Compositional Feature Augmentation for Unbiased Scene Graph Generation
************* Supplementary Document *************

Appendix

This supplementary document is organized as follows:

- The datasets and implementation details mentioned in Sec. 4.1 are shown in Sec. A.
- The hierarchical clustering process and results mentioned in Sec. 3.2.1 are shown in Sec. B.
- A more detailed discussion of prior knowledge mentioned in Sec. 4.2 is shown in Sec. C.
- More experimental comparisons and analyses are shown in Sec. D
- The limitations of the proposed method are shown in Sec. E
- The potential negative societal impacts of the proposed method are shown in Sec. F

A. Datasets and Implementation Details

VG Dataset. We followed the widely-used strategy [12] to split dataset (*i.e.*, 75K/32K images for train/test, 150 object categories and 50 predicate categories). Besides, 5K images within the training set are sampled as val set following [10]. All predicate categories are divided into three groups {head, body, tail} based on the number of samples as [9].
GQA Dataset. We followed the prior work [3] to split dataset (*i.e.*, 70%/30% of the images for train/test, 200 object categories and 100 predicate categories). Similarly, 5K images within the training set are sampled as val set. We utilized the same group category ratio as VG to divide all predicate categories into {head, body, tail} groups.
Implementation Details. Following the training protocol in prior SGG works [10], we adopted the object detector Faster R-CNN with the ResNeXt-101-FPN [8] backbone trained by [10] to detect all the bounding boxes and extract their visual features. The parameters of the backbone were kept frozen during the training. This detector could achieve 28.14 mAP on the VG test set (*i.e.*, using 0.5 IoU threshold for evaluation). **To avoid messy hyperparamter tuning, most of the hyperparamters follow preivous works**.

Algorithm 1: Hierarchical Clustering

Input: Entity category set $\mathcal{C} = \{ c_i \mid i = 1,2,\cdots, N\}$, similarity measure function Sim, and cluster number K.
Output: Clusters $\mathcal{S} = \{s_i | i = 1, 2, \cdots, K\}$.

```
/* Initialize each entity category as a cluster */
for i = 1, 2, ..., N do
    s_i = {c_i}
/* Initialize the size of each cluster l_i and the similarity matrix A_sim */
for i = 1, 2, ..., N do
    l_i = 1
    for j = 1, 2, ..., N do
        A_sim(i, j) = Sim(s_i, s_j)
        A_sim(j, i) = A_sim(i, j)
/* Merge the two most similar cluster until the number of clusters is smaller than K */
while LEN(S) > K do
    s_i, s_j = SELECT_MAX(A_sim(i, j)/(l_i + l_j))
    s_i = MERGE(s_i, s_j)
    S = S - s_j
    l_i = l_i + l_j + 1
    UPDATE(A_sim)
```

More specifically, the hyperparameters λ and γ were set to 0.07 and 0.7 in CFA (*c.f.* Sec. 3.2). The weights of pattern, context and semantic similarity were 1.0, 1.0 and 0.01 in intrinsic-CFA (*c.f.* Sec. 3.2.1). The threshold σ was set to 0.5 in spatial restriction (*c.f.* Sec. 3.2.1). The β was set to 0.1 to regulate the loss during training (*c.f.* Sec. 3.3). In this paper, SGD optimizer was used to train the model. The batch size was set to 12 and the initial learning rate was set to 0.01. After the performance on the val set reached the plateau period, the learning rate would be decayed by 10 for two times. All experiments were carried out with PyTorch and NVIDIA 2080Ti GPU.

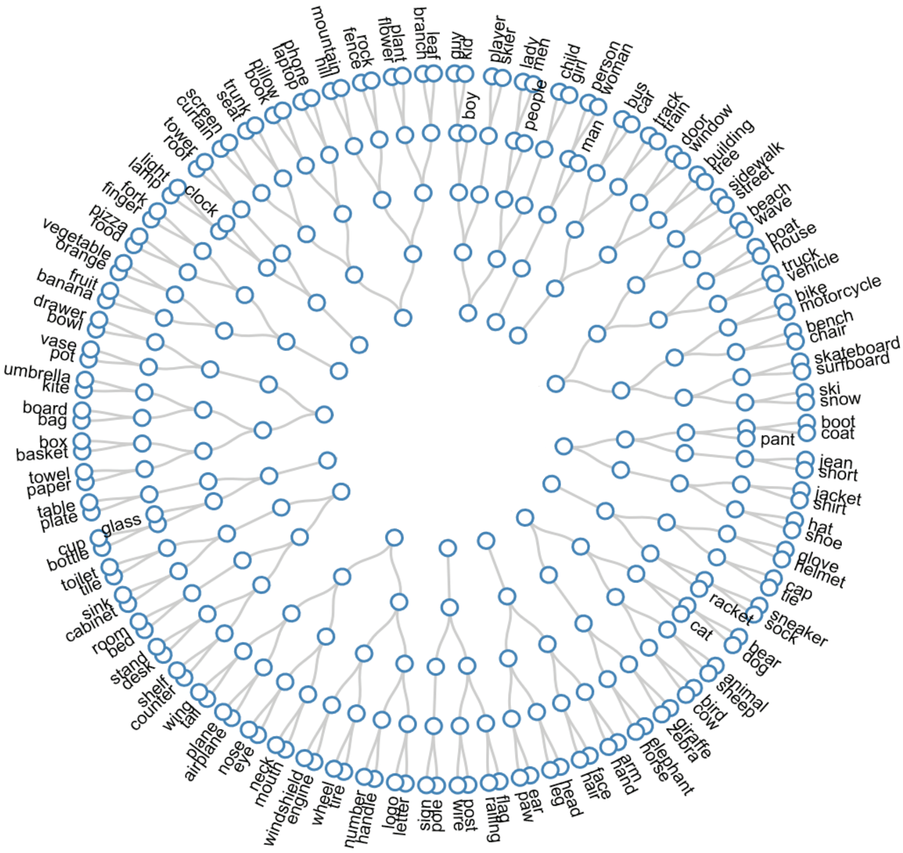

Figure 7: The results of hierarchical clustering on VG [4].

B. Hierarchical Clustering in Extrinsic-CFA

As mentioned in Sec. 3.2.1, we use hierarchical clustering to mine the correlations between entity categories, and the specific implementation is displayed in Algorithm 1. Treating clusters as a collection of multiple entity categories, the pattern and context similarity can be transferred directly from class-level to cluster-level. For semantic similarity, we take the mean similarities for the categories between clusters.

After initializing the variables, we perform clustering by the following steps in each iteration. First, we select the two most similar clusters (SELECT_MAX) based on similarity matrix A_{sim} and size of cluster l. Since the number of different relation triplet in the whole dataset is extremely imbalanced, we follow [11] to introduce a penalty term l (*i.e.*, the size of the cluster) to avoid dominating clustering by entity categories with a large proportion of samples. Then, we merge the entity categories in cluster s_j into s_i (MERGE), remove cluster s_j, and update the size of cluster l_i. Since s_i has changed, we recalculate the similarity of other clusters to it and then update A_{sim} (UPDATE). The above steps are repeated until the number of clusters is equal to K.

Prior			PredCls		
Triplet	Subject	Object	mR@50 / 100	R@50 / 100	Mean
			35.7 / 38.2	54.1 / 56.6	**46.2**
✓			39.1 / 42.0	44.9 / 47.8	43.5
	✓		39.5 / 42.3	45.7 / 48.4	44.0
		✓	38.5 / 41.1	46.6 / 49.2	43.9
✓	✓		38.4 / 41.8	40.8 / 44.0	41.3
✓		✓	38.2 / 41.9	40.7 / 44.2	41.3
	✓	✓	**39.9 / 43.0**	42.3 / 45.1	42.6
✓	✓	✓	37.8 / 42.0	37.6 / 41.6	39.8

Table 6: Performance (%) of different prior knowledge combinations on VG [4]. All Experiments are based on Motifs+CFA.

The clustering result is shown in Figure 7. Obviously, some entity categories that share common characters are grouped into the same cluster, *e.g.*, `light` and `lamp`, `mountain` and `hill`, and so on. **Notably, our clustering method focuses on selecting reasonable entity category for query triplet in the same cluster, and does not em-**

SGG Models	PredCls					SGCls					SGGen				
	mR@K		R@K		Mean	mR@K		R@K		Mean	mR@K		R@K		Mean
	50	100	50	100		50	100	50	100		50	100	50	100	
Motifs+TDE [10]$_{CVPR'20}$	24.2	27.9	45.0	50.6	36.9	13.1	14.9	27.1	29.5	21.2	9.2	11.1	17.3	20.8	14.6
Motifs+CogTree [13]$_{IJCAI'21}$	26.4	29.0	35.6	36.8	32.0	14.9	16.1	21.6	22.2	18.7	10.4	11.8	20.0	22.1	16.1
Motifs+RTPB[1]$_{AAAI'22}$	35.3	37.7	40.4	42.5	39.0	20.0	21.0	26.0	26.9	23.5	13.1	15.5	19.0	22.5	17.5
Motifs+PPDL[6]$_{CVPR'22}$	32.2	33.3	47.2	47.6	40.1	17.5	18.2	28.4	29.3	23.4	11.4	13.5	21.2	23.9	17.5
Motifs+GCL[3]$_{CVPR'22}$	36.1	38.2	42.7	44.4	40.4	20.8	21.8	26.1	27.1	24.0	16.8	19.3	18.4	22.0	19.1
Motif+HML[2]$_{ECCV'22}$	36.3	38.7	47.1	49.1	42.8	20.8	22.1	26.1	27.4	24.1	14.6	17.3	17.6	21.1	17.7
Motifs+CFA‡ (ours)	39.9	43.0	42.3	45.1	42.6	20.9	22.4	25.7	27.4	24.1	15.3	18.1	20.7	24.4	19.6
VCTree+TDE [10]$_{CVPR'20}$	26.2	29.6	44.8	49.2	37.5	15.2	17.5	28.8	32.0	23.4	9.5	11.4	17.3	20.9	14.8
VCTree+CogTree [13]$_{IJCAI'21}$	27.6	29.7	44.0	45.4	36.7	18.8	19.9	30.9	31.7	25.3	10.4	12.1	18.2	20.4	15.3
VCTree+RTPB [1]$_{AAAI'22}$	33.4	35.6	41.2	43.4	38.4	24.5	25.8	28.7	30.0	27.3	12.8	15.1	18.1	21.3	16.8
VCTree+PPDL [6]$_{CVPR'22}$	33.3	33.8	47.6	48.0	40.7	21.8	22.4	32.1	33.0	27.3	11.3	13.3	20.1	22.9	16.9
VCTree+GCL [3]$_{CVPR'22}$	37.1	39.1	40.7	42.7	39.9	22.5	23.5	27.7	28.7	25.6	15.2	17.5	17.4	20.7	17.7
VCTree+HML [2]$_{ECCV'22}$	36.9	39.2	47.0	48.8	43.0	25.0	26.8	27.0	28.4	26.8	13.7	16.3	17.6	21.0	17.2
VCTree+CFA‡ (ours)	39.2	42.5	41.9	45.0	42.2	26.3	28.3	32.3	33.8	30.2	15.1	17.9	20.5	24.2	19.4
Transformer+CogTree [13]$_{IJCAI'21}$	28.4	31.0	38.4	39.7	34.4	15.7	16.7	22.9	23.4	19.7	11.1	12.7	19.5	21.7	16.3
Transformer+HML [2]$_{ECCV'22}$	33.3	35.9	45.6	47.8	40.7	19.1	20.4	22.5	23.8	21.5	15.0	17.7	15.4	18.6	16.7
Transformer+CFA‡ (ours)	38.6	41.5	46.2	48.9	43.8	20.9	22.7	28.1	29.6	25.3	15.0	17.9	21.0	24.7	19.7

Table 7: Performance (%) of state-of-the-art tail-focused SGG models on VG [4]. "Mean" is the average of mR@50/100 and R@50/100. ‡ means using the component prior knowledge.

phasize that entity categories located in other clusters must be unreasonable for the query triplet (*e.g.*, `man` is in the same cluster as `woman` of the query triplet `woman-walking in-street`, so `woman` may be replaced to `man`. `men` in another cluster may also be reasonable for `woman`, but we cannot choose to replace the `woman` with it). In addition, since our method is based on statistic of dataset, clustering results may vary from dataset to dataset. Even if the cluster results are not reasonable for all relations, the generated "noisy" clusters are good enough to meet the requirements (*i.e.*, our methods have consistent performance gains on both VG and GQA datasets). We design this clustering for simplicity, and we will leave more comprehensive versions for future works.

C. Component Prior Knowledge

As discussed in prior SGG works [7], the statistic prior of the predicate distribution under a given condition can improve the mR@K performance of the unbiased SGG. In the inference phase, we calculate statistic component prior $b_{s,o,r}$, and add it to the predicted logits to predict predicate category:

$$b_{s,o,r} = -\log \frac{count_{s,o,r}}{\sum_{i=1}^{H} count_{s,o,i}}, \tag{1}$$

where H is the number of predicate categories. As for the prior "Triplet", "Subject", and "Object", $count_{s,o,r}$ is the amount of the triplets whose predicate category is r in the training set given subject-object pair, subject, and object respectively. The result is shown in Table 6, they are all based on the Motifs+CFA under the PredCls setting. The superposition of predicate statistic prior "Subject", and "Object" achieves the best performance on mR@K. We speculate that statistic component prior can improve the performance of predicates with limited component diversity [7] (*i.e.*, tail predicates) at the statistic level, and CFA at the feature level, they complement each other.

Strategy	PredCls		
	mR@50 / 100	R@50 / 100	Mean
Motifs [14]	16.5 / 17.8	65.6 / 67.2	41.8
+Reweight [10]	30.8 $_{\uparrow 14.3}$ / 34.5 $_{\uparrow 16.7}$	36.1 / 40.4	35.5
+Resample [10]	18.5 $_{\uparrow 2.0}$ / 20.0 $_{\uparrow 2.2}$	64.6 / 66.7	42.5
+CFA	**35.7** $_{\uparrow 18.6}$ / **38.2** $_{\uparrow 20.4}$	54.1 / 56.6	**46.2**

Table 8: Performance (%) of re-balancing strategies and CFA on VG [4].

D. Extra Comparisons and Analyses

D.1. Comparison with SOTA Tail-focused Methods

Due to the common label noises in dataset (*e.g.*, head predicate `on` and tail predicate `laying on` are all reasonable for `man-bed`, but the only groundtruth label in the test set is `on`) [5], the improvement of mR@K will inevitably lose the performance of R@K. Therefore, we compared with these tail-focused approaches aiming at improving mR@K separately in this section.

To compare more fully with the SOTA tail-focused approach, we listed all of the metrics (*i.e.*, mR@K, R@K and Mean) in Table 7. As can be seen from the results: 1) After adding component prior knowledge, our CFA‡ has been greatly improved at mR@K metric, *i.e.*, further improve

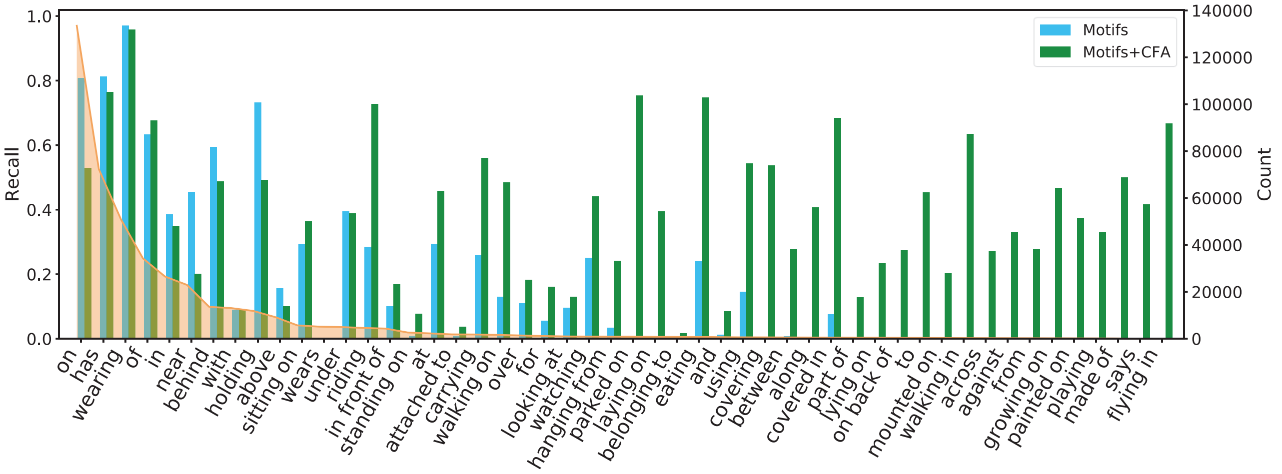

Figure 8: Performance(%) comparison between Motifs [14] and Motifs+CFA over all predicates on test set of VG [4]. The orange area denotes the predicate distribution of training set.

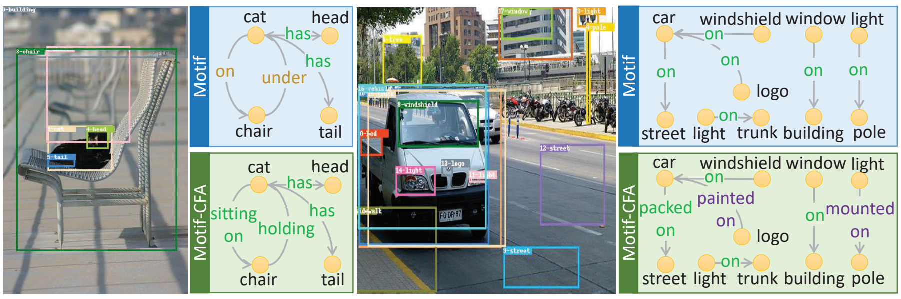

Figure 9: The results of scene graphs generated by Motifs (blue) and Motifs-CFA (green) on VG [4]. Green predicates are correct (*i.e.*, match GT), brown predicates are acceptable (*i.e.*, does not match GT but still reasonable), and purple predicates are more informative (*i.e.*, does not match GT and more reasonable).

the performance of the tail. This is consistent with our intent to use component prior knowledge to improve the performance of predicates with less component diversity (*i.e.*, tail predicates). 2) All the tail-focused methods sacrifice a lot on R@K. Our CFA‡ can maintain a high R@K, when mR@K is significantly increased, *i.e.*, a higher Mean. It proves that our method has a small performance loss for the head predicates while improving the tail performance as much as possible.

D.2. Comparison with Re-balancing Methods.

To demonstrate the superiority of CFA compared with the prevalent re-balancing methods (*i.e.*, reweight and resample) [10], we conducted the three strategies on the baseline Motifs [14]. The results under the PredCls setting are reported in Table 8. From the results, we can observe: 1) The Motifs baseline can achieve the best R@K. However, the high R@K is mainly due to the frequency bias of the dataset [10], and they suffer severe drops in tail predicates. 2) The reweight method achieves better performance at mR@K, but it also sacrifices the performance of head predicates excessively, resulting in a low Mean. 3) The resample method keeps R@K high, but the improvement of mR@K is slight. The reason is that those decision boundaries may be still biased toward the head. 4) CFA achieves the highest mR@K and maintains high R@K, *i.e.*, it achieves the best trade-off over different predicate categories (highest performance on Mean).

D.3. Comparison over All Predicates

To further demonstrate the performance of each predicate, we displayed the predicate distribution of the training set in the VG dataset [4] and the performance of Motifs [14] and Motifs+CFA on R@100 for each predicate category in Figure 8. Obviously, our approach slightly compromises the performance of the head predicates (*e.g.*, `has`), but greatly improves the tail predicates (*e.g.*,`laying on`). This proves the superiority of our method in considering the

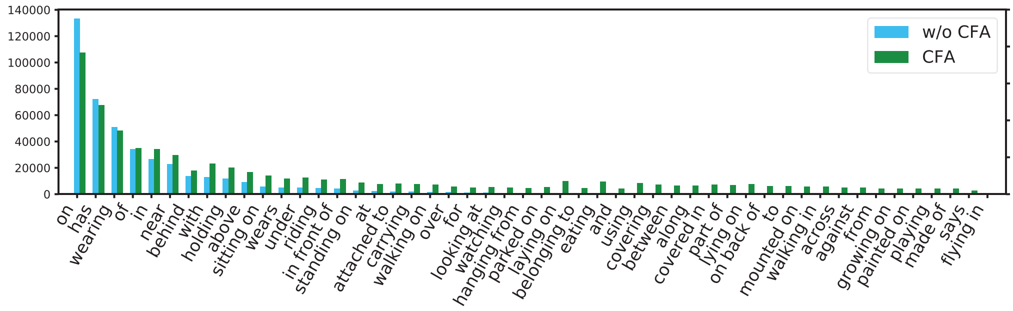

Figure 10: Predicate distribution during training on VG [4].

performance of all predicates.

D.4. Qualitative Analysis.

Figure 9 shows some qualitative results generated by Motifs [14] and Motifs+CFA under PredCls setting. We can observe that CFA can not only predict more accurate predicates (*e.g.*, `under` *vs.* `holding`), but also more fine-grained and informative predicates (*e.g.*, `on` *vs.* `painted on`, and `on` *vs.* `mounted on`).

D.5. Quantitive Results.

To further investigate how CFA works, we visualized the change in the number of training samples for each predicate after applying CFA. As shown in Figure 10, CFA generated considerable training samples for tail predicates, which can effectively increase the diversity of features.

E. Limitations

Although our CFA can enrich the feature diversity, we cannot guarantee that the triplets before and after intrinsic-CFA are absolutely reasonable. In addition, the category of the triplets enhanced by our model is limited by the only triplet categories in the training set, and the triplet enhancement for open-set needs to be explored.

F. Potential Negative Societal Impacts

The enhanced triplets may change the intention in the original triplet, such as `person -laying on-snow` instead of `person-laying on-beach`. In addition, if the feature augmentation method is abused, it may cause data redundancy and waste computing resources.

References

[1] Chao Chen, Yibing Zhan, Baosheng Yu, Liu Liu, Yong Luo, and Bo Du. Resistance training using prior bias: toward unbiased scene graph generation. In *AAAI*, pages 212–220, 2022. 3

[2] Youming Deng, Yansheng Li, Yongjun Zhang, Xiang Xiang, Jian Wang, Jingdong Chen, and Jiayi Ma. Hierarchical memory learning for fine-grained scene graph generation. In *ECCV*, 2022. 3

[3] Xingning Dong, Tian Gan, Xuemeng Song, Jianlong Wu, Yuan Cheng, and Liqiang Nie. Stacked hybrid-attention and group collaborative learning for unbiased scene graph generation. In *CVPR*, pages 19427–19436, 2022. 1, 3

[4] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017. 2, 3, 4, 5

[5] Lin Li, Long Chen, Yifeng Huang, Zhimeng Zhang, Songyang Zhang, and Jun Xiao. The devil is in the labels: Noisy label correction for robust scene graph generation. In *CVPR*, pages 18869–18878, 2022. 3

[6] Wei Li, Haiwei Zhang, Qijie Bai, Guoqing Zhao, Ning Jiang, and Xiaojie Yuan. Ppdl: Predicate probability distribution based loss for unbiased scene graph generation. In *CVPR*, pages 19447–19456, 2022. 3

[7] Xingchen Li, Long Chen, Jian Shao, Shaoning Xiao, Songyang Zhang, and Jun Xiao. Rethinking the evaluation of unbiased scene graph generation. In *BMVC*, 2022. 3

[8] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 1

[9] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, pages 2537–2546, 2019. 1

[10] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training. In *CVPR*, pages 3716–3725, 2020. 1, 3, 4

[11] Meng Wei, Chun Yuan, Xiaoyu Yue, and Kuo Zhong. Hose-net: Higher order structure embedded network for scene graph generation. In *ACM MM*, pages 1846–1854, 2020. 2

[12] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *CVPR*, pages 5410–5419, 2017. 1

[13] Jing Yu, Yuan Chai, Yue Hu, and Qi Wu. Cogtree: Cognition tree loss for unbiased scene graph generation. In *IJCAI*, 2021. 3

[14] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *CVPR*, pages 5831–5840, 2018. 3, 4, 5