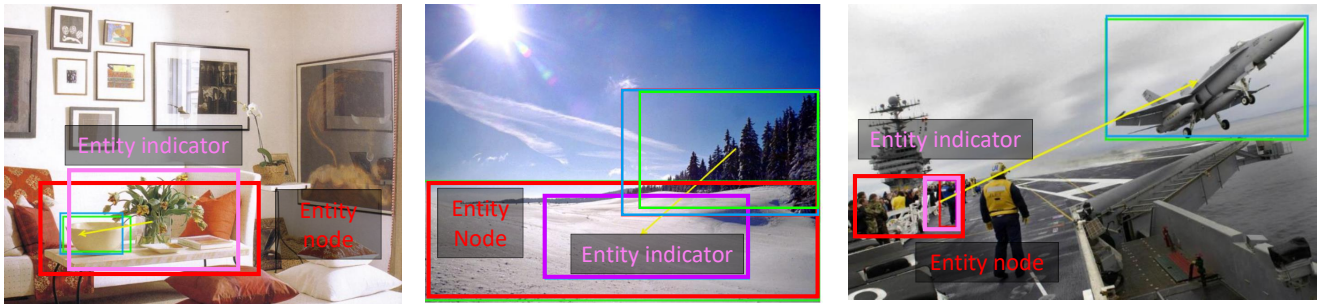

Figure 3. **Comparison of entity indicator of predicate node and entity node.** We use the different colored bounding boxes of entities to distinguish between the entity node (red) and the entity indicator (pink). The yellow arrow indicates the predicate between the entities. (best viewed in color)

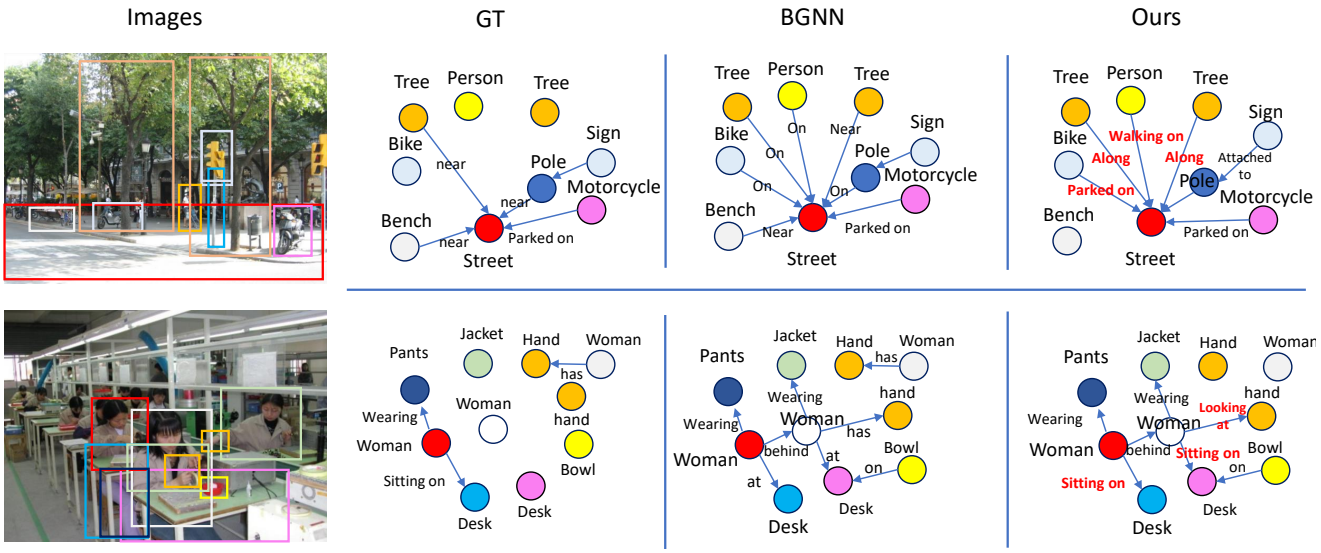

Figure 4. **Qualitative comparison between our method and BGNN† in SGG.** Both methods predict many reasonable relationships which are not annotated in GT. We mark the relationships of rare semantic predicate categories retrieved by SGTR in **red** (best viewed in color).

1.6. Qualitative Results

Visualization of Entity Indicator and Entity Node To demonstrate the effectiveness of our proposed graph assembling mechanism, we visualize predictions of entity indicators of our predicate representation and entity nodes after the graph assembling. As shown in Fig. 3, the entity indicator only provides a rough localization and classification of entities rather than precise bounding boxes. This information can be refined into more accurate entity results with graph assembling, which significantly improves the quality of the generated scene graph.

Prediction Comparision between Different Design We compare different method (*e.g.*, BGNN [6]) by visualizing the relationship predictions. In Fig. 4, we mark the different relationship predictions between BGNN and SGTR with red color. It shows that SGTR retrieves more relationships of *less frequent semantic categories* than BGNN.

2. Implementation Details

We implement our method based on the PyTorch 1.8 [9] and cvpods [15]. Our training process consists of two phases: *1) entity detector pre-training* and *2) SGTR joint training*.

Entity Detector Pre-training Phase: We follow the DETR training configuration to learn the entity detector on Visual Genome and Openimage datasets. We train the entity detector with the AdamW optimizer with a learning rate of 1e-5, a batch size of 16, and the model takes 100 epochs for convergence on 4 TITAN V GPUs. We use the same scale augmentation with DETR, resizing the input images such that the shortest side is at least 480 and at most 600 pixels, while the longest is at most 1000. The hyper-parameters of

Transformer (*e.g.,* number of attention heads, drop-out rate) are also kept the same with the DETR.

Joint Learning Phase: In the joint training phase, we adopt the same optimizer, learning rate, and batch size configuration as in the entity detector pre-training stage. In contrast with the two-stage SGG model, we refine the parameters of the detector in the joint learning rather than freezing the detector. We empirically observe that this refinement further improves the performance of entity detection. We train the SGTR for the Visual Genome dataset for 8.39e4 iterations without learning rate decay by using an early-stopping strategy. For the Openimage dataset, we train the model with 1.5e5 iterations, and the learning rate is decreased by 0.1x after 1e5 iterations.

We re-implement the previous two-stage methods (BGNN [6] and RelDN [13]) in our codebase by using the same configuration as the released codes. For fair comprasion, we replace the ResNeXt-101 FPN backbone with the ResNet-101 backbone to learn the model. We also apply the one-stage HOI works (AS-Net [2] and HOTR [5]) on the SGG. We use the hyper-parameters of the models reported by the authors. All experiments are conducted by training the model until convergence.

3. Correspondence Matrix for Assembling

For clarity, we will use the correspondence matrix between the subject entity and predicate $\mathbf{M}_s$ to introduce the details. The correspondence matrix is determined by the distance function, which takes the semantic outputs (*e.g.* bounding boxes $\mathbf{B}$, classification $\mathbf{P}$ of the entity detector and entity indicator of the predicate structural decoder) as input. Specifically, the distance function consists of two parts: spatial matching distance $d_{loc} \in \mathbb{R}^{N_r \times N_e}$ and category matching distance $d_{cls} \in \mathbb{R}^{N_r \times N_e}$, as shown in Eq. 1

$$\mathbf{M}^s = d_{loc}(\mathbf{B}_s, \mathbf{B}_e) \cdot d_{cls}(\mathbf{P}_s, \mathbf{P}_e) \tag{1}$$

Each element in the correspondence matrix $\mathbf{M}^s$ is calculated by pairing the N_r predicate predictions with N_e entity predictions, as shown in the following equations:

$$\begin{aligned} \mathbf{M}^s_{i,j} &= d_{loc}(\mathbf{B}_s(i), \mathbf{B}_e(j)) \cdot d_{cls}(\mathbf{P}_s(i), \mathbf{P}_e(j)) && (2)\\ &= d_{loc}(\mathbf{b}_{s,i}, \mathbf{b}_{e,j}) \cdot d_{cls}(\mathbf{p}_{s,i}, \mathbf{p}_{e,j}) && (3) \end{aligned}$$

where $i \in [0, N_e], j \in [0, N_r]$ for enumerating each pair between the predicate proposal and entity set.

Then we present the two components of the distance function, d_{loc} and d_{cls}. Specifically, the d_{loc} consists of the $d_{giou} \in \mathbb{R}^{N_r \times N_e}$ and $d_{center} \in \mathbb{R}^{N_r \times N_e}$, as show in Eq. 4.

$$d_{loc}(\mathbf{b}_s, \mathbf{b}_e) = \frac{d_{giou}(\mathbf{b}_s, \mathbf{b}_e)}{d_{center}(\mathbf{b}_s, \mathbf{b}_e)} \tag{4}$$

Concretely, the d_{giou} is the clipped GIOU of the entity and the indicator's bounding boxes, and d_{center} is the L1 distance between the bounding boxes' centers in Eq. 5, 6. The center points-based matching has also been adopted in HOI methods [2, 7, 10].

$$\begin{aligned} d_{giou}(\mathbf{b}_s, \mathbf{b}_e) &= \max(\min(\text{GIOU}(\mathbf{b}_s, \mathbf{b}_e), 0), 1) && (5)\\ d_{center}(\mathbf{b}_s, \mathbf{b}_e) &= ||[x_c, y_c]^s_i - [x_e, y_e]^e_i||_1 && (6) \end{aligned}$$

Here $[x_s, y_s]^s$ and $[x_e, y_e]^e$ are the normalized center coordinates of the bounding box in $\mathbf{b}_s$ and $\mathbf{b}_e$ respectively. For the d_{cls}, we use the cosine distance to calculate the similarity of the classification distribution between two entity predictions, as shown in following equation:

$$d_{cls}(\mathbf{p}_s, \mathbf{p}_e) = \frac{\mathbf{p}_s \cdot \mathbf{p}_e^\intercal}{||\mathbf{p}_s|| \cdot ||\mathbf{p}_e||} \tag{7}$$

4. Matching Cost and Loss Function

4.1. Triplets Matching Cost

We use the set-matching strategy to supervise the relationship predictions $\mathcal{T} = \{(\mathbf{b}^s_e, \mathbf{p}^s_e, \mathbf{b}^o_e, \mathbf{p}^o_e, \mathbf{p}_p, \mathbf{b}_p)\}$. To obtain the matches, we need to calculate and minimize the matching cost $\mathcal{C} \in \mathbb{R}^{N_r \times N_{gt}}$ between the N_r relationship predictions and the N_{gt} GT relationships. Concretely, the matching cost $\mathcal{C}$ includes two parts: the predicate cost $\mathcal{C}_p$ and the entity cost $\mathcal{C}_e$, as:

$$\mathcal{C} = \lambda_p \mathcal{C}_p + \lambda_e \mathcal{C}_e \tag{8}$$

where λ_p, λ_e is the coefficients of two cost terms.

The predicate cost, $\mathcal{C}_p(i, j)$ between the i-th predicate prediction and the j-th ground-truth relationship is computed according to the predicate classification distribution and location prediction in Eq. 9:

$$\mathcal{C}_p(i, j) = \exp\left(-\mathbf{p}^{gt}_{p,j} \cdot \mathbf{p}^\intercal_{p,i}\right) + \|\mathbf{b}_{p,i} - \mathbf{b}^{gt}_{p,j}\|_1 \tag{9}$$

where $\mathbf{p}_{p,i} \in \mathbb{R}^{1 \times C_p}$ is the i-th $\mathbf{P}_p$, and $\mathbf{p}^{gt}_{p,j} \in \mathbb{R}^{1 \times C_p}$ is the one-hot predicate label of the j-th ground truth relationship. Similarly, $\mathbf{b}_{p,i} \in \mathbb{R}^{1 \times 4}$ and $\mathbf{b}^{gt}_{p,j} \in \mathbb{R}^{1 \times 4}$ are the center coordinates of the entity pair from the i-th relationship prediction and the j-th ground truth relationship, respectively.

The entity cost $\mathcal{C}_e(i, j)$ between the i-th predicted relationship and j-th ground-truth relationship is given by:

$$\begin{aligned} \mathcal{C}_e(i, j) =& w_{giou} \cdot \prod_{\star=\{s,o\}} \exp\left(-d_{giou}(\mathbf{b}^\star_{e,i}, \mathbf{b}^\star_{gt,j})\right) && (10)\\ &+ w_{l1} \cdot \sum_{\star=\{s,o\}} ||\mathbf{b}^\star_{e,i} - \mathbf{b}^\star_{gt,j}||_1 && (11)\\ &+ w_{cls} \cdot \prod_{\star=\{s,o\}} \exp\left(-\mathbf{p}^{(\star,gt)}_{e,j} \cdot \mathbf{p}^{\star}_{e,i}{}^\intercal\right) && (12) \end{aligned}$$

where the $\mathbf{b}^{\star}_{e,i}$ and $\mathbf{p}^{\star}_{e,i}$ are the i-th subject/object entity box and category distribution of relationship $\mathcal{T}$ after graph assembling, respectively. The $\mathbf{b}^{\star}_{gt,j}$ and $\mathbf{p}^{(\star,gt)}_{e,j}$ is the subject/object bounding boxes and one-hot entity category label from j-th ground truth relationships.

4.2. Loss Calculation

Our total loss $\mathcal{L}$ is composed of entity detector loss $\mathcal{L}^{enc}$ and predicate node generator loss $\mathcal{L}^{pre}$:

$$\mathcal{L} = \mathcal{L}^{enc} + \mathcal{L}^{pre} \tag{13}$$

The entity detector loss $\mathcal{L}^{enc}$ is calculated independently by following the same design in DETR [1].

The loss of predicate node generator $\mathcal{L}^{pre}$ is determined by the prediction and ground-the relationships according to the matching index $\mathbf{I}^{tri} \in \mathbb{N}^{N_{gt}}$. The $\mathbf{I}^{tri}$ stores the index of matched predictions for each GT relationship. Specifically, the predicate node generator loss consists of entity indicator loss $\mathcal{L}^{pre}_{i}$ and predicate sub-decoder loss $\mathcal{L}^{pre}_{p}$:

$$\mathcal{L}^{pre} = \mathcal{L}^{pre}_{i} + \mathcal{L}^{pre}_{p} \tag{14}$$

For loss of predicate sub-decoder loss $\mathcal{L}^{pre}_{p}$, we have:

$$\mathcal{L}^{pre}_{p} = \sum_{i}^{N_{gt}} \left(\|\mathbf{b}_{p,\mathbf{I}^{tri}(i)} - \mathbf{b}^{gt}_{p,i}\|_1 + \text{CE}\left(\mathbf{p}_{p,\mathbf{I}^{tri}(i)}, \mathbf{p}^{gt}_{p,i}\right)\right) \tag{15}$$

where $\mathbf{b}^{gt}_{p,i}$ and $\mathbf{b}_{p,\mathbf{I}^{tri}(i)}$ is the entity center coordinates of the GT relationship and prediction, respectively. The CE denotes the cross entropy loss between the predicate classification $\mathbf{b}_{p,\mathbf{I}^{tri}(i)}$ and the GT predicate category one-hot vector $\mathbf{p}^{gt}_{p,i}$.

For the entity indicator loss $\mathcal{L}^{pre}_{i}$:

$$\mathcal{L}^{pre}_{i} = \sum_{\star=\{s,o\}} \left(\mathcal{L}^{\star}_{ent\_loc} + \mathcal{L}^{\star}_{ent\_cls}\right) \tag{16}$$

where $\star = \{s, o\}$ indicates the subject/object role of an entity in relationships. The indicator loss is composed of two factors, $\mathcal{L}^{\star}_{ent\_loc}$ and $\mathcal{L}^{\star}_{ent\_cls}$, for two types of semantic representation: bounding boxes $\mathbf{b}_s, \mathbf{b}_o$ and classification $\mathbf{p}_s, \mathbf{p}_o$.

$$\mathcal{L}^{\star}_{ent\_loc} = \sum_{i}^{N_{gt}} \left(\|\mathbf{b}_{\star,\mathbf{I}^{tri}(i)} - \mathbf{b}^{gt}_{\star,i}\|_1 \right. \tag{17}$$

$$\left. +1 - \text{GIOU}\left(\mathbf{b}_{\star,\mathbf{I}^{tri}(i)}, \mathbf{b}^{gt}_{\star,i}\right)\right) \tag{18}$$

$$\mathcal{L}^{\star}_{ent\_cls} = \sum_{i}^{N_{gt}} \text{CE}\left(\mathbf{p}_{\star,\mathbf{I}^{tri}(i)}, \mathbf{p}^{gt}_{\star,i}\right) \tag{19}$$

The $\mathcal{L}^{\star}_{ent\_loc}$ is computed by the $L1$ distance and GIoU loss between the bounding box outputs $\mathbf{b}_{\star,\mathbf{I}^{tri}(i)}$ and ground-truth entity boxes $\mathbf{b}^{gt}_{\star,i}$. The $\mathcal{L}^{\star}_{ent\_cls}$ is calculated from the cross entropy loss of classification prediction $\mathbf{p}_{\star,\mathbf{I}^{tri}(i)}$ according to ground-truth entities'category $\mathbf{p}^{gt}_{\star,i}$.

5. Social Impacts

Our method has no direct potential negative impact, one possible negative impact is that SGG may serve as a base module for surveillance abuse.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020. 1, 5

[2] Mingfei Chen, Yue Liao, Si Liu, Zhiyuan Chen, Fei Wang, and Chen Qian. Reformulating hoi detection as adaptive set prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9004–9013, 2021. 4

[3] Alakh Desai, Tz-Ying Wu, Subarna Tripathi, and Nuno Vasconcelos. Learning of visual relations: The devil is in the tails. *arXiv preprint arXiv:2108.09668*, 2021. 2

[4] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *International Conference on Learning Representations(ICLR)*, 2019. 2

[5] Bumsoo Kim, Junhyun Lee, Jaewoo Kang, Eun-Sol Kim, and Hyunwoo J Kim. Hotr: End-to-end human-object interaction detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 74–83, 2021. 4

[6] Rongjie Li, Songyang Zhang, Bo Wan, and Xuming He. Bipartite graph network with adaptive message passing for unbiased scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11109–11119, 2021. 1, 2, 3, 4

[7] Yue Liao, Si Liu, Fei Wang, Yanjie Chen, Chen Qian, and Jiashi Feng. Ppdm: Parallel point detection and matching for real-time human-object interaction detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 482–490, 2020. 4

[8] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3651–3660, 2021. 1

[9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu

Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 3

[10] Tiancai Wang, Tong Yang, Martin Danelljan, Fahad Shahbaz Khan, Xiangyu Zhang, and Jian Sun. Learning human-object interaction detection using interaction points. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4116–4125, 2020. 4

[11] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based detector. *arXiv preprint arXiv:2109.07107*, 2021. 1

[12] Zhuyu Yao, Jiangbo Ai, Boxun Li, and Chi Zhang. Efficient detr: Improving end-to-end object detector with dense prior. *arXiv preprint arXiv:2104.01318*, 2021. 1

[13] J. Zhang, M. Elhoseiny, S. Cohen, W. Chang, and A. Elgammal. Relationship proposal networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5226–5234, 2017. 4

[14] Songyang Zhang, Zeming Li, Shipeng Yan, Xuming He, and Jian Sun. Distribution alignment: A unified framework for long-tail visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2361–2370, June 2021. 2

[15] Benjin Zhu*, Feng Wang*, Jianfeng Wang, Siwei Yang, Jianhu Chen, and Zeming Li. cvpods: All-in-one toolbox for computer vision research, 2020. 3

[16] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 1