Supplementary Material for Structured Sparse R-CNN for Direct Scene Graph Generation

Yao Teng Limin Wang ✉
State Key Laboratory for Novel Software Technology, Nanjing University, China
tengyao19980325@gmail.com, lmwang@nju.edu.cn

A. Technical Details of Modules

Dynamic convolution [10] is used for the feature extraction. We first perform the dynamic convolution on the subject box and object box with the relation vector to obtain the relation feature. Then, we utilize the same dynamic convolution to perform convolution on the union of the two boxes with the relation feature. The specific process for each dynamic convolution is as follows:

$$\begin{aligned}
\mathcal{F}_1 &= \sum_{i=1}^{d_r} x_i \cdot W_i^{(1)}, V_1 = \mathrm{ReLU}(\mathrm{LN}(\mathrm{Conv}_{1\times1}(V_0, \mathcal{F}_1))), \\
\mathcal{F}_2 &= \sum_{i=1}^{d_r} x_i \cdot W_i^{(2)}, V_2 = \mathrm{ReLU}(\mathrm{LN}(\mathrm{Conv}_{1\times1}(V_1, \mathcal{F}_2))), \\
v_2 &= \mathrm{Flatten}(V_2), y = \mathrm{LN}(x + \mathrm{ReLU}(\mathrm{LN}(W_v v_2))),
\end{aligned} \tag{1}$$

where $x \in \mathbb{R}^{d_r}$ denotes one relation vector. $V_0 \in \mathbb{R}^{C\times H\times W}$ is the flattened features from roi pooling. $W_i^{(1)} \in \mathbb{R}^{K_1\times C}$, $W_i^{(2)} \in \mathbb{R}^{C\times K_1}$ and $W_v \in \mathbb{R}^{d_r\times CHW}$ are linear transformation matrices. K_1 denotes the number of filters. $\mathrm{Flatten}(\cdot)$ means reshaping a matrix to the vector form. $\mathrm{Conv}_{\mathrm{h}\times\mathrm{w}}(A, B)$ means $h \times w$ convolution on feature map A with filters B. Bias terms are ignored. Furthermore, since the input V_0 for each dynamic convolution has a fixed size (e.g. $H = W = 7$), we can take W_v as a special case of $H \times W$ convolutions. Thus, following [9], we decouple W_v into depthwise $H \times W$ convolutions and 1×1 convolutions with intermediate expansion to high dimensions (e.g. 2048). Since the dimensionality of object features is larger than the channel of feature maps, this operation can reduce the quantity of parameters.

As for the relation classification, we adopt the multi-branch structure similar to [12, 15] and utilize the fusion operator in [5]. The specific process is as follows:

$$\begin{aligned}
G_s &= \mathrm{LN}(W_{r_1}^s F_s), \quad G_o = \mathrm{LN}(W_{r_1}^o F_o), \\
G_{so} &= \mathrm{ReLU}(G_s + G_o) - (G_s - G_o) \odot (G_s - G_o), \\
f_r' &= W_{cls}^r \mathrm{ReLU}(G_{so}), \quad f_r'' = f_r + f_r',
\end{aligned} \tag{2}$$

where f_r is the logit calculated from the main part of our relation detection. f_r'' is the final logit for relation classification. $W_{r_1}^s, W_{r_1}^o \in \mathbb{R}^{d\times d}$ and $W_{cls}^r \in \mathbb{R}^{N_{cls}\times d}$ are linear transformation matrices. $\odot$ represents the element-wise multiplication.

B. Two-stage Triplet Label Assignment

The triplet detector includes both object pair detection and relation recognition. Its performance heavily depends on object pair detection component. However, the existing SGG benchmarks often contain sparse annotations and fail to cover all object pair instances. To increase the recall of object pair detection, we need to generate some virtual object pairs as pseudo-labels. Thus, we use an extra object detector to yield a set of detected boxes, serving as the candidates to form virtual object pairs. Even though these generated object pairs are not in the SGG ground-truth, they could be used to train the object pair detection component under the object bounding box and classification loss.

As shown in Fig. 1, the procedure of two-stage triplet label assignment is as follows: *first*, the object label assignment is conducted between predicted objects and the ground-truth objects, and the first-stage label assignment is conducted between predicted triplets and the ground-truth triplets; *second*, the predicted objects that match the ground-truth in previous label assignment are replaced by the ground-truth. Also, the classification scores of the predicted objects that do not match the ground-truth in the label assignment are replaced by hard-labels of the background, but *their boxes are not the replaced*; *third*, these objects are organized into a set of object pairs, *i.e*. the pseudo-label set U; *forth*, predicted object pairs are directly taken from the predicted triplets that do not match the ground-truth; *last*,

✉: Corresponding author.

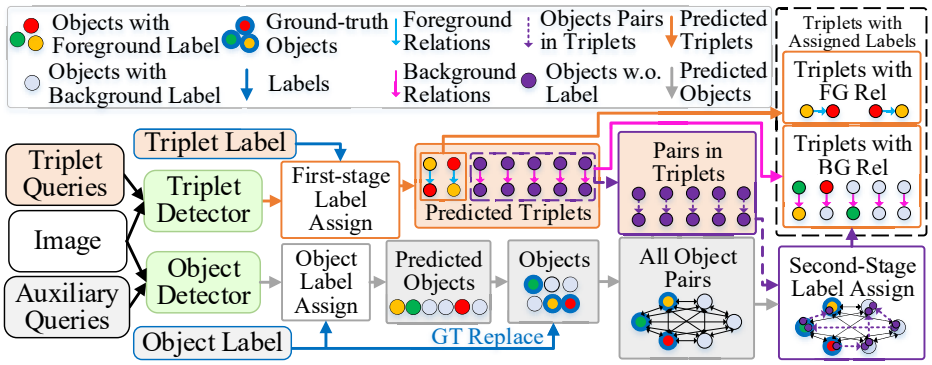


Figure 1. The procedure of two-stage triplet label assignment.

the label assignment is carried out between predicted object pairs and pseudo-label set. Overall, in our method, when assigning ground-truth labels to predicted triplets, instead of using the padding in original set prediction [1], we employ the label assignment with pseudo-labels.

The key to our label assignment is the pseudo-label set U. The detections from Siamese Sparse R-CNN are paired with each other to form the set U, thereby making U many elements. Actually, due to the huge gap between the magnitude of the triplet queries and pseudo-labels ($O(N)$ versus $O(N^2)$), taking U directly for bipartite matching [3] to get the optimal solution costs too much computation resources. Therefore, in practice, we adopt an alternative solution to reduce the computation complexity.

Inspired by [11], we first consider a relaxed strategy which takes the first K small matches for each query. After getting the $M \times K$ pseudo-label candidates, where M indicates the number of queries, we remove the duplicates and get C remaining candidates. If $C > M$, we accept these C pseudo-labels as the candidate set for matching, otherwise we increase K. In practice, we use a binary search to determine the minimum of K, thereby enabling a lower computation complexity than the original algorithm.

The total loss for our whole framework is as follows:

$$\mathcal{L} = \mathcal{L}_F + \mathcal{L}_B + \mathcal{L}_{obj}, \tag{3}$$

where $\mathcal{L}_F$ and $\mathcal{L}_B$ is to train the triplet detector with triplet queries. $\mathcal{L}_{obj}$ is the loss for training Siamese Sparse R-CNN.

C. Application of Logit Adjustment

Logit adjustment [7] is proposed to tackle long-tailed distribution, and it can be implemented in a post-hoc manner. The post-hoc logit adjustment is performed by subtracting the logarithm of category frequency multiplied by a tuning parameter τ from the logit for classification. Formally, TDE and TE [12] are similar to this approach. However, compared with logit adjustment, TDE and TE [12] lack the

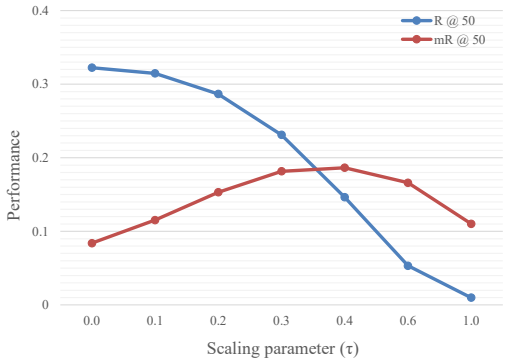


Figure 2. The performance of our method on R@50 and mR@50 under various scaling parameters.

temperature scaling τ, which is critical to the performance. Moreover, they calculate statistics for each instance, which costs more resources.

First, we present the selection of τ for our method. As depicted in Fig. 2, we find that the best performance on mR@50 is when τ is between 0.3 and 0.4. However, when $\tau = 0.4$, the performance of the head classes is quite poor, and the performance drop on Recall@50 is severe. Therefore, we choose $\tau = 0.3$ as the best choice.

Then, we present the performance of other methods equipped with logit adjustment in Tab. 1. In these methods, we find $\tau = 1.75$ is a relatively suitable choice.

D. Visualization of Model Prediction

Shown in Fig. 3, we present more results of our model compared with another method. We find that our method yields more precise relations where the same entities are detected. Considering all qualitative analysis in this paper, our framework performs well on various scenes.

Moreover, we visualize the change of an object pair in one image. Shown in Fig. 4, the initial two learnable boxes almost cover the whole image. As the network gets deeper, the two boxes gradually focus on the main part of the ob-

Model	R@20	R@50	R@100	zR@20	zR@50	zR@100	mR@20	mR@50	mR@100
MOTIFS$^{\dagger}_{\text{LA}}$	16.2	21.2	24.5	1.1	1.7	2.9	10.7	13.7	16.1
Transformer$^{\dagger}_{\text{LA}}$	16.9	21.8	24.9	1.1	2.0	3.1	10.5	13.7	16.2

Table 1. Other methods with post-hoc approach at SGDet on VG. LA: logit adjustment. The reimplemented model is denoted by the superscript †.

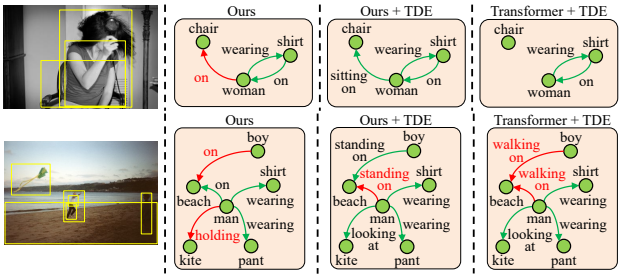


Figure 3. Results of Recall@100 from our model and another method. We present the directed edges matching the ground-truth pairs, and mark the misclassified relations in red.

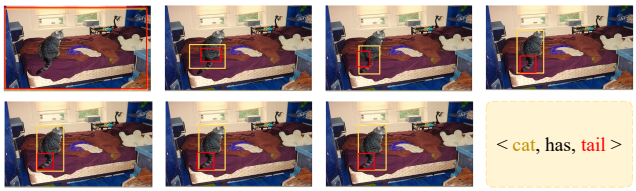


Figure 4. Visualization of an object pair changing with the depth of the triplet detector. The two boxes are marked in orange and red. The direction is from left to right.

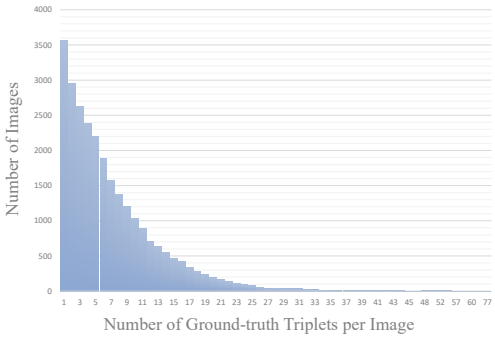


Figure 5. The relationship between the number of images and the number of ground-truth triplets per image.

jects (*e.g.* the cat and its tail). Finally, the queries detect the correct object pair with one relation prediction.

E. More Results and Analysis

In this part, we will analyze how our method outperforms the state-of-the-art such as transformer [14].

Model	mR@50	mR@100
Transformer$_{\text{LA}}$ [14]	6.0	7.0
Ours$_{\text{LA}}$	**6.3**	**7.6**

Table 2. The average mean Recalls (%) of various methods with logit adjustment [7] on images with more than 20 labeled triplets. LA: logit adjustment [7].

Model	R@50	R@100
Transformer [14]	26.2	32.9
Ours	26.9	32.3
Ours*	**28.4**	**33.8**

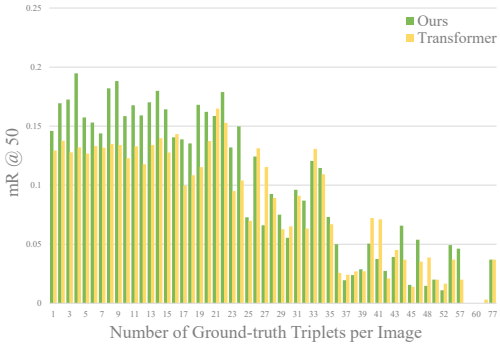
Table 3. The average Recalls (%) of various methods on images with more than 20 labeled triplets. * refers to the 800 queries.

E.1. Influence of Annotation Quantity

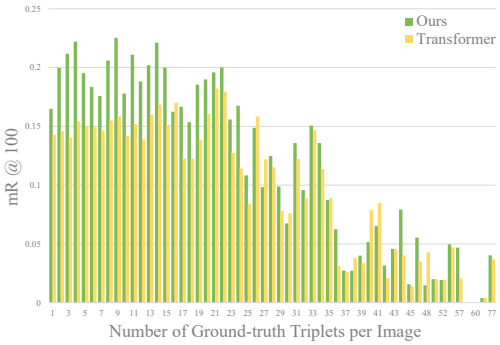
The mechanism of our method for SGG is based on a limited number of triplet queries. Thus, an intuitive idea is to evaluate the methods on images with different number of ground-truth triplets. As depicted in Fig. 5, most images contain fewer than 20 labeled triplets.

Then, we compare our model with transformer with logit adjustment [7] on mean Recalls [13] in Fig. 6a and Fig. 6b. Consistently, our model outperforms transformer by a great margin on most images with few labeled triplets. As for the images with more than 20 labeled triplets, their performance is not easy to distinguish. Therefore, we calculate the average performance on images with more than 20 labeled triplets. Shown in Tab. 2, our model still outperforms transformer [14] on various mean Recalls with logit adjustment [7].

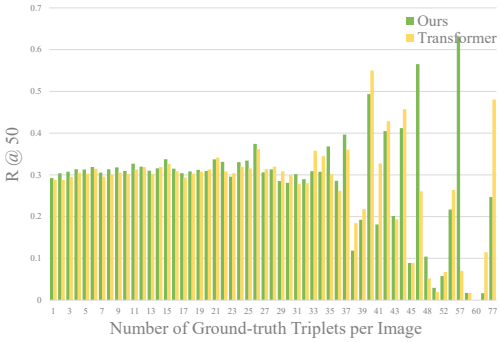
We also compare our model with transformer directly on Recalls [6] in Fig. 6c and Fig. 6d. In line with the performance on mean Recalls, our performance on most images with fewer than 20 triplets is slightly better than transformer. Consistently, we calculate the average Recalls of various methods on images with more than 20 labeled triplets in Tab. 3. In this table, we find that transformer can outperform our method with a little margin on R@100 when evaluated on images with more triplets. Thus, we evaluate our model with more queries (e.g. 800), and its performance on R@100 is quite better.



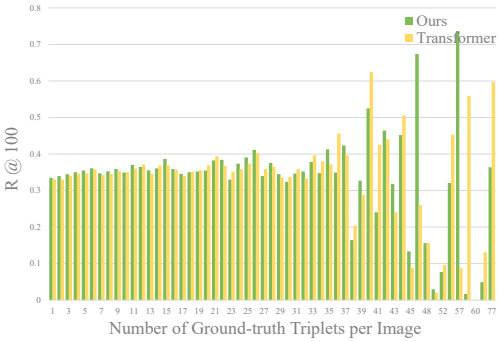
(a) The performance on mean Recall@50.



(b) The performance on mean Recall@100.

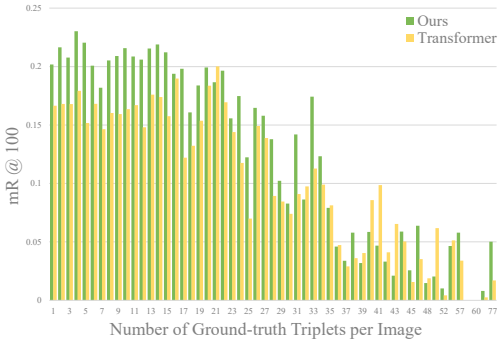


(c) The performance on Recall@50.

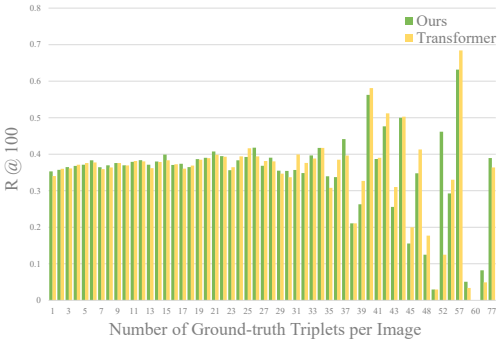


(d) The performance on Recall@100.

Figure 6. The performance of our model and transformer.



(a) The performance on mean Recall@100.



(b) The performance on Recall@100.

Figure 7. The performance of our model and transformer.

E.2. Influence of Object Detector

A straightforward idea is investigating how to set the magnitude of queries if both the paradigms achieve similar performance. Thus, in this part, we compare the transformer [14] with vanilla Sparse R-CNN [10] as the object detector to our method on VG [2].

In Tab. 4, we can see that our model with 800 queries can be comparable with the transformer based on vanilla Sparse R-CNN on Recalls. However, our model performs better than transformer on mean Recalls, thereby demonstrating the effectiveness of our relation modeling. Therefore, it is meaningful to design specific structures directly for relation features. Although these structures may cost much computation resources, the sparse modeling on triplets can greatly alleviate this problem. Moreover, the inference speed of our model is faster than that of the dense detector.

Then, we select AP_{50} for evaluation because all the metrics of SGG utilize an IoU of 0.5 for distinguishing true positive and false positive samples. The performance of object detection is shown in Tab. 5. It is obvious that Sparse R-CNN performs better than Faster R-CNN without additional context encoder. However, when both of them are

Model	Object Detector	R@20	R@100	mR@20	mR@100	Speed
Transformer† [14]	Sparse R-CNN [10]	26.1	38.2	5.7	9.3	0.37
Transformer†$_{\text{LA}}$ [14]	Sparse R-CNN [10]	17.1	27.2	10.5	17.3	0.37
Ours*	-	26.1	38.4	6.2	10.3	**0.32**
Ours*$_{\text{LA}}$	-	18.2	27.3	**13.7**	**22.5**	**0.32**

Table 4. Comparisons with the state-of-the-art methods at SGDet on Visual Genome (VG). * refers to the 800 queries. LA: logit adjustment [7]. The reimplemented model is denoted by the superscript †.

Object Detector	Context Encoder	AP_{50}
Faster R-CNN [8]	-	28.1
Sparse R-CNN [10]	-	29.0
Faster R-CNN [8]	Transformer [14]	30.2
Sparse R-CNN [10]	Transformer [14]	30.4

Table 5. The performance (%) of object detectors on VG [2].

equipped with transformer encoder and used for relation detection, their performance is similar. We speculate that the setting in [12] forcing the object detector to provide only few high-confident candidates narrows the performance gap between these models. Furthermore, Sparse R-CNN itself depend heavily on context information, and its performance may be similar to that of Faster R-CNN under the same case of utilizing the context.

Consistent with Appendix E.1, we also investigate the detailed performance of multi-stage model based on transformer and Sparse R-CNN, and conduct the same comparison with our method with 800 queries. As shown in Fig. 7, the difference between the two method is quite similar with that in Fig. 6, which suggests the performance gain of our method is mostly from the images with few annotations.

F. Societal Impact

Scene graph generation (SGG) is a traditional visual scene understanding task and we adopt the open datasets, Visual Genome [2] and OpenImage [4], so there is no negative social impact if the methods in the area of SGG are used properly.

G. Limitation

Our method adopts a unified framework for SGG with many fresh operations such as dynamic convolution. Therefore, compared with the previous multi-stage methods, it requires more computation resources for training. 8 RTX 2080ti with 11G GPU memory are necessary for our model with 300 queries. As for training the model with more than 300 queries (*e.g.* 800 queries), 8 Tesla V100 with 32G GPU memory are needed. However, the testing phase only needs 1 RTX 2080ti.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020. 2

[2] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 123(1):32–73, 2017. 4, 5

[3] Harold W. Kuhn. The hungarian method for the assignment problem. In *50 Years of Integer Programming*, pages 29–47. Springer, 2010. 2

[4] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper R. R. Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The open images dataset V4: unified image classification, object detection, and visual relationship detection at scale. *CoRR*, abs/1811.00982, 2018. 5

[5] Xin Lin, Changxing Ding, Jinquan Zeng, and Dacheng Tao. Gps-net: Graph property sensing network for scene graph generation. In *CVPR*, pages 3743–3752, 2020. 1

[6] Cewu Lu, Ranjay Krishna, Michael S. Bernstein, and Fei-Fei Li. Visual relationship detection with language priors. In *ECCV*, pages 852–869, 2016. 3

[7] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *ICLR*, 2021. 2, 3, 5

[8] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 1137–1149, 2017. 5

[9] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018. 1

[10] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, and Ping Luo. Sparse R-CNN: end-to-end object detection with learnable proposals. In *CVPR*, pages 14454–14463, 2021. 1, 4, 5

[11] Jing Tan, Jiaqi Tang, Limin Wang, and Gangshan Wu. Relaxed transformer decoders for direct action proposal generation. In *ICCV*, pages 13506–13515, 2021. 2

[12] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training. In *CVPR*, pages 3713–3722, 2020. 1, 2, 5

[13] Kaihua Tang, Hanwang Zhang, Baoyuan Wu, Wenhan Luo, and Wei Liu. Learning to compose dynamic tree structures for visual contexts. In *CVPR*, pages 6619–6628, 2019. 3

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. 3, 4, 5

[15] Ji Zhang, Kevin J. Shih, Ahmed Elgammal, Andrew Tao, and Bryan Catanzaro. Graphical contrastive losses for scene graph parsing. In *CVPR*, pages 11535–11543, 2019. 1