

# Self-Supervised Relation Alignment for Scene Graph Generation

Bicheng Xu<sup>1,2</sup>

<sup>1</sup>University of British Columbia

bichengx@cs.ubc.ca

Renjie Liao<sup>1,2,3</sup>

<sup>2</sup>Vector Institute for AI

rjliao@ece.ubc.ca

Leonid Sigal<sup>1,2,3</sup>

<sup>3</sup>Canada CIFAR AI Chair

lsigal@cs.ubc.ca

## Abstract

The goal of scene graph generation is to predict a graph from an input image, where nodes correspond to identified and localized objects and edges to their corresponding interaction predicates. Existing methods are trained in a fully supervised manner and focus on message passing mechanisms, loss functions, and/or bias mitigation. In this work we introduce a simple-yet-effective self-supervised relational alignment regularization designed to improve the scene graph generation performance. The proposed alignment is general and can be combined with any existing scene graph generation framework, where it is trained alongside the original model’s objective. The alignment is achieved through distillation, where an auxiliary relation prediction branch, that mirrors and shares parameters with the supervised counterpart, is designed. In the auxiliary branch, relational input features are partially masked prior to message passing and predicate prediction. The predictions for masked relations are then aligned with the supervised counterparts after the message passing. We illustrate the effectiveness of this self-supervised relational alignment in conjunction with two scene graph generation architectures, SGTR [25] and Neural Motifs [53], and show that in both cases we achieve significantly improved performance.

## 1. Introduction

Scene graph generation has emerged as a core problem in computer vision in recent years. The task involves producing a graph-based representation of the scene from an image. As the name suggests, scene graph representation, encodes a scene as a graph where *nodes* correspond to objects, with corresponding (bounding box) locations and class labels, and directed *edges* encode pairwise relations among these objects. The ultimate goal of scene graph generation is to produce such representations from raw images (or videos [17]). Such representations have proved valuable for a variety of higher-level AI tasks, *e.g.*, image/video captioning [27, 34], and visual question answering [38].

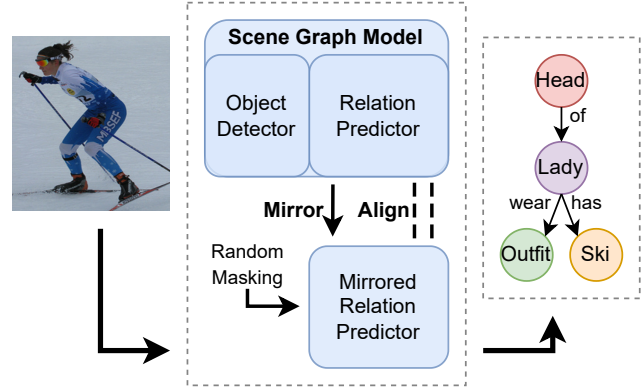


Figure 1. **Illustration of our proposed self-supervised relation alignment mechanism.** A scene graph generation model typically contains an object detector to detect objects of interest, and a relation predictor to predict relations among the detected objects. We create a mirror copy of the original relation predictor, apply random masking on the input to the mirrored relation predictor, and align the relation predictions between the mirrored copy and the original one.

Scene graph generation models usually contain two message passing networks: one for detecting objects of interest, and the other for relation prediction among the detected objects. The message passing network can be a recurrent neural network [44, 53], a graph neural network [42, 50], or more recently, a stack of Transformer blocks [12, 23, 25]. Newly developed architectures continue to push state-of-the-art in terms of performance. However, one of the fundamental challenges in scene graph generation is sparsity of data, owing to large number of objects, relations and their exponential cross product. Recent approaches have tried to address this by mediating dataset bias in relational classes with either re-sampling [9, 25, 26] or loss re-weighting [19] techniques. While this mediates bias among relation predicates, it does not address the core problem of data sparsity, where even for the frequent relations, it is often difficult to observe all appearance variations in objects and their interactions from a dataset like Visual Genome [22] or similar.

Equipped with this intuition, we propose a novel self-

supervised relational alignment mechanism to alleviate this limitation. Our mechanism is simple, effective and generic, *i.e.*, can be combined with any underlying scene graph generation architecture. Consequently, we illustrate it in conjunction with both a traditional two-stage scene graph generation approach – Neural Motifs [53] and a recent Transformer-based SGTR [25]. Specifically, given a scene graph generation model with its original objective, we create an auxiliary (mirrored) relation prediction branch which is identical in structure and shares parameters with the main (original) branch, but whose only goal is to align relational predictions with those coming from the main supervised branch, as shown in Figure 1. The *alignment* is implemented using KL divergence over predicted predicate label distributions. Importantly, In the mirrored branch, we apply a random masking on the relational feature input<sup>1</sup> prior to the relational message passing / prediction head. The goal of this masking is to form impoverished augmented views of the data, with the alignment objective driving *denoising*, which enhances feature formation and refinement within and across relations. Further, unlike other self-supervised methods that tend to pre-train models using self-supervision and then fine-tune them, we train both the supervised and auxiliary branches jointly.

**Contributions.** Our contributions are as follows. First, we introduce a simple, but effective and general self-supervised alignment mechanism for relation prediction. This alignment is achieved by instantiating an auxiliary branch of the relation prediction network that shares structures and parameters with the original scene graph model (whatever it may be). In this auxiliary branch we create alternate views of relation features, through random masking, ahead of the message passing. The corresponding relational predictions are then aligned with the predicate distributions obtained from the original relational branch (main branch). Second, unlike other self-supervised learning approaches that require two stage training, first self-supervised pre-training and then fine-tuning to the task. We show that we can leverage self-supervised alignment as an effective *regularization* during training of the main branch, *i.e.*, allowing us to train the two branches jointly and at the same time. Third, we illustrate the effectiveness of our self-supervision, implemented via alignment, by pairing it with two very diverse scene graph generation architectures (Neural Motifs [53] and SGTR [25]) and showing significant improvements in both cases, without the need to change the original architectures, losses or training in any way.

<sup>1</sup>The specifics of this depend on the structure of the original scene graph generation model.

## 2. Related Work

### 2.1. Scene Graph Generation (SGG)

Works on scene graph generation can be categorized into two classes: two-stage and one-stage.

*Two-stage models* first train a Faster-RCNN [40] (or similar) object detector, and use this trained object detector to obtain image features and bounding box proposals (including their locations and feature representations). With these as fixed inputs, two-stage approaches then design various message passing networks to refine object and relation features and produce the final scene graph by classifying them. Message passing networks within these approaches can take many forms: LSTMs [44, 53], graph neural networks [26, 50], or others [18, 28, 33, 37, 49, 54]. From the objective perspective, while most leverage cross-entropy, energy-based losses [42], that better capture structure in the output space, have also been proposed.

*One-stage models* are more recent and attempt to directly generate the scene graph from a given image without first pre-training an object region proposal network. These approaches overcome the limitation of the aforementioned two-stage models, where the object proposals are fixed and can not be modified by the scene graph generation model. One-stage models, usually build on one-stage object detectors (*e.g.*, Yolo [39] or DETR [2]) and employ network structures like fully convolutional networks [29, 45] or Transformers [8, 10, 12, 19, 25] for predictions.

Our proposed self-supervised relation alignment mechanism is generic, and can be applied to both one-stage and two-stage scene graph generation models. In this paper, we use an one-stage SGTR model [25] and a two-stage Neural Motifs model [53], as examples, to show its effectiveness.

### 2.2. Self-Supervision in SGG

Self-supervised learning is a powerful paradigm to obtain useful feature representations, examples including Bert-type models [1, 11, 24, 30, 32, 41], denoising auto-encoding based approaches [15, 36, 47, 48], and contrastive learning [4, 16, 35, 46]. These methods typically first define some pretext tasks to train a generic feature extractor, and then utilize the extracted features or fine-tune the pre-trained model to a range of downstream tasks.

In the scene graph generation literature, self-supervised learning has not been explored much. Zareian *et al.* [52] use a denoising auto-encoder (DAE) type structure to learn the commonsense that exists in the scene graph labels. Given a scene graph annotation, they mask out some of the object and/or relation annotations, and train a DAE structure to reconstruct these masked out labels. They show that their approach can be applied to any trained scene graph generation model as a post-processing step to correct some predictions which do not follow common sense. Hasegawa *et*

al. [14] proposes a scene graph generation model consisting of a pre-trained object detector, a relational encoder, and object and relation classifiers. They first use self-supervision with a contrastive loss to train the relation encoder, and with the relation encoder fixed, they then train the classifiers to generate the scene graph. Their self-supervised technique is specific to their model, and a pre-trained object detector is required.

In contrast to the multi-stage training or post-processing, we employ random masking as a type of data augmentation and our self-supervision is achieved via aligning the predictions from the masked feature input with those from the unmasked one. Our introduced self-supervised loss is added to the original loss of the scene graph generation model during training.

### 3. Self-Supervised Relation Alignment

In this section, we first describe the general idea behind our proposed self-supervised relation alignment mechanism, and then instantiate it within two popular scene graph generation models: one-staged SGTR [25] and two-staged Neural Motifs [53].

#### 3.1. General Pipeline

Scene graph generation models typically contain three parts: a pre-trained visual feature extractor, an object detector, and a relation predictor. As illustrated in Figure 2, the feature extractor takes an image as input and computes visual features. Conditioned on the extracted features, the object detector localizes and classifies the objects of interest in the image. Finally, the relation predictor predicts the relationship between any pair of detected objects. Depending on the architecture, the relation predictor relies either on the concatenation of features from the pair of objects in question, its own extracted features, or both.

Motivated by the sparsity of visual data for relations, which stems from the possible exponential space of appearances for interacting objects, we focus on building a self-supervised alignment mechanism for the relation predictor. The goal of this alignment is two-fold: (1) to regularize learning with augmented data samples; and (2) to encourage the relation predictor, which includes message passing and refinement of relational features, to more effectively propagate information both within a single relation representation and across such representations.

To build the self-supervised alignment, we first create a duplicate of the relation predictor, called *mirrored relation predictor*. Note that both the neural network architecture and the weights are shared between the mirrored and the original relation predictors (*i.e.*, the weights are initially the same and updated in the same way at each iteration). The input to the mirrored relation predictor is the same as that to the original one, which usually contains image features

from the feature extractor and/or object features from the object detector. We then apply a random masking (details in Section 3.1.1) on the input to the mirrored relation predictor. An alignment loss is introduced to align the relation predictions from the mirrored predictor with those from the original branch. To make sure the introduced alignment loss only affects the relation predictor, we stop the gradient of the alignment loss at the input feature. This ensures that the alignment loss only affects the message passing and feature refinement in the relation predictor. The resulting alignment loss is added to the original training loss.

It is important to mention that the auxiliary branch, in the form of the mirrored relation predictor, is only used during training of the network. Once trained, during evaluation, we still only use the original relation predictor to generate the scene graph. In other words, one can view our self-supervised alignment mechanism as a form of (more sophisticated) regularization. The whole pipeline is shown in Figure 2. In the following, we describe in detail about how the masking is applied and how the alignment is achieved.

#### 3.1.1 Random Masking

The original and the mirrored relation predictors both take a feature matrix  $\mathbf{R} \in \mathbb{R}^{N \times d}$  as input, where  $N$  is the number of relations and  $d$  is the dimensionality of feature vector for each relation. The relation predictor typically refines  $\mathbf{R}$  (*e.g.*, through a series of LSTM, GNN or Transformer layers) and uses the refined representations to classify each feature to produce a relation or no relation label.

We apply random masking on the input to the mirrored relation predictor. Specifically, for each element  $\mathbf{R}_i$  in  $\mathbf{R}$ , the masking operation independently zeros it out with probability  $p$ . That is,

$$\text{masking}(\mathbf{R}_i) = \begin{cases} 0 & \text{w. probability } p, \\ \mathbf{R}_i & \text{w. probability } (1 - p). \end{cases} \quad (1)$$

We then feed the masked input  $\mathbf{R}_{\text{masked}}$  to the mirrored relation predictor, while the original predictor still receives  $\mathbf{R}$ .

#### 3.1.2 Self-Supervised Alignment Loss

With the masked input  $\mathbf{R}_{\text{masked}}$ , the mirrored relation predictor generates relation predictions, *i.e.*, the probability distribution over possible relationships, denoted as  $P_{\text{masked}}$ . We align  $P_{\text{masked}}$  with the relation predictions from the original relation predictor, denoted as  $P_{\text{original}}$ . We use the Kullback-Leibler (KL) divergence to measure the closeness between  $P_{\text{original}}$  and  $P_{\text{masked}}$ . In particular, we fix  $P_{\text{original}}$  and treat it as the target in the KL loss to optimize  $P_{\text{masked}}$ . We term this alignment loss,

$$\mathcal{L}_{\text{align}} = \text{KL}(P_{\text{original}} || P_{\text{masked}}). \quad (2)$$

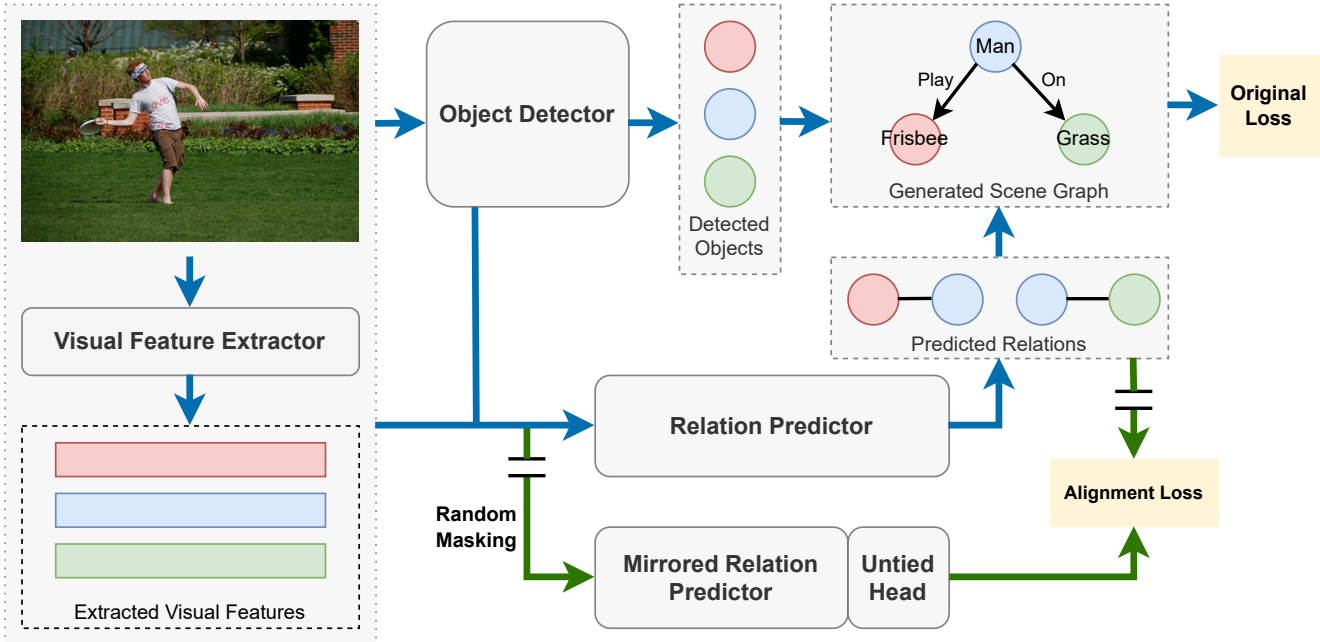


Figure 2. **Our proposed self-supervised relation alignment mechanism.** The blue arrows indicate a standard scene graph generation pipeline, and the green arrows illustrate our proposed relation alignment. The architecture and parameters of Relation and Mirrored Relation Predictors are all *shared* except the last relation projection head, which is denoted as *Untied Head*. This untied projection head helps alleviate the burden of performing two tasks (*i.e.*, supervised relation prediction and self-supervised relation alignment) for the original projection head, resulting in better learned representations. In the figure, = indicates the stopping gradient operation.

We add the alignment loss  $\mathcal{L}_{\text{align}}$  to the original training loss (typically a weighted sum of cross-entropy losses for object and relation predictions with one-hot class targets) in the scene graph generation model, denoted as  $\mathcal{L}_{\text{original}}$ . The final training loss can be formulated as

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{original}} + \lambda \mathcal{L}_{\text{align}}, \quad (3)$$

where  $\lambda$  is a hyper-parameter to control the relative weights between  $\mathcal{L}_{\text{original}}$  and  $\mathcal{L}_{\text{align}}$ .

We use an *untied projection head* to predict the logits of relations in the mirrored relation predictor. This means that the network weights between the mirrored and the original relation predictors are all tied except for the last prediction head. Similar to other self-supervised learning works [4, 5], we observe that having the untied projection head helps alleviate the burden of performing two tasks (*i.e.*, the supervised relation prediction and the self-supervised relation alignment) for the original projection head, resulting in better learned representations.

### 3.2. Instantiation under SGTR

To test the effectiveness of our proposed self-supervised relation alignment, we leverage it with a recently published Transformer-based SGTR model [25]. The choice of SGTR is motivated by two factors: (1) it is a state-of-the-art

scene graph generation approach; and (2) its design mirrors DETR [2], resulting in an one-stage end-to-end method.

SGTR model [25] mainly contains four modules: (1) a *CNN feature extractor* with a multi-layer Transformer encoder to extract image features, (2) an *entity node generator* which predicts objects using a Transformer decoder with learned entity embedding queries conditioned on the image encoding, (3) a *predicate node generator* which predicts relations using a Transformer decoder with learned predicate embedding queries, and (4) a *bipartite graph assembling module* for constructing the final bipartite matching between entities and predicates. This matching results in the final generated scene graph.

The predicate node generator has two main components: (1) one *relation image feature encoder*, which refines the image feature extracted from the feature extraction part, specifically for relation prediction, and (2) a *structural predicate decoder*, which is a stack of self-attention and cross-attention Transformer blocks, to generate relation predictions. The structural predicate decoder receives two inputs, the refined image feature from the relation image feature encoder, and the entity feature from the entity node generator. It then makes three sets of predictions: one predicate/relation label distribution and the relation location, and two entity (the subject and the object related to the relation) label distributions and their locations.

We instantiate our self-supervised relation alignment mechanism on the structural predicate decoder. Specifically, we first duplicate the structural predicate decoder and then apply our proposed masking on the input (image/entity feature) to the mirrored branch. The input to the structural predicate decoder is treated as `key` and `value` of its cross-attention Transformer blocks. Rather than applying the random masking on the input directly, inside a cross-attention Transformer block, the random mask is multiplied with the attention matrix in an element-wise fashion. In other words, each element in the attention matrix has an independent probability  $p$  to be set to  $-\text{inf}$  before fed to the softmax operation. We generate a random mask per cross-attention Transformer block independently.

The alignment loss is enforced on the three label distributions predicted by the mirrored structural predicate decoder. Although the original structural predicate decoder makes predictions at each Transformer layer, we only apply the alignment loss at the last layer for simplicity. As described in Section 3.1.2, we untie the last projection layer of the mirrored predicate decoder with the original branch.

### 3.3. Instantiation under Neural Motifs

To illustrate the generality of the self-supervised relational alignment, we also apply it to one of the most popular two-stage architectures for scene graph generation – Neural Motifs [53]. Neural Motifs is built upon the FasterRCNN [40] object detector. It first utilizes a pre-trained object detector to extract input image features and generate object proposals. It then uses RoI Align operation to extract the object features and the relation features per pair of object proposals. Relying on these extracted features, it predicts a scene graph for a given input image.

Neural Motifs first inputs the object features and the object proposal location embeddings to a bi-directional LSTM (*object context LSTM*) to predict object class labels. The predicted object labels, along with the object features, and the object context LSTM’s hidden states, are then fed to another bi-directional LSTM (*edge context LSTM*) to generate more refined object features. It then combines a pair of refined object features with the extracted relation feature and learned predicate prior (obtained from the empirical distribution of predicates in training data) to form a feature that is used to predict a relation label per pair of object proposals.

We apply our self-supervised relation alignment mechanism to the relation predictor. Specifically, we duplicate the edge context LSTM and its subsequent prediction layers. Random masks are applied on both input to the edge context LSTM and the relation feature. We replace the last two linear layers of the mirrored relation predictor with a three-layer MLP with ReLU activations. Similarly as before, this MLP is untied from the original relation predictor. In the mirrored predictor, the pairwise combined object features

and the masked relation features are fed to the MLP which predicts the relation label distributions. The predicate prior is not used in the mirrored relation predictor.

The alignment loss is again calculated between the relation label distributions from the mirrored and the original relation predictors.

## 4. Experiments

**Dataset.** We conduct experiments on the widely used scene graph generation dataset, Visual Genome [22]. We use the same pre-processing procedure and train/val/test splits as previous works [25, 42, 49, 53]. This pre-processed Visual Genome dataset has 150 object and 50 relation categories.

**Evaluation Metrics.** We mainly use the mean-Recall@K (**mR@K**) metric to evaluate models, following previous works [6, 43, 44]. Mean-Recall averages the Recall values computed for each relation class, which is a better evaluation metric than Recall, since it is less biased to dominant classes as suggested in [43]. For a full comparison on the SGTR model, same as [25], we also report the Recall@K (**R@K**) values and mR@100 for each long-tail category subset (**HEAD**, **BODY**, and **TAIL**), as proposed in [26]. The evaluation for the Neural Motifs model is conducted under three settings, (1) predicate classification (**PredCls**): predicting the relation labels given the image, ground-truth object bounding box locations and labels; (2) scene graph classification (**SGCls**): predicting both object labels and relation labels given the image and ground-truth bounding box locations; and (3) scene graph detection (**SGDet**): generating the whole scene graph based on the input image. Since there is no operation similar to RoI Align existing in the SGTR model, we only evaluate the SGTR model under the SGDet setting following [25].

### 4.1. SGTR with Relation Alignment

**Implementation Details.** We use the codebase released by [25] to conduct the experiments. The pre-trained DETR [2], learning rate scheduler, and all other hyperparameter values are the same as those provided in the codebase. There are dropout operations throughout SGTR. For the SGTR model where our self-supervised relation alignment mechanism is applied, to eliminate the effect from the dropout, we turn off all the dropout operations inside the mirrored structural predicate decoder, and we conduct another forward pass of the original decoder without the dropout operations to form the target of the alignment loss. We set the masking probability  $p$  (in Equation 1) to 0.1 and the  $\lambda$  (in Equation 3) to 10 for all three prediction alignments (the relation, the subject and the object related to the relation). Since our alignment mechanism introduces some randomness, we run both the original SGTR model and our self-supervised aligned SGTR variant 4 times and report the

average results along with the standard deviation. We select the best trained model for testing based on the mR@50 on the validation set.

**Results.** Quantitative results are displayed in Table 1. Our retrained SGTR model (denoted as SGTR\*) and the SGTR variant with our proposed self-supervised relation alignment during training (denoted as Align-SGTR\*) are listed at the last two rows of the table. To have a complete comparison with existing scene graph generation models, we also show the results of other competitors reported in [25]. There is some performance drop on Recalls comparing the results from the models trained using the DETR detector with those from the Faster RCNN detector. The main reason is that DETR is not as good as Faster RCNN in detecting small objects, while in the Visual Genome dataset more than half of the relations involve small objects; this was also discussed in [25].

The results of our retrained SGTR model (SGTR\*) match those reported in [25]. In particular, we obtain slightly higher BODY and TAIL results but slightly worse HEAD result compared to the reported, which results in slightly higher mean-Recalls, but slightly worse Recalls. This is reasonable, because it is now well-known that for a specific model, increase in mean-Recalls often leads to decrease in Recalls [33, 43, 51].

Comparing the results between SGTR\* and our Align-SGTR\*, it is clear that the self-supervised alignment provides consistent and significant (at least two standard deviation higher) improvements on *all* evaluation metrics. This demonstrates the effectiveness of our proposed mechanism. Our proposed self-supervised relation alignment is simple; it only introduces the extra network weights for the additional untied projection head and a KL loss during training, but very effective. By comparing the results between Align-SGTR\* and all other scene graph generation models in the table, Align-SGTR\* achieves the best mean-Recall, BODY, and TAIL results.

## 4.2. Neural Motifs with Relation Alignment

**Implementation Details.** Neural Motifs, as other two-stage scene graph generation models, requires pre-training an object detector. We first train a Faster RCNN object detector with VGG-16 backbone using the codebase provided by [43], and then train the Neural Motifs model based on that. While training the Neural Motifs model, we use a batch size of 24, and SGD optimizer with momentum 0.9 and weight decay 0.0001. We set the initial learning rate to 0.01 and use the cosine annealing schedule [31] to decay the learning rate during training, with minimum learning rate set to 0. Learning rate is updated every 1K iterations and we train the model for a total of 100K iterations. We select the best trained model for testing based on the validation mR@50. For our self-supervised relation alignment, we set

the masking probability  $p$  (in Equation 1) to 0.1 and the  $\lambda$  (in Equation 3) to 10. The alignment loss is only added for the relation prediction. We run the training of both the Neural Motifs model and our self-supervised relation aligned variant 4 times.

**Results.** Quantitative results are shown in Table 2. The results of our retrained Neural Motifs model (denoted as Motifs\*) and the variant with our self-supervised relation alignment (denoted as Align-Motifs\*) are listed at the last two rows of the table. We again report the mean and the standard deviation of the results collected from 4 runs. We also copy the results of other scene graph generation models from [20]. All these competitors use Faster RCNN (with VGG-16 backbone) as the object detector. The results of our re-trained Motifs\* roughly match those of Motifs<sup>†</sup> provided by [20]. The main discrepancy of the results is caused by the fact that we use a different codebase to train the Faster RCNN object detector than theirs.

From Table 2, we can see that our Align-Motifs\* achieves significantly better (larger than at least one standard deviation) results than those of Motifs\* on *all* metrics. Moreover, under the PredCls and SGCls settings, our Align-Motifs\* achieves the best results among all the (Faster RCNN with VGG-16 backbone) models. The improvements of Align-Motifs\* over Motifs\* across the three evaluation settings are PredCls > SGCls > SGDet. This is reasonable, because our proposed self-supervised alignment mechanism is only applied to relation prediction. Applying a similar self-supervised alignment mechanism to the object prediction part could potentially further improve the performance under the SGCls and SGDet settings. One possible reason for the marginal improvement under the SGDet setting is that the pre-trained Faster RCNN is fixed during Neural Motif’s training. If the object proposals are not good (*e.g.*, the detection recall is low), then the performance of the scene graph generation model will be significantly reduced. This is an inherent limitation of a two-stage scene graph generation model.

We visualize some qualitative results for relation prediction in Figure 3. These results are obtained from Motifs\* and Align-Motifs\* models trained under the PredCls setting. The qualitative results confirm the performance gain shown in the quantitative results. The Align-Motifs\* model generally generates more accurate relation predictions than the Motifs\* model.

## 4.3. Ablation Study

We conduct an ablation study based on the SGTR model to verify the effectiveness of our design choices: (1) our self-supervised alignment vs. a supervised alignment, and (2) the untied projection head vs. a shared projection head in the mirrored relation predictor. Experimental results are listed in Table 3. In the table, the performance of the base-

| <b>B</b> | <b>D</b>    | <b>Method</b>               | <b>mR@50</b>      | <b>mR@100</b>     | <b>R@50</b> | <b>R@100</b> | <b>HEAD</b> | <b>BODY</b>       | <b>TAIL</b>      |
|----------|-------------|-----------------------------|-------------------|-------------------|-------------|--------------|-------------|-------------------|------------------|
| *        | *           | FCSGG [29]                  | 3.6               | 4.2               | 21.3        | 25.1         | -           | -                 | -                |
| X101-FPN | Faster RCNN | RelDN [54]                  | 6.0               | 7.3               | 31.4        | 35.9         | -           | -                 | -                |
|          |             | Motifs [43]                 | 5.5               | 6.8               | <b>32.1</b> | <b>36.9</b>  | -           | -                 | -                |
|          |             | VCTree [43]                 | 6.6               | 7.7               | 31.8        | 36.1         | -           | -                 | -                |
|          |             | VCTree-TDE [43]             | 9.3               | 11.1              | 19.4        | 23.2         | -           | -                 | -                |
|          |             | VCTree-DLFE [7]             | 11.8              | 13.8              | 22.7        | 26.3         | -           | -                 | -                |
|          |             | VCTree-EBM [42]             | 9.7               | 11.6              | 20.5        | 24.7         | -           | -                 | -                |
|          |             | VCTree-BPLSA [13]           | 13.5              | 15.7              | 21.7        | 25.5         | -           | -                 | -                |
|          |             | RelDN <sup>‡</sup> [25, 54] | 4.4               | 5.4               | 30.3        | 34.8         | <b>31.3</b> | 2.3               | 0.0              |
| R101     | DETR        | AS-Net <sup>‡</sup> [3, 25] | 6.1               | 7.2               | 18.7        | 21.1         | 19.6        | 7.7               | 2.7              |
|          |             | HOTR <sup>‡</sup> [21, 25]  | 9.4               | 12.0              | 23.5        | 27.7         | 26.1        | 16.2              | 3.4              |
|          |             | SGTR [25]                   | 12.0              | 15.2              | 24.6        | 28.4         | 28.2        | 18.6              | 7.1              |
|          |             | SGTR*                       | 12.0 ± 0.4        | 16.0 ± 0.5        | 23.7 ± 0.4  | 26.9 ± 0.4   | 26.4 ± 0.3  | 20.0 ± 0.3        | 8.8 ± 0.8        |
|          |             | Align-SGTR*                 | <b>12.7 ± 0.3</b> | <b>16.8 ± 0.2</b> | 24.6 ± 0.2  | 27.8 ± 0.2   | 27.3 ± 0.2  | <b>20.8 ± 0.1</b> | <b>9.8 ± 0.5</b> |

Table 1. **SGDet test results** for existing scene graph generation models on the Visual Genome dataset. Numbers are borrowed from [25] except for the last two rows. **B** means the backbone used for object detection, and **D** means the object detector type. \* denotes the special backbone HRNetW48-5S-FPN<sub>x2-f</sub> and object detector CenterNet [55] used in [29]. RelDN<sup>‡</sup>, AS-Net<sup>‡</sup>, and HOTR<sup>‡</sup> are trained by [25]. SGTR\* is the original SGTR [25] model which we train and evaluate 4 times using the authors’ released code, and Align-SGTR\* is the SGTR model equipped with our proposed self-supervised relation alignment mechanism during training, for which we also run 4 times. Note that the structure and cost during inference for SGTR\* and Align-SGTR\* are identical; they only differ in training.

| <b>Method</b>                             | <b>PredCls</b>    |                   |                   | <b>SGCls</b>     |                   |                   | <b>SGDet</b> |              |               |
|---|-------------------|-------------------|-------------------|------------------|-------------------|-------------------|--------------|--------------|---------------|
| <b>B: VGG-16</b><br><b>D: Faster RCNN</b> | <b>mR@20</b>      | <b>mR@50</b>      | <b>mR@100</b>     | <b>mR@20</b>     | <b>mR@50</b>      | <b>mR@100</b>     | <b>mR@20</b> | <b>mR@50</b> | <b>mR@100</b> |
| IMP [49]                                  | -                 | 9.8               | 10.5              | -                | 5.8               | 6.0               | -            | 3.8          | 4.8           |
| Motifs [53]                               | 10.8              | 14.0              | 15.3              | 6.3              | 7.7               | 8.2               | 4.2          | 5.7          | 6.6           |
| VCTree [44]                               | 14.0              | 17.9              | 19.4              | <b>8.2</b>       | 10.1              | 10.8              | <b>5.2</b>   | <b>6.9</b>   | <b>8.0</b>    |
| Motifs <sup>‡</sup> [20, 53]              | 13.7              | 17.5              | 18.9              | 7.5              | 9.2               | 9.8               | <b>5.2</b>   | 6.8          | 7.9           |
| Motifs*                                   | 15.0 ± 0.8        | 19.8 ± 0.9        | 21.9 ± 0.9        | 7.2 ± 0.3        | 8.9 ± 0.4         | 9.6 ± 0.4         | 4.2 ± 0.4    | 5.9 ± 0.5    | 7.2 ± 0.3     |
| Align-Motifs*                             | <b>16.5 ± 0.4</b> | <b>21.7 ± 0.5</b> | <b>24.2 ± 0.5</b> | <b>8.2 ± 0.2</b> | <b>10.5 ± 0.2</b> | <b>11.5 ± 0.3</b> | 4.4 ± 0.2    | 6.4 ± 0.3    | 7.8 ± 0.3     |

Table 2. **PredCls, SGCls, SGDet test results** on the Visual Genome dataset for models with VGG-16 backbone (**B**) and Faster-RCNN detector (**D**). Numbers are borrowed from [20] except for the last two rows. Motifs<sup>‡</sup> is trained by [20]. Motifs\* is our trained Neural Motifs [53] model, and Align-Motifs\* is the Neural Motifs model containing our proposed self-supervised relation alignment mechanism during training, and for both we train the models 4 times.

line SGTR is our reproduced result. These are validation results obtained from a single run on the same machine with the same random seed.

**Self-Supervised vs. Supervised Alignment.** We first compare our self-supervised alignment loss to a supervised one. Particularly, we construct a supervised relation alignment module by following the exact same setting as the self-supervised module except that we remove the KL loss and plug the same supervised loss in the mirrored relation predictor. This supervised loss will push the mirrored predictor towards predicting the correct relation labels (*i.e.*, aligning

with the ground truth labels) whereas the self-supervised loss will push it to align with the original predictor. By switching our self-supervised loss to a supervised one, the mean-Recall results are reduced by almost 1 point. Due to the random masking applied on the input, placing a supervised loss on the mirrored branch only slightly improves the robustness of the learned representations. However, our self-supervised loss acts differently. It is like the distillation or a student-teacher model, *i.e.*, the mirrored predictor learns to mimic the original one. This provides a more informative label distribution supervision, *e.g.*, with a higher

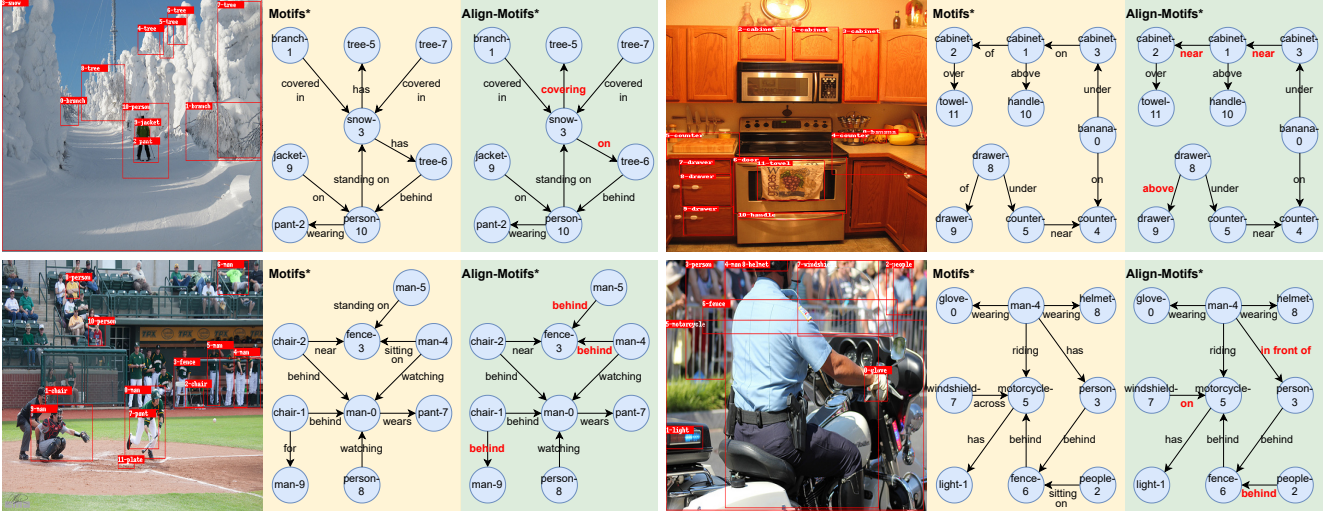


Figure 3. **Qualitative Results.** Relation prediction results from Motifs\* and Align-Motifs\* trained under the PredCls setting.

| Method           | mR@50       | mR@100      | R@50        | R@100       |
|------------------|-------------|-------------|-------------|-------------|
| SGTR             | 14.9        | 18.4        | 24.0        | 27.4        |
| SGTR + SA + UPH  | 15.0        | 18.6        | 24.5        | 27.8        |
| SGTR + SSA + PH  | 15.9        | 19.2        | 24.6        | <b>27.9</b> |
| SGTR + SSA + UPH | <b>16.0</b> | <b>20.5</b> | <b>24.8</b> | <b>27.9</b> |

Table 3. **Ablation study with validation results.** SGTR is re-trained by us. Self-supervised alignment (SSA) has one supervised loss in the original predictor and a KL loss between the original and the mirrored predictors. Supervised alignment (SA) has two supervised losses on the original and the mirrored predictors respectively. PH/UPH denotes the tied/untied projection head in the mirrored relation predictor.

entropy compared to the one-hot ground truth labels (Dirac delta distribution). It can also provide some supervision on unlabeled relation examples, which often appears as the ground truth relation labels are incomplete and very long-tail distributed.

**Untied vs. Tied Projection Head.** We then validate the effect of the untied projection head vs. a tied projection head in the mirrored relation predictor, where the weights of the relation logits predictor are also shared between the mirrored and the original predictors. As can be seen, though the self-supervised alignment loss alone already brings a significant benefit, the untied projection head further improves the performance. Adding this extra projection head helps alleviate the burden of performing two tasks (*i.e.*, the supervised relation prediction and the self-supervised relation alignment) simultaneously, resulting in better learned representations. Similar observation has been made in other self-supervised learning works [4, 5].

## 5. Conclusion

We propose a simple-yet-effective self-supervised relational alignment module that can be plugged into any existing scene graph generation model as an additional loss term. Particularly, we mirror the relation prediction branch and feed randomly masked input to the mirrored branch. The alignment between the predictions from the mirrored and the original branches encourages the model to learn better representations. Experiments with two scene graph models (one one-stage, and one two-stage) on the Visual Genome dataset show that our method significantly improves performance. In the future, we plan to design more sophisticated random masking patterns and explore this method in the setting where a larger scale of unsupervised data is available.

**Discussion on potential negative societal impact.** The scene graph generation task is for better representation of a given input image. The task itself and the models built for the task are neutral; potential negative societal impact may be introduced in how human beings using this representation. The method proposed in our work aims to improve the performance of existing models on this specific task. We do not introduce any further potential negative societal impact.

**Acknowledgments.** This work was funded, in part, by the Vector Institute for AI, Canada CIFAR AI Chairs, NSERC CRC, and NSERC DGs. Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, the Digital Research Alliance of Canada [alliance.can.ca](http://alliance.can.ca), [companies](http://companies) sponsoring the Vector Institute, and Advanced Research Computing at the University of British Columbia. Additional hardware support was provided by John R. Evans Leaders Fund CFI grant and Compute Canada under the Resource Allocation Competition award.



## References

- [1] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. [2](#)
- [2] 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. [2](#), [4](#), [5](#)
- [3] 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. [7](#)
- [4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. [2](#), [4](#), [8](#)
- [5] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255, 2020. [4](#), [8](#)
- [6] Tianshui Chen, Weihao Yu, Riquan Chen, and Liang Lin. Knowledge-embedded routing network for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6163–6171, 2019. [5](#)
- [7] Meng-Jiun Chiou, Henghui Ding, Hanshu Yan, Changhu Wang, Roger Zimmermann, and Jiashi Feng. Recovering the unbiased scene graphs from the biased ones. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 1581–1590, 2021. [7](#)
- [8] Yuren Cong, Michael Ying Yang, and Bodo Rosenhahn. Reltr: Relation transformer for scene graph generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. [2](#)
- [9] Alakh Desai, Tz-Ying Wu, Subarna Tripathi, and Nuno Vasconcelos. Learning of visual relations: The devil is in the tails. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15404–15413, 2021. [1](#)
- [10] Alakh Desai, Tz-Ying Wu, Subarna Tripathi, and Nuno Vasconcelos. Single-stage visual relationship learning using conditional queries. *Advances in Neural Information Processing Systems*, 35:13064–13077, 2022. [2](#)
- [11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. [2](#)
- [12] Qi Dong, Zhuowen Tu, Haofu Liao, Yuting Zhang, Vijay Mahadevan, and Stefano Soatto. Visual relationship detection using part-and-sum transformers with composite queries. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3550–3559, 2021. [1](#), [2](#)
- [13] Yuyu Guo, Lianli Gao, Xuanhan Wang, Yuxuan Hu, Xing Xu, Xu Lu, Heng Tao Shen, and Jingkuan Song. From general to specific: Informative scene graph generation via balance adjustment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16383–16392, 2021. [7](#)
- [14] So Hasegawa, Masayuki Hiromoto, Akira Nakagawa, and Yuhei Umeda. Improving predicate representation in scene graph generation by self-supervised learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2740–2749, 2023. [3](#)
- [15] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. [2](#)
- [16] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. [2](#)
- [17] Jingwei Ji, Ranjay Krishna, Li Fei-Fei, and Juan Carlos Niebles. Action genome: Actions as compositions of spatio-temporal scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. [1](#)
- [18] Deunsol Jung, Sanghyun Kim, Won Hwa Kim, and Minsu Cho. Devil’s on the edges: Selective quad attention for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18664–18674, 2023. [2](#)
- [19] Siddhesh Khandelwal and Leonid Sigal. Iterative scene graph generation. *Advances in Neural Information Processing Systems*, 35:24295–24308, 2022. [1](#), [2](#)
- [20] Siddhesh Khandelwal, Mohammed Suhail, and Leonid Sigal. Segmentation-grounded scene graph generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15879–15889, 2021. [6](#), [7](#)
- [21] 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. [7](#)
- [22] 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. *International journal of computer vision*, 123(1):32–73, 2017. [1](#), [5](#)
- [23] Sanjoy Kundu and Sathyanarayanan N Aakur. Is-ggt: Iterative scene graph generation with generative transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6292–6301, 2023. [1](#)
- [24] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019. [2](#)
- [25] Rongjie Li, Songyang Zhang, and Xuming He. Sgtr: End-to-end scene graph generation with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pat-*

- tern Recognition*, pages 19486–19496, 2022. 1, 2, 3, 4, 5, 6, 7, 12
- [26] 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, 5, 12
- [27] X. Li and S. Jiang. Know more say less: Image captioning based on scene graphs. In *IEEE Transactions on Multimedia*, pages 2117—2130, 2019. 1
- [28] Xin Lin, Changxing Ding, Jinquan Zeng, and Dacheng Tao. Gps-net: Graph property sensing network for scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3746–3753, 2020. 2
- [29] Hengyue Liu, Ning Yan, Masood Mortazavi, and Bir Bhanu. Fully convolutional scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11546–11556, 2021. 2, 7
- [30] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. 2
- [31] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 6
- [32] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019. 2
- [33] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15931–15941, 2021. 2, 6
- [34] Kien Nguyen, Subarna Tripathi, Bang Du1, Tanaya Guha, and Truong Q. Nguye. In defense of scene graphs for image captioning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 1
- [35] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. 2
- [36] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 2
- [37] Mengshi Qi, Weijian Li, Zhengyuan Yang, Yunhong Wang, and Jiebo Luo. Attentive relational networks for mapping images to scene graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3957–3966, 2019. 2
- [38] Tianwen Qian, Jingjing Chen, Shaoxiang Chen, Bo Wu, and Yu-Gang Jiang. Scene graph refinement network for visual question answering. In *IEEE Transactions on Multimedia*, 2022. 1
- [39] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2016. 2
- [40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 2, 5
- [41] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vi-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019. 2
- [42] Mohammed Suhail, Abhay Mittal, Behjat Siddiquie, Chris Broaddus, Jayan Eledath, Gerard Medioni, and Leonid Sigal. Energy-based learning for scene graph generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13936–13945, 2021. 1, 2, 5, 7
- [43] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3716–3725, 2020. 5, 6, 7
- [44] Kaihua Tang, Hanwang Zhang, Baoyuan Wu, Wenhan Luo, and Wei Liu. Learning to compose dynamic tree structures for visual contexts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6619–6628, 2019. 1, 2, 5, 7
- [45] Yao Teng and Limin Wang. Structured sparse r-cnn for direct scene graph generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19437–19446, 2022. 2
- [46] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020. 2
- [47] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010. 2
- [48] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simsim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9653–9663, 2022. 2
- [49] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5419, 2017. 2, 5, 7
- [50] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 670–685, 2018. 1, 2
- [51] Alireza Zareian, Svebor Karaman, and Shih-Fu Chang. Bridging knowledge graphs to generate scene graphs. In

- European conference on computer vision*, pages 606–623. Springer, 2020. [6](#)
- [52] Alireza Zareian, Zhecan Wang, Haoxuan You, and Shih-Fu Chang. Learning visual commonsense for robust scene graph generation. In *European Conference on Computer Vision*, pages 642–657. Springer, 2020. [2](#)
- [53] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5831–5840, 2018. [1](#), [2](#), [3](#), [5](#), [7](#), [12](#)
- [54] Ji Zhang, Kevin J Shih, Ahmed Elgammal, Andrew Tao, and Bryan Catanzaro. Graphical contrastive losses for scene graph parsing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11535–11543, 2019. [2](#), [7](#)
- [55] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019. [7](#)

# Self-Supervised Relation Alignment for Scene Graph Generation (Supplementary Material)

## 6. Hyperparameters

### 6.1. Model Performance w.r.t. the Masking Ratio

We conduct ablation experiments with respect to the masking ratio  $p$  (in Equation 1 of the main paper) on the Neural Motifs [53] model under the PredCls setting. Table 4 below shows the validation mean-Recall results with different  $p$  values, which are obtained from a single run on the same machine with the same random seed. The loss weight  $\lambda$  (in Equation 3 of the main paper) is fixed to 10 for all the experiments. As can be seen,  $p$  being 0.1 gives us the best validation results among the values we tried. Based on this ablation, we set  $p$  to be 0.1 for other Neural Motif’s training settings (SGCls and SGDet), and for SGTR. Notably, even with a sub-optimal  $p$  value, over a wide range of  $p$  values, we obtain significant improvements over the Motifs baseline.

| Method                      | mR@50       | mR@100      |
|-----------------------------|-------------|-------------|
| Motifs                      | 22.1        | 24.2        |
| Align-Motifs ( $p = 0.05$ ) | 23.7        | 25.9        |
| Align-Motifs ( $p = 0.1$ )  | <b>23.9</b> | <b>26.2</b> |
| Align-Motifs ( $p = 0.2$ )  | 23.5        | 25.8        |
| Align-Motifs ( $p = 0.4$ )  | 23.2        | 25.4        |
| Align-Motifs ( $p = 0.6$ )  | 22.8        | 24.8        |

Table 4. **Validation results of different masking ratio  $p$  values for the Neural Motifs model under the PredCls setting.** Motifs is our trained Neural Motifs [53] model. Align-Motifs is the Neural Motifs model containing our proposed self-supervised relation alignment mechanism during training, where the  $p$  value in the bracket is the masking ratio used for the experiment.

### 6.2. Model Performance w.r.t. the Loss Weight

Again, on the Neural Motifs [53] model under the PredCls setting, we conduct ablation experiments with respect to the loss weight  $\lambda$  (in Equation 3 of the main paper). Table 5 below shows the validation mean-Recall results with different  $\lambda$  values, which are obtained from a single run on the same machine with the same random seed. The masking ratio  $p$  (in Equation 1 of the main paper) is fixed to 0.1 for all the experiments. As the results suggest,  $\lambda$  being 10 gives us the most effective validation results among the values we experimented. Based on this ablation, we set  $\lambda$  to be 10 for all other experiment settings.

| Method                           | mR@50       | mR@100      |
|----------------------------------|-------------|-------------|
| Motifs                           | 22.1        | 24.2        |
| Align-Motifs ( $\lambda = 0.1$ ) | 22.0        | 23.5        |
| Align-Motifs ( $\lambda = 1$ )   | 23.8        | 25.9        |
| Align-Motifs ( $\lambda = 10$ )  | <b>23.9</b> | <b>26.2</b> |
| Align-Motifs ( $\lambda = 50$ )  | 22.5        | 24.2        |
| Align-Motifs ( $\lambda = 100$ ) | 21.7        | 23.4        |

Table 5. **Validation results of different loss weight  $\lambda$  values for the Neural Motifs model under the PredCls setting.** Motifs is our trained Neural Motifs [53] model. Align-Motifs is the Neural Motifs model containing our proposed self-supervised relation alignment mechanism during training, where the  $\lambda$  value in the bracket is the loss weight used for the experiment.

## 7. Masking Illustration - Align-SGTR\*

Figure 4 illustrates the last layer of the mirrored relation predictor (mirrored structural predicate decoder) of Align-SGTR\* – the SGTR [25] model equipped with our proposed self-supervised relation alignment mechanism during training. The mirrored structural predicate decoder shares weights with the original one, except for the untied projection heads. Random masking is applied on the attention matrix of the cross-attention Transformer blocks. Mask is generated independently for every cross-attention Transformer block at each Transformer layer, while the alignment losses are only enforced at the last layer.

## 8. Per-Predicate R@100 Difference - SGTR

Figure 5 shows the per-predicate R@100 difference between Align-SGTR\* and SGTR\* (the SGTR [25] model trained by us). The predicates are sorted by their frequencies in descending order from left to right. The predicate order, and the head-body-tail partitions are from [26]. Results are averaged across 4 runs. As can be seen, our Align-SGTR\* is better than SGTR\* on 40 predicate labels (out of a total of 50), in many cases by a sizable margin.

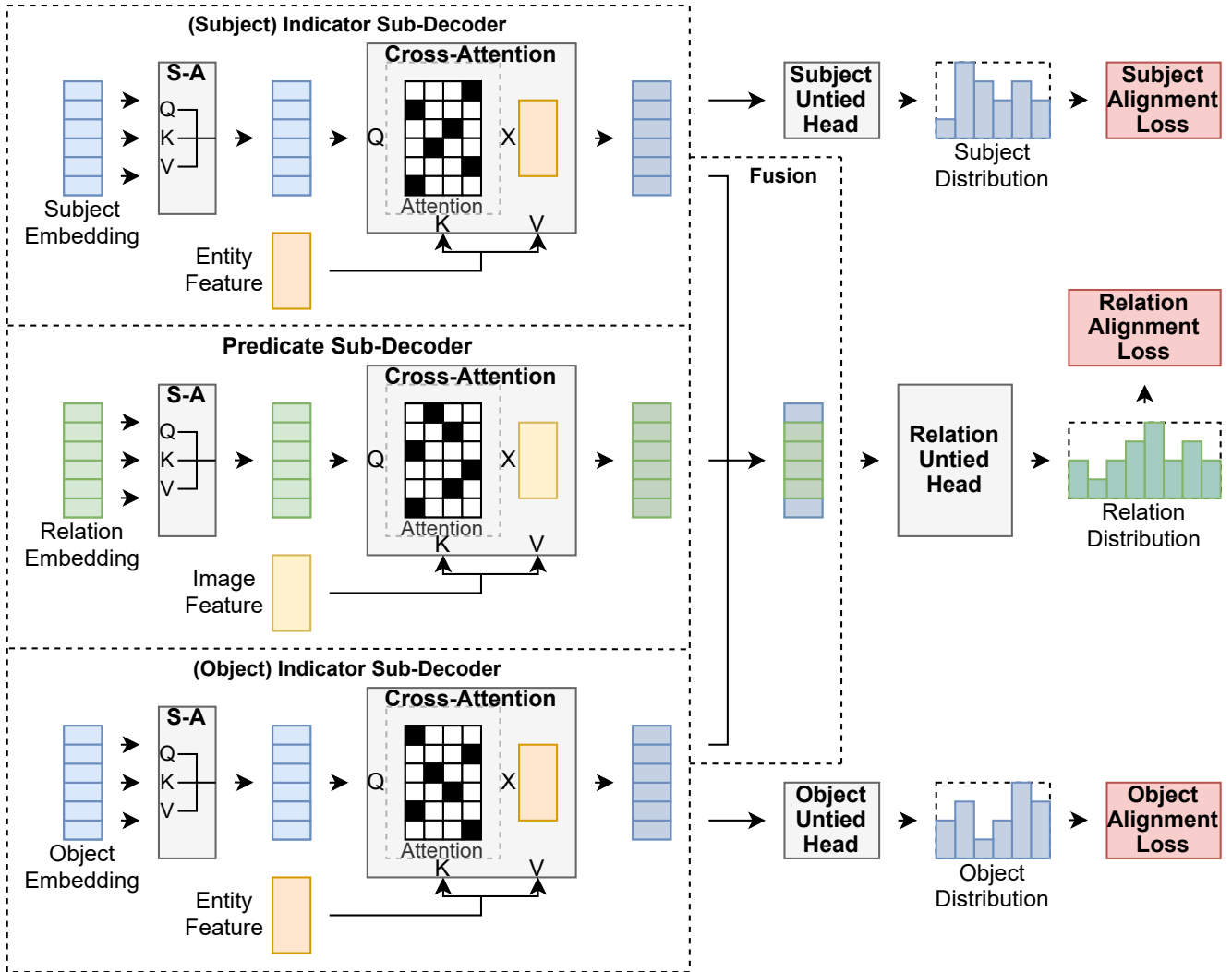


Figure 4. **Instantiation of our proposed self-supervised relation alignment mechanism under SGTR.** This figure illustrates the last Transformer layer of the mirrored structural predicate decoder, which shares weights with the original one, except for the untied projection heads (Relation/Subject/Object Untied Head). Random masking is applied on the attention matrix of the cross-attention Transformer blocks. Mask is generated independently for every cross-attention Transformer block at each Transformer layer, while the alignment losses are only enforced at the last layer. In the figure, S-A stands for a self-attention Transformer block. Q, K, and V are the query, key, and value of a Transformer block respectively.

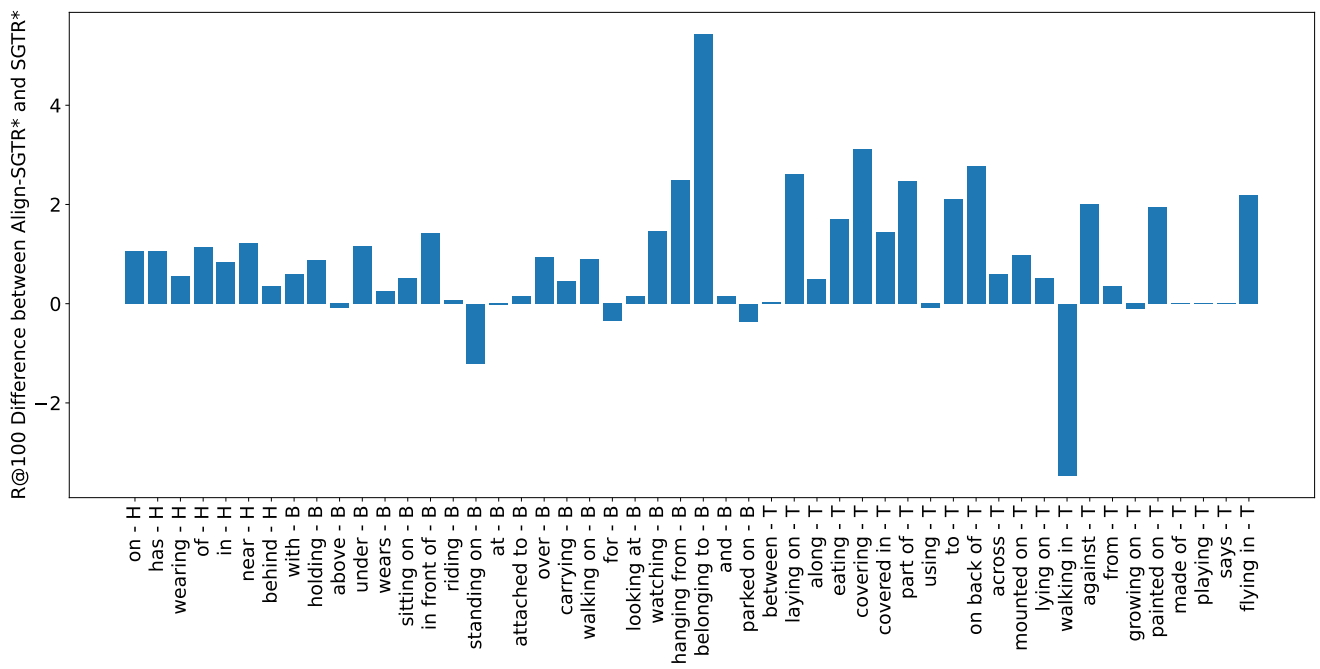


Figure 5. **Per-predicate R@100 difference between Align-SGTR\* and SGTR\***. The predicates are sorted by their frequencies in descending order from left to right. H, B, and T indicate the head, body, and tail partitions respectively. Results are averaged across 4 runs.