EGTR: Extracting Graph from Transformer for Scene Graph Generation

Jinbae Im[1]   JeongYeon Nam[1]   Nokyung Park[1,2,3]*   Hyungmin Lee[2]   Seunghyun Park[1]

[1]NAVER Cloud AI   [2]NAVER   [3]Korea University

{jinbae.im, jy.nam, nokyung.park99, hyungmin.lee, seung.park}@navercorp.com

Abstract

Scene Graph Generation (SGG) is a challenging task of detecting objects and predicting relationships between objects. After DETR was developed, one-stage SGG models based on a one-stage object detector have been actively studied. However, complex modeling is used to predict the relationship between objects, and the inherent relationship between object queries learned in the multi-head self-attention of the object detector has been neglected. We propose a lightweight one-stage SGG model that extracts the relation graph from the various relationships learned in the multi-head self-attention layers of the DETR decoder. By fully utilizing the self-attention by-products, the relation graph can be extracted effectively with a shallow relation extraction head. Considering the dependency of the relation extraction task on the object detection task, we propose a novel relation smoothing technique that adjusts the relation label adaptively according to the quality of the detected objects. By the relation smoothing, the model is trained according to the continuous curriculum that focuses on object detection task at the beginning of training and performs multi-task learning as the object detection performance gradually improves. Furthermore, we propose a connectivity prediction task that predicts whether a relation exists between object pairs as an auxiliary task of the relation extraction. We demonstrate the effectiveness and efficiency of our method for the Visual Genome and Open Image V6 datasets. Our code is publicly available at https://github.com/naver-ai/egtr.

1. Introduction

Scene Graph Generation (SGG) [8] aims to generate a scene graph that represents objects as nodes and relationships between objects as edges from an image as shown in Fig. 1b. SGG is a challenging task as it is required to go beyond simply detecting objects but predicting the relationships between them based on a comprehensive understanding of the

_____
* Most work was done during the internship at NAVER Cloud AI.

(a) Visual Genome example

(b) Scene Graph
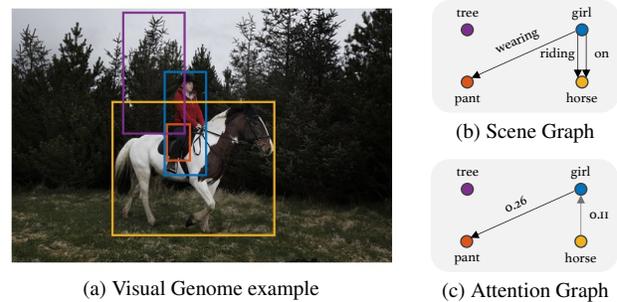
(c) Attention Graph

Figure 1. **Motivation.** SGG task aims to predict scene graph (Fig. 1b) with objects as nodes and relations as edges. We draw a plausible attention graph (Fig. 1c) simply by connecting objects with high attention weights to edges from the self-attention layers of the pre-trained DETR. It shows the potential for the self-attention from the object detector to contain rich information that aids in predicting the relations of the scene graph.

scene. Since a scene graph provides structural information of the image, it can be used for various vision tasks that require a higher level of understanding and reasoning about images such as image captioning [5, 10, 41], image retrieval [8, 28], and visual question answering [14, 43].

Most previous studies [16, 19, 33, 39, 42] took two-stage SGG approaches that detect objects first and then predict their relations. However, these approaches incurred high computational costs and risks of error propagation from the object detection. To address the drawbacks of the two-stage approaches, one-stage SGG models that perform object detection and relation prediction at once [3, 9, 17, 20, 26, 29, 35] have been studied recently, leveraging one-stage object detectors such as DETR [1]. Since edges in a scene graph can be represented as subject-predicate-object triplets, many studies embraced triplet-based approaches as shown in Figs. 2a and 2b. However, object-triplet detection models [3, 17] required sophisticated triplet detectors to obtain necessary information for triplet queries from the object detector. Moreover, triplet detection models [9, 35] lacked the ability to detect objects without relation such as "tree" in Fig. 1b, by focusing solely on the triplet detection without an object detector. Considering that objects without relations account for more than
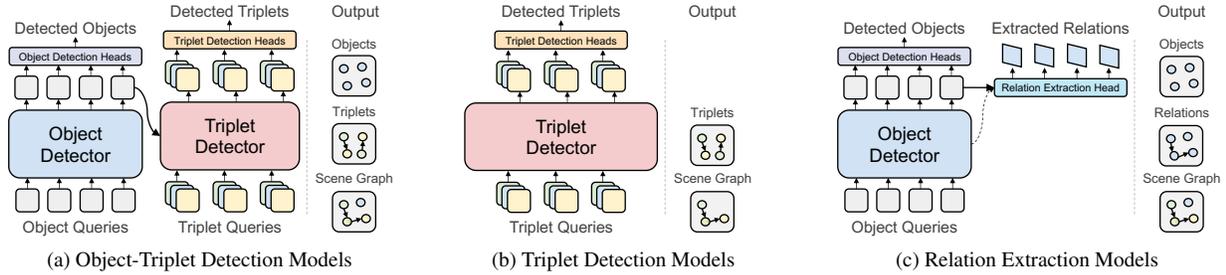
Figure 2. **Comparison with existing one-stage SGG models.** (a) Object-Triplet Detection Models introduce additional triplet queries and a triplet detector to the object detector. The triplet detector requires additional modules to incorporate information from the object detector into the triplet queries. (b) Triplet Detection Models focus on detecting triplets directly without an object detector. Objects without relations may not be detected. (c) Relation Extraction Models extract relations from the object detector without a separate triplet detector. In particular, ours extracts relations more effectively by utilizing by-products from the self-attention of the object detector.

42% in Visual Genome [12] data, their priority lies in detecting a sub-graph rather than the complete scene graph.

To address the shortcomings of the existing one-stage SGG models, we focus on the relationships between objects inherent in the object detector. As shown in Fig. 1a, objects are related to each other. For instance, when a horse appears in a scene, a person is likely to appear in the scene, and clothing such as hats, jackets, and pants often depicts the person's current situation. From this intuition, there has been a long belief [2, 7, 22] that modeling relationship or context between objects would be beneficial to object detection task. Accordingly, recent one-stage object detectors [1, 32, 46] have incorporated self-attention layers [36] to implicitly model the relationships among the object queries. We hypothesize that self-attention between object queries learned in a one-stage object detector might contain valuable information for predicting triplet outputs. In our preliminary investigation, we are able to extract a plausible attention graph by simply connecting two object queries with high attention weights from the pre-trained DETR, as shown in Fig. 1c. It shows the potential that the attention weights between object queries can be interpreted as relations between them.

From the findings, we propose a lightweight one-stage scene graph generator **EGTR**, which stands for **E**xtracting **G**raph from **TR**ansformer. We design the model to comprehensively leverage the by-products of the object detector, eliminating the need for a separate triplet detector, as depicted in Fig. 2c. From the multi-head self-attention layers of the object detector, we regard an attention query and key, where their relations are learned in the attention weights, as the subject entity and object entity, respectively. Subsequently, we leverage a shallow classifier to predict relations between them. Due to the abundant information about the relationships among the objects present in the by-products derived from all layers of self-attention, we can effectively extract the scene graph.

Since the relation extraction task is dependent on the object detection task, we speculate that performing relation extraction without sufficiently learned representations of the object queries might be harmful. Therefore, we devise a novel adaptive smoothing technique that smooths the value of the ground truth relation label based on the object detection performance. With the adaptive smoothing, the model is trained with a continuous curriculum that initially focuses on object detection and gradually performs multi-task learning. Furthermore, we propose a connectivity prediction task as an auxiliary task for relation extraction, aiming to predict the existence of any relationship between a subject entity and an object entity. This auxiliary task facilitates the acquisition of representations for the relation extraction.

To verify the effectiveness of the proposed method, we conduct experiments on two representative SGG datasets: Visual Genome [12] and Open Images V6 [13]. By actively utilizing the by-products of the object detector, EGTR shows the best object detection performance and comparable triplet detection performance, with the fewest parameters and the fastest inference speed.

Our main contributions can be summarized as follows:

- We propose EGTR that generates scene graphs efficiently and effectively by utilizing the multi-head self-attention by-products derived from the object detector.
- We present adaptive smoothing, enabling effective multi-task learning for both object detection and relation extraction. In addition, the proposed connectivity prediction offers clues to the relation extraction.
- Our comprehensive experiments show the superiority of the proposed model framework and the effectiveness of the devised training techniques.

2. Related Work

SGG models can be categorized into two groups: two-stage models and one-stage models. For two-stage models [4, 11, 15, 16, 19, 23–25, 30, 33, 37, 39, 42, 45], separate object detection model and relation prediction model are trained sequentially. They usually detect N objects from the off-the-shelf object detector such as Faster R-CNN [27],

and then all possible combinations of the detected objects are fed into the relation prediction model to predict the relations between each object pair. Although they showed high relation extraction performance, they have the inherent limitation that the object detector that helps with relation extraction is trained separately, resulting in a significant increase in model complexity.

As for the one-stage models [3, 9, 17, 20, 26, 29, 35], the object detection and relation prediction are trained in an end-to-end manner. Early studies [20, 26] proposed fully convolutional SGG models and took pixel-based approaches. After DETR [1] brought a huge success as a Transformer [36]-based one-stage object detector, many one-stage SGG studies are based on one-stage object detectors [1, 32, 46]. They efficiently modeled SGG by introducing object queries or triplet queries. We categorize them into three distinct groups: (a) object-triplet detection models, (b) triplet prediction models, and (c) relation extraction models, as shown in Fig. 2.

Object-Triplet Detection Models. Object-triplet detection models are characterized by introducing additional triplet queries and building a triplet predictor on top of the object detector as shown in Fig. 2a. RelTR [3] introduced paired subject queries and object queries, and SGTR [17] introduced compositional queries decoupled into subjects, objects, and predicates. As the introduced queries are initialized without prior cues from the object detector, the triplet predictor requires modules to incorporate information from the outputs of the object detector and modules for triplet queries to exchange information with each other. It led to the triplet predictor having an intricate structure. In contrast, we refrain from introducing additional queries and regard the attention queries and attention keys, whose relationships are learned in the object detector, as subject and object queries, respectively.

Triplet Detection Models. Triplet detection models directly detect triplets using triplet queries without an object detector, as shown in Fig. 2b. Iterative SGG [9] introduced subject, object, and predicate queries and modeled the conditional dependencies among them with separate subject, object, and predicate multi-layer Transformer decoders. Inspired by Sparse R-CNN [32], Structured Sparse R-CNN (SSR-CNN) [35] designed triplet queries consisting of subject box, object box, subject, object, and predicate queries. Since it was difficult to train the model only with sparse triplet annotations, SSR-CNN introduced somewhat intricate training details: it detected object pairs with a Siamese Sparse RCNN model and auxiliary queries and performed additional triplet matching using the detected object pairs as a pseudo label. Although they show excellent triplet detection performance, the model architecture became even more intricate to detect subjects and objects separately since

an explicit object detector is not used. Last but not least, the triplet prediction models focus on detecting a sub-graph composed only of objects with relations, overlooking objects in an image that lack explicit relations.

Relation Extraction Models. Relation extraction models extract a scene graph using a lightweight relation predictor without separate triplet queries or triplet detector. Relationformer [29] added a special "[rln]" token to capture global information in conjunction with object queries. They concatenated the final hidden representations of the object query pairs and the relation token, followed by a shallow fully connected network for relation prediction. In this work, in conjunction with the final hidden representations, we use the inherent relationship information among the object queries learned in the multi-head self-attention layer of the object detector. Furthermore, we propose training techniques to boost multi-task learning, leading to significantly improved performance with architectural simplicity.

3. Method

3.1. Preliminaries

In this section, we first introduce the formulation of the SGG task and then provide a brief review of the one-stage object detector that serves as the basis for our study.

3.1.1 Formulation

Scene graph generation is a task of generating a scene graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ from an image, where $\mathcal{V}$ denotes the node set consisting of objects and $\mathcal{E}$ denotes the edge set that represents the relation between objects. Each object entity $v_i \in \mathcal{V}$ has an object category label v_i^c from a set of object categories $\mathcal{C}_v$ and box coordinates v_i^b. Each relation $e_j \in \mathcal{E}$ represents the j-th triplet (s_j, p_j, o_j), where subject s_j and object o_j indicate related object entities and predicate p_j has a relation category label p_j^c from a set of predicate categories $\mathcal{C}_p$. Generating $\mathcal{V}$ and $\mathcal{E}$ correspond to object detection and relation extraction, respectively.

3.1.2 One-stage Object Detector

Our model framework is deeply influenced by DETR [1], a one-stage object detector. In DETR, representations for input images are learned through CNN and Transformer [36] encoder. Transformer decoder enhances object queries by utilizing self- and cross-attention mechanisms. The final object detection heads predict object category labels and box coordinates from the contextualized object queries.

Backbone. CNN backbone generates C-dimensional feature map $F \in \mathbb{R}^{C \times H_F \times W_F}$ from an input image $x_{\text{img}} \in \mathbb{R}^{3 \times H_0 \times W_0}$. For ResNet-50 backbone, H_F and W_F are set to $H_0/32$ and $W_0/32$, respectively.
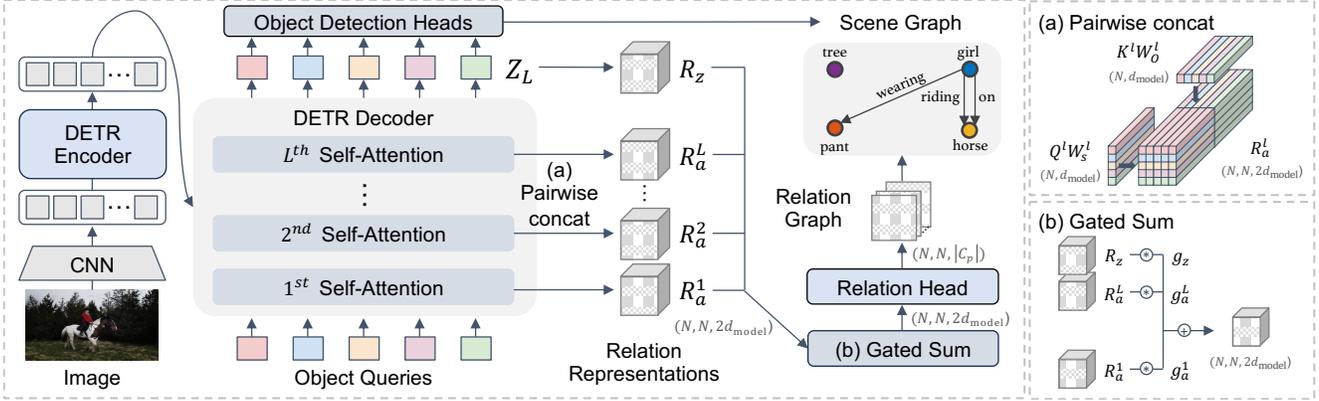
Figure 3. **The overall architecture of EGTR**. We present a novel lightweight relation extractor, EGTR, which fully utilizes the self-attention of the DETR decoder. We extract query and key representations from each self-attention layer and concatenate them pairwise to represent relations between them. Additionally, we leverage the last hidden representation in same manner. To effectively aggregate information, we apply a gated sum and then predict relation with a shallow relation head.

Transformer Encoder. Transformer encoder enhances image representations based on multi-head self-attention. After applying a 1×1 convolution to reduce dimension from C to d_{model}, the feature map is flattened to get an input sequence with length $H_F W_F$ for the Transformer encoder. Additional positional encodings are used to reflect the spatial information of the feature maps.

Transformer Decoder. Transformer decoder takes N object queries as input and returns their representations. Each object query detects an individual object, and N is usually set large enough to cover all objects in the image. Through alternating self-attention and cross-attention layers, object queries learn features of object candidates in the input image. Note that causal attention mask is not applied to the self-attention layer; therefore, fully connected $N \times N$ attention weights among the N object queries are learned from the multi-head self-attention as follows:

$$A_h^l = f(Q_h^l, K_h^l) = \text{softmax}(Q_h^l K_h^{l\,T} / \sqrt{d_{\text{head}}}), \quad (1)$$

where $A_h^l \in \mathbb{R}^{N \times N}$ denotes the attention weights from the h-th head in the l-th layer. $Q_h^l \in \mathbb{R}^{N \times d_{\text{head}}}$ and $K_h^l \in \mathbb{R}^{N \times d_{\text{head}}}$ are attention queries and keys, respectively.

Object Detection Heads. Object detection heads detect N object candidates $\{\hat{v}_i\}_{i=1}^N$ from the last layer representations of the object queries Z^L. For each object query, a linear layer is used to predict the object category label $\hat{v}_i^c \in \mathbb{R}^{N \times |\mathcal{C}_v|}$, and three-layer perceptron with ReLU activation is used to obtain the box coordinates $\hat{v}_i^b \in \mathbb{R}^{N \times 4}$.

Object Detection Loss. To match N detected object candidates $\{\hat{v}_i\}_{i=1}^N$ and M ground truth objects $\{v_i\}_{i=1}^M$, Carion *et al.* [1] pad the ground truth objects with ϕ (no object) and find the best permutation of predicted objects that minimizes total bipartite matching costs. From the permutated

predictions $\{\hat{v}_i'\}_{i=1}^N$, the loss is calculated as follows:

$$\mathcal{L}_{\text{od}} = \Sigma_{i=1}^N [\lambda_c \mathcal{L}_c(\hat{v}_i'^c, v_i^c) + \mathbb{1}_{v_i^c \neq \phi}(\lambda_b \mathcal{L}_b(\hat{v}_i'^b, v_i^b))], \quad (2)$$

where $\mathcal{L}_c$ and $\mathcal{L}_b$ denote loss function for category labels and box coordinates, respectively.

3.2. EGTR

We propose a novel lightweight relation extractor, EGTR, which exploits the self-attention of DETR decoder, as depicted in Fig. 3. Since the self-attention weights in Eq. (1) contain $N \times N$ bidirectional relationships among the N object queries, our relation extractor aims to extract the predicate information from the self-attention weights in the entire L layers, by considering the attention queries and keys as subjects and objects, respectively.

In order to preserve rich information learned in the self-attention layer, we devise another relation function f between the attention queries and keys instead of the dot product attention in Eq. (1). As concatenation preserves representations completely, we concatenate the representations on all $N \times N$ pairs of attention queries and keys as shown in Fig. 3 (a) to get the relation representations of the l-th layer $R_a^l \in \mathbb{R}^{N \times N \times 2d_{\text{model}}}$. Before pairwise concatenation, to help queries and keys play the role of subjects and objects, a linear projection is added as follows:

$$R_a^l = [Q^l W_S^l; K^l W_O^l], \quad (3)$$

where $Q^l \in \mathbb{R}^{N \times d_{\text{model}}}$ and $K^l \in \mathbb{R}^{N \times d_{\text{model}}}$ refer to attention queries and keys of the l-th layer, and $[\,\cdot\,;\,\cdot\,]$ denotes the pairwise concatenation. W_S^l and W_O^l are linear weights of shape $\mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$.

We also utilize the last layer representations of the object queries $Z^L \in \mathbb{R}^{N \times d_{\text{model}}}$, which are used for the object

detection in the same manner:

$$R_z = [Z^L W_S; Z^L W_O], \qquad (4)$$

where W_S, W_O are linear weights of shape $\mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$. To effectively use the various information learned in all layers, we introduce a gating mechanism as follows:

$$g_a^l = \sigma(R_a^l W_G), g_z = \sigma(R_z W_G), \qquad (5)$$

where g_a^l and $g_z \in \mathbb{R}^{N \times N \times 1}$ represent the gate values obtained through a single linear layer $W_G \in \mathbb{R}^{2d_{\text{model}} \times 1}$ for R_a^l and R_z, respectively. Finally, we extract the relation graph from the gated summation of the relation representations across all layers as follows:

$$\hat{G} = \sigma(\text{MLP}_{\text{rel}}(\Sigma_{l=1}^{L}(g_a^l * R_a^l) + g_z * R_z)), \qquad (6)$$

where $\hat{G} \in \mathbb{R}^{N \times N \times |\mathcal{C}_p|}$ denotes predicted relation graph and MLP_{rel} is a three-layer perceptron with ReLU activation. Note that we use sigmoid function σ so that multiple relationships can exist between objects.

3.3. Learning and Inference

To train EGTR, we perform multi-task learning. In addition to object detection and relation extraction, we devise connectivity prediction, an auxiliary task for relation extraction. The overall loss for the framework is as follows:

$$\mathcal{L} = \mathcal{L}_{\text{od}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}} + \lambda_{\text{con}}\mathcal{L}_{\text{con}}, \qquad (7)$$

where $\mathcal{L}_{\text{od}}$ is object detection loss in Eq. (2). $\mathcal{L}_{\text{rel}}$ and $\mathcal{L}_{\text{con}}$ are loss functions for the relation extraction and connectivity prediction, respectively. Since explicit object detection loss is used, EGTR has the capability to detect all nodes in the scene graph. The details of the loss for each task are provided below.

3.3.1 Relation Extraction

We use binary cross-entropy loss for the relation extraction. To match predicted graph $\hat{G} \in \mathbb{R}^{N \times N \times |\mathcal{C}_p|}$ and ground truth triplet set $\mathcal{E}$, we first encode $\mathcal{E}$ as a one-hot ground truth graph $G \in \mathbb{R}^{N \times N \times |\mathcal{C}_p|}$ by padding the regions that do not correspond to relations between the ground truth objects as zero. Then, we permutate indices of the predicted graph using the permutation found in the object detection. From the permutated graph $\hat{G}'$, the loss for the relation extraction is calculated as $\mathcal{L}_{\text{rel}} = \mathcal{L}_r(\hat{G}', G)$.

However, since the size of the graph is proportional to the square of N, the sparsity of the ground truth graph G is too severe. For instance, the density is only 10^{-14} when N is set to 200 for the Visual Genome [12] validation dataset. Therefore, we divide G into three regions: (1) GT region,
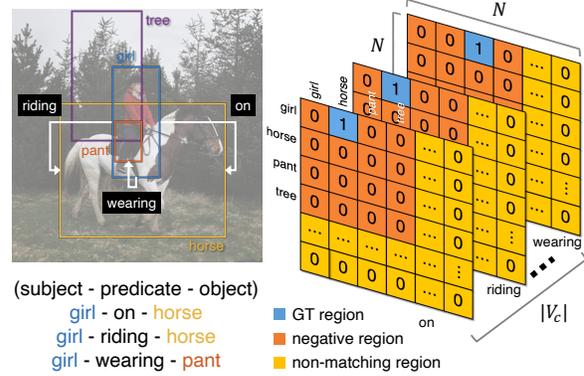
Figure 4. **Example of graph G region.** The target graph has a shape of $N \times N \times |\mathcal{V}_c|$. The graph is severely sparse since the number of object queries N is set large enough to cover objects in the image. We split G into GT, negative, and non-matching regions.

(2) negative region, and (3) non-matching region, as shown in Fig. 4. The GT region indicates the ground truth triplets where the value of G is 1. The negative region consists of the triplets in which both subjects and objects are composed of the ground truth objects, but no relation exists between them. The non-matching region represents the zero-padded region. It is paired with a region consisting of the object candidates that do not match the ground truth objects and match ϕ in Eq. (2). For each region, we apply different techniques to effectively train the relation extraction with object detection as subsequently described.

Adaptive Smoothing. We propose a novel adaptive smoothing for the GT region. For G_{ijk} belonging to the GT region, the model is trained to predict the k-th predicate category between subject entity v_i and object entity v_j. However, since $\hat{v}_i'$ and $\hat{v}_j'$, which match v_i and v_j respectively, do not have enough representations about the corresponding ground truth object at the beginning of the training, it may not be appropriate to predict the probability of the predicate as 1. Moreover, even when the object detection performance is reasonably assured, the detection performance may still vary for individual object candidates. Therefore, we reflect the detection performance of each object candidate on the relation label via adaptive smoothing.

We first measure the uncertainty of each object candidate with the corresponding bipartite matching cost. For object candidate $\hat{v}_i'$, we define the uncertainty as follows:

$$u_i = \sigma(\text{cost}_i - \text{cost}_{\min} + \sigma^{-1}(\alpha)), \qquad (8)$$

where cost_i denotes the matching cost and $\text{cost}_{\min}$ indicates the matching cost when $\hat{v}_i'$ perfectly matches v_i. α is a nonnegative hyperparameter representing the minimum uncertainty. We set the value of G_{ijk} to $(1 - u_i)(1 - u_j)$ taking the uncertainty into account. By using the relation label adjusted by the uncertainty, the multi-task learning of object

detection and relation extraction is dynamically regulated according to the quality of the detected objects.

Negative and Non-matching Sampling. Rather than employing all negatives, we sample them from the negative region. Inspired by the hard negative mining introduced by Liu *et al*. [21], we sort all negatives based on the predicted relation score $\hat{G}'_{ijk}$ and choose the top $k_{\text{neg}} \times |\mathcal{E}|$ most challenging negatives. Similarly, we extract $k_{\text{non}} \times |\mathcal{E}|$ hard samples from the non-matching region. As the non-matching region typically encompasses a substantial part of the graph G, this approach notably diminishes sparsity.

3.3.2 Connectivity Prediction

Influenced by Graph-RCNN [40] that predicted a relatedness to prune object pairs, we propose a connectivity prediction that predicts whether at least one edge between two object nodes exists for the relation extraction. We get a connectivity graph $\hat{E} \in \mathbb{R}^{N \times N \times 1}$ in a similar way to get the relation graph in Eq. (6) with the same relation source representations. Instead of MLP_{rel} for multi-label prediction, we use another MLP_{con} for binary classification. We calculate the binary cross entropy loss from the permuted connectivity graph as follows: $\mathcal{L}_{\text{con}} = \mathcal{L}_r(\hat{E}', E)$.

3.3.3 Inference

For model inference, we get triplet scores by multiplying predicate score $\hat{G}_{ijk}$ by the corresponding class scores of $\hat{v}_i^c$ and $\hat{v}_j^c$. Note that we set $\hat{G}_{iik}$ to 0 to prevent self-connections in which the subject and object are the same entity. In addition, we enhance the triplet scores by utilizing the connectivity score $\hat{E}_{ijk}$. By multiplying the connectivity score, we can effectively filter out the triplets that are not likely to have a relation between the subject and object.

4. Experiments

4.1. Datasets and Evaluation Settings

We conduct experiments on two SGG datasets. We describe the datasets and evaluation settings for each dataset. Detailed settings for each dataset are presented in the supplementary materials.

Visual Genome. Visual Genome [12] is the most representative SGG dataset, consisting of 57K training images, 5K validation images, and 26K test images. We follow the popular Visual Genome split, retaining the most frequent 150 object categories and 50 relation categories. We adopt Scene Graph Detection (SGDet) evaluation settings and report Recall@k (R@k) which is class agnostic and mean Recall@k (mR@k) that aggregates the recalls for each predicate category. Following Motifs [42], these metrics are measures with graph constraint, which means each

object pair can have a single predicate category. Since they are only related to triplet detection of the scene graph, we report AP50 that evaluates the detection performance of all objects appearing in the scene. Furthermore, we report the number of model parameters and Frames Per Second (FPS) to measure efficiency.

Open Image V6. Open Image V6 [13] is also widely used dataset, comprising of 126K images for training, 2K for validation, and 5K for test set. It includes 601 object categories and 30 relation categories. For evaluation, we adopt both recall and weighted mean AP (wmAP) following standard settings. For recall evaluation, micro-R@50 is adopted. The wmAP is evaluated with two settings: wmAP_{rel} for predicting boxes of subject entity and object entity separately and wmAP_{phr} for predicting a union box of them. The final score is calculated by $0.2 \times \text{micro-R@}50 + 0.4 \times \text{wmAP}_{\text{rel}} + 0.4 \times \text{wmAP}_{\text{phr}}$.

4.2. Implementation Details

We employ Deformable DETR [46] that improves the convergence speed of the DETR with ResNet-50 [6] as a backbone. It is worth noting that our approach can be extended to any object detector that incorporates self-attention mechanisms between object features, including models like DETR [1], Sparse R-CNN [32], and others. We follow the configurations of the original Deformable DETR, except that we use only 200 object queries. To speed up convergence, we first train the object detector with the target dataset and subsequently train the SGG task using the pretrained object detector, similar to prior work [17]. To calculate the overall loss in Equation 7, λ_{rel} is set to 15. λ_{con} is set to 30 and 90 for the Visual Genome and Open Image V6, respectively. We set α as 10^{-14} for adaptive smoothing and both k_{neg} and k_{non} are set to 80.

4.3. Results

We present quantitative results and perform a comparative analysis of our proposed framework with representative SGG models. Additional experimental results are included in the supplementary materials.

Visual Genome. Visual Genome results are shown in Tab. 1. Our proposed method demonstrates competitive performance with the current one-stage SGG models that have 1.5 to 6.5 times larger parameters with the fastest inference speed. In particular, our method achieves the highest object detection performance and shows comparable performance in the triplet detection compared to SSR-CNN [35], the state-of-the-art method. The results demonstrate that our method can generate scene graphs in an efficient and effective way by exploiting the by-products of the object detector. By applying logit adjustment [35], a technique for predicting tail predicate classes well, our

	Model	# params (M)	FPS	AP50	R@20	R@50	R@100	mR@20	mR@50	mR@100
two-stage	IMP (EBM) [31, 39]	*322.2*	*2.0*	28.1	18.1	25.9	31.2	2.8	4.2	5.4
	VTransE [44]	*312.3*	*3.5*	-	24.5	31.3	35.5	5.1	6.8	8.0
	Motifs [42]	*369.9*	*1.9*	28.1	25.1	32.1	36.9	4.1	5.5	6.8
	VCTree [33]	*361.5*	*0.8*	28.1	24.8	31.8	36.1	4.9	6.6	7.7
	VCTree (TDE) [33, 34]	*361.3*	*0.8*	28.1	14.0	19.4	23.2	6.9	9.3	11.1
	VCTree (EBM) [31, 33]	*372.5*	-	28.1	24.2	31.4	35.9	5.7	7.7	9.1
	GPS-Net [19]	-	-	-	-	31.1	35.9	-	6.7	8.6
	BGNN [16]	341.9	*1.7*	29.0	23.3	31.0	35.8	7.5	10.7	12.6
one-stage	FCSGG [20]	87.1	*6.0*	<u>28.5</u>	16.1	21.3	25.1	2.7	3.6	4.2
	RelTR [7]	<u>63.7</u>	<u>13.4</u>	26.4	21.2	27.5	-	6.8	10.8	-
	SGTR [17]	*117.1*	*6.2*	25.4	-	24.6	28.4	-	12.0	15.2
	Relationformer [29]	92.9	8.5	26.3	22.2	28.4	31.3	4.6	9.3	10.7
	Iterative SGG [9]	93.5	*6.0*	27.7†	-	29.7	32.1	-	8.0	8.8
	SSR-CNN [35]	*274.3*	*4.0*	23.8†	**25.8**	**32.7**	**36.9**	6.1	8.4	10.0
	SSR-CNN [35] $_{LA, \tau=0.3}$	*274.3*	*4.0*	23.8†	18.4	23.3	26.5	**13.5**	**17.9**	<u>21.4</u>
	EGTR (Ours)	**42.5**	**14.7**	**30.8**	<u>23.5</u>	<u>30.2</u>	<u>34.3</u>	5.5	7.9	10.1
	EGTR (Ours) $_{LA, \tau=0.7}$	**42.5**	**14.7**	**30.8**	15.7	18.7	20.5	<u>12.1</u>	<u>17.8</u>	**21.7**
	EGTR (Ours) $_{LA, \tau=0.5}$	**42.5**	**14.7**	**30.8**	19.7	24.2	26.7	11.0	17.1	<u>21.4</u>
	EGTR (Ours) $_{LA, \tau=0.3}$	**42.5**	**14.7**	**30.8**	22.4	28.2	31.7	8.8	14.0	18.3

Table 1. **Graph-Constraint results on the test set of Visual Genome dataset.** We report the results of representative two-stage SGG models based on the Faster R-CNN [27] object detector with a ResNeXt-101-FPN [18, 38] backbone and concurrent one-stage SGG models. Among the one-stage models, the best is highlighted in bold, and the second-best is indicated with underlining. LA denotes logit adjustment proposed in SSR-CNN [35]. † represents that we concatenate the predicted subjects and objects and then apply non-maximum-suppression with a threshold of 0.3. *Italic* indicates the evaluation of metrics using publicly available model checkpoints. FPS is measured with a single V100 for images resized to a minimum of 600 for the shortest side and a maximum of 1000 for the longest side.

Model	score	micro-R@50	wmAP$_{rel}$	wmAP$_{phr}$
Motifs [42]	38.9	71.6	29.9	31.6
VCTree [33]	40.2	74.1	34.2	33.1
GPS-Net [19]	41.7	74.8	32.9	34.0
BGNN [16]	42.1	75.0	33.5	34.2
RelTR [7]	43.0	71.7	34.2	37.5
SGTR [17]	42.3	59.9	37.0	38.7
SSR-CNN [35]	**49.4**	**76.7**	<u>41.5</u>	**43.6**
EGTR (Ours)	<u>48.6</u>	<u>75.0</u>	**42.0**	<u>41.9</u>

Table 2. **Results on test set of Open Image V6.** The score is a weighted sum of micro-R@50, wmAP$_{rel}$, and wmAP$_{phr}$.

method performs favorably in the trade-off between R@k and mR@k, showing superiority over existing state-of-the-art models. Note that for triplet detection models [9, 35] without an explicit object detector, AP50 is measured by applying non-maximum-suppression (NMS) to the union of predicted subjects and objects set. AP50 performance is discussed further in the section 4.5.

Open Image V6. As depicted in Tab. 2, the experiments conducted on the Open Image V6 dataset also demonstrate competitive performance, underscoring the effectiveness and robustness of our method across different datasets.

4.4. Ablation Studies

We analyze the effects of our model components. All of the ablation studies are done with the Visual Genome dataset.

Relation Sources. In Tab. 3, we investigate the source of relations used for the relation extractor. To verify the benefits of using attention queries and keys, we change Q^l and K^l in Eq. (3) to Z^l, the hidden states of each layer. Remarkably, when simply using the hidden states of all layers, the triplet detection performance decreases. Surprisingly, the performance is lower than that of using only R_z from the last layer. On the other hand, when only attention by-products are used as a relation source, the performance is similar to that of using only hidden states of the last layer, the most contextualized representations for detecting objects. Incorporating both attention by-products and final hidden states yields improved performance by amalgamating varied information. The results underscore that the attention by-products of the object detectors contain rich information for relation extraction.

Training Techniques. In Tab. 4, we ablate proposed techniques. Without any techniques, our model framework shows low performance. When each technique is applied, it substantially improves both R@50 and mR@50, and the best performance is achieved when all techniques are combined. The results demonstrate that our proposed techniques help to train the model framework effectively. Additional experimental results are provided in the supplementary materials.

Sampling Methodology. In Tab. 5, we investigate the impact of hard sampling methods for the negative and non-matching regions. Hard negatives and hard non-matchings show a trade-off that improves the performance of R@50

R_a^l source	R_a^l	R_z	R@50	mR@50
Q^l & K^l	✓	✓	**30.2**	**7.9**
Z^l	✓	✓	29.6	7.4
-		✓	29.9	7.6
Q^l & K^l	✓		29.8	7.7

Table 3. **Ablation study on relation source.**

adaptive smoothing	$\mathcal{L}_{con}$	sampling	R@50	mR@50
			26.6	5.3
✓			28.3	6.5
	✓		29.6	7.0
		✓	28.9	7.1
✓	✓	✓	**30.2**	**7.9**

Table 4. **Ablation study on proposed techniques.**

hard negative	hard non-matching	R@50	mR@50
✓	✓	**30.2**	**7.9**
✓		30.0	7.1
	✓	29.6	7.7

Table 5. **Ablation study on sampling methods.**

and mR@50, respectively, and additional performance improvements are achieved for both measures when they are used together. For hard negatives, we speculate that R@50 performance is improved by selecting negatives for the tail predicate class, which is likely to exist in reality but is not in the annotation. For hard non-matchings, mR@50 performance seems to have improved by choosing object candidates that are akin to the ground truth objects but unmatched with them. This encourages the model to predict diverse predicate classes by preventing duplicate object pairs, which are likely to predict head predicate classes, from being predicted.

4.5. Discussion

Regarding the highest AP50 score, we posit that it is because our proposed method focuses on not only objects with relations but also objects without any relations. To validate this, we calculate AP50 for two subsets of the ground-truth objects: those with relations and those without relations. We compare our method with two triplet detection-based models. Tab. 6 illustrates that our method achieves significantly high $AP50_{no-rel}$. Considering that triplet detection performance is related to $AP50_{rel}$, while triplet detection-based models only focus on objects with relations to detect triplets well, our method is capable of extracting the complete scene graph, including objects without relations.

For further analysis, we visualize bounding boxes for subjects and objects from the top 20 predictions in Fig. 5. For SSR-CNN [35], which directly predicts triplets, we observe a significant number of overlapping bounding boxes. On the other hand, our proposed model leverages predic-

Model	AP50	$AP50_{rel}$	$AP50_{no-rel}$
Iterative SGG [9]†	27.7	**24.3**	7.8
SSR-CNN [35]†	23.8	20.2	7.4
EGTR (Ours)	**30.8**	**24.3**	**10.7**

Table 6. **AP50 for two subsets of objects.** $AP50_{rel}$ and $AP50_{no-rel}$ denote scores evaluated only on objects having at least one edge and no edge, respectively. Note that each measure operates on its unique scale since the ground-truth objects used for each measure differ. † indicates that additional NMS is applied to the union of subjects and objects set.
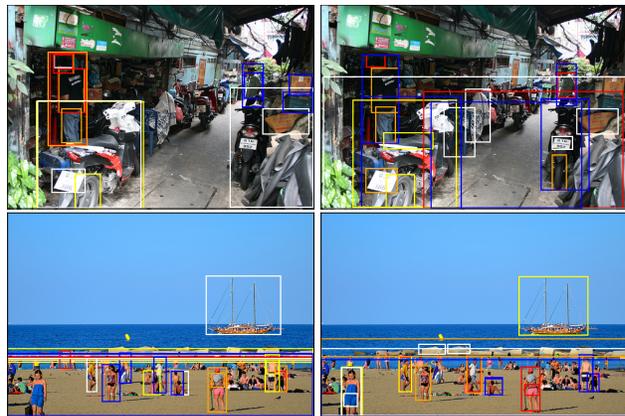
(a) SSR-CNN [35]   (b) EGTR (Ours)

Figure 5. **Comparison of detected subjects and objects.** We select the top 20 predictions of SSR-CNN and EGTR and visualize bounding boxes of detected subjects and objects.

tions from an explicit object detector, reducing redundant predictions. Furthermore, our method detects various subjects and objects that appear in the scene rather than focusing on objects that are more likely to have relations.

5. Conclusion

In this study, we propose the lightweight one-stage scene graph generator EGTR. We significantly reduce the model complexity by harnessing the relationships among the object queries learned from the self-attention layers of the object decoder. Furthermore, we devise a novel adaptive smoothing technique that helps multi-task learning of object detection and relation extraction by adjusting the relation label according to the object detection performance. As an auxiliary task of relation extraction, connectivity prediction contributes to the effective learning of EGTR. We conduct extensive experiments and ablation studies, and the results demonstrate that EGTR achieves the highest object detection performance and competitive triplet detection capabilities with the fastest inference speed.

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. 1, 2, 3, 4, 6

[2] Zhe Chen, Shaoli Huang, and Dacheng Tao. Context refinement for object detection. In *ECCV*, pages 71–86, 2018. 2

[3] Yuren Cong, Michael Ying Yang, and Bodo Rosenhahn. Reltr: Relation transformer for scene graph generation. *IEEE TPAMI*, 2023. 1, 3

[4] Naina Dhingra, Florian Ritter, and Andreas Kunz. Bgt-net: Bidirectional gru transformer network for scene graph generation. In *CVPR*, pages 2150–2159, 2021. 2

[5] Lizhao Gao, Bo Wang, and Wenmin Wang. Image captioning with scene-graph based semantic concepts. In *ICMLC*, pages 225–229, 2018. 1

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6

[7] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, pages 3588–3597, 2018. 2, 7

[8] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *CVPR*, pages 3668–3678, 2015. 1

[9] Siddhesh Khandelwal and Leonid Sigal. Iterative scene graph generation. In *NeurIPS*, 2022. 1, 3, 7, 8

[10] Dong-Jin Kim, Jinsoo Choi, Tae-Hyun Oh, and In So Kweon. Dense relational captioning: Triple-stream networks for relationship-based captioning. In *CVPR*, pages 6271–6280, 2019. 1

[11] Rajat Koner, Suprosanna Shit, and Volker Tresp. Relation transformer network. *ECCV*, pages 422–439, 2022. 2

[12] 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*, 123:32–73, 2017. 2, 5, 6

[13] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 128(7):1956–1981, 2020. 2, 6

[14] Linjie Li, Zhe Gan, Yu Cheng, and Jingjing Liu. Relation-aware graph attention network for visual question answering. In *ICCV*, pages 10313–10322, 2019. 1

[15] Lin Li, Guikun Chen, Jun Xiao, Yi Yang, Chunping Wang, and Long Chen. Compositional feature augmentation for unbiased scene graph generation. In *ICCV*, pages 21685–21695, 2023. 2

[16] Rongjie Li, Songyang Zhang, Bo Wan, and Xuming He. Bipartite graph network with adaptive message passing for unbiased scene graph generation. In *CVPR*, pages 11109–11119, 2021. 1, 2, 7

[17] Rongjie Li, Songyang Zhang, and Xuming He. Sgtr: End-to-end scene graph generation with transformer. In *CVPR*, pages 19486–19496, 2022. 1, 3, 6, 7

[18] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 7

[19] Xin Lin, Changxing Ding, Jinquan Zeng, and Dacheng Tao. Gps-net: Graph property sensing network for scene graph generation. In *CVPR*, pages 3746–3753, 2020. 1, 2, 7

[20] Hengyue Liu, Ning Yan, Masood Mortazavi, and Bir Bhanu. Fully convolutional scene graph generation. In *CVPR*, pages 11546–11556, 2021. 1, 3, 7

[21] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, pages 21–37, 2016. 6

[22] Yong Liu, Ruiping Wang, Shiguang Shan, and Xilin Chen. Structure inference net: Object detection using scene-level context and instance-level relationships. In *CVPR*, pages 6985–6994, 2018. 2

[23] Cewu Lu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. Visual relationship detection with language priors. In *ECCV*, pages 852–869, 2016. 2

[24] Yichao Lu, Himanshu Rai, Jason Chang, Boris Knyazev, Guangwei Yu, Shashank Shekhar, Graham W Taylor, and Maksims Volkovs. Context-aware scene graph generation with seq2seq transformers. In *ICCV*, pages 15931–15941, 2021.

[25] Yukuan Min, Aming Wu, and Cheng Deng. Environment-invariant curriculum relation learning for fine-grained scene graph generation. In *ICCV*, pages 13296–13307, 2023. 2

[26] Alejandro Newell and Jia Deng. Pixels to graphs by associative embedding. *NeurIPS*, 30, 2017. 1, 3

[27] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NeurIPS*, 28, 2015. 2, 7

[28] Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D Manning. Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In *Proceedings of the fourth workshop on vision and language*, pages 70–80. Association for Computational Linguistics, 2015. 1

[29] Suprosanna Shit, Rajat Koner, Bastian Wittmann, Johannes Paetzold, Ivan Ezhov, Hongwei Li, Jiazhen Pan, Sahand Sharifzadeh, Georgios Kaissis, Volker Tresp, et al. Relationformer: A unified framework for image-to-graph generation. In *ECCV*, pages 422–439, 2022. 1, 3, 7

[30] Gopika Sudhakaran, Devendra Singh Dhami, Kristian Kersting, and Stefan Roth. Vision relation transformer for unbiased scene graph generation. In *ICCV*, pages 21882–21893, 2023. 2

[31] Mohammed Suhail, Abhay Mittal, Behjat Siddiquie, Chris Broaddus, Jayan Eledath, Gerard Medioni, and Leonid Sigal. Energy-based learning for scene graph generation. In *CVPR*, pages 13936–13945, 2021. 7

[32] 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. 2, 3, 6

[33] 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. 1, 2, 7

[34] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training. In *CVPR*, pages 3716–3725, 2020. 7

[35] Yao Teng and Limin Wang. Structured sparse r-cnn for direct scene graph generation. In *CVPR*, pages 19437–19446, 2022. 1, 3, 6, 7, 8

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017. 2, 3

[37] Sanghyun Woo, Dahun Kim, Donghyeon Cho, and In So Kweon. Linknet: Relational embedding for scene graph. *NeurIPS*, 31, 2018. 2

[38] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, pages 1492–1500, 2017. 7

[39] 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, 2, 7

[40] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *ECCV*, pages 670–685, 2018. 6

[41] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-encoding scene graphs for image captioning. In *CVPR*, pages 10685–10694, 2019. 1

[42] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *CVPR*, pages 5831–5840, 2018. 1, 2, 6, 7

[43] Cheng Zhang, Wei-Lun Chao, and Dong Xuan. An empirical study on leveraging scene graphs for visual question answering. In *BMVC*, 2019. 1

[44] Hanwang Zhang, Zawlin Kyaw, Shih-Fu Chang, and Tat-Seng Chua. Visual translation embedding network for visual relation detection. In *CVPR*, page 3107–3115, 2017. 7

[45] Chaofan Zheng, Xinyu Lyu, Yuyu Guo, Pengpeng Zeng, Jingkuan Song, and Lianli Gao. Learning to generate scene graph from head to tail. In *ICME*, pages 1–6. IEEE, 2022. 2

[46] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2, 3, 6