Appendix for
Composite Relationship Fields with Transformers for Scene Graph Generation

George Adaimi, David Mizrahi, Alexandre Alahi
EPFL, VITA
firstname.lastname@epfl.ch

A. Additional Implementation Details

A.1. Object Detection

We employ the one-stage anchor-free CenterNet [99] detection approach for the object detector. CenterNet has shown high performance on the widely-used object detection dataset, MSCOCO [42]. Given feature map F extracted by the backbone, the objection detection head outputs three feature maps: (1) a heatmap indicating the centers and categories of the objects, (2) a center offset to deal with the precision error caused by the output stride, and (3) the objects' width and height. To better understand how the ground-truth feature maps are generated, let the objects in a scene be represented by a category c and bounding box $b = [x_c, y_c, w, h]$ where x_c and y_c are the center locations and w and h are the width and height respectively. The object center heatmap $H^o \in [0,1]^{C_o \times \frac{H}{s} \times \frac{W}{s}}$, where C_o is the number of object categories, indicates the location of the objects in an image. Following Zhou *et al.* [99], the object center heatmap is generated by applying a Gaussian kernel at every downsampled object location based on the following equation:

$$H^o_{c,y,x} = e^{-\frac{\left(x - \lfloor \frac{x_c}{s} \rfloor\right)^2 + \left(y - \lfloor \frac{y_c}{s} \rfloor\right)^2}{2\sigma^2}}, \tag{1}$$

where σ changes with the size of the object. Since the output heatmap is downsampled, a discretization error occurs when remapping the predicted centers to their location in the input image. This affects the location of the bounding boxes, especially for small objects. Thus, an offset map $\theta \in \mathbb{R}^{2 \times \frac{H}{s} \times \frac{W}{s}}$ is built that outputs the precise offset lost due to downsampling:

$$\theta_{\lfloor \frac{y_c}{s} \rfloor, \lfloor \frac{x_c}{s} \rfloor} = \left(\frac{x_c}{s} - \lfloor \frac{x_c}{s} \rfloor, \frac{y_c}{s} - \lfloor \frac{y_c}{s} \rfloor \right). \tag{2}$$

The third output map $B \in \mathbb{R}^{2 \times \frac{H}{s} \times \frac{W}{s}}$ predicts the width and height of the objects at the downsampled center location $B_{\lfloor \frac{y_c}{s} \rfloor, \lfloor \frac{x_c}{s} \rfloor} = (w, h)$.

A.2. Relationship Detection

Our Composite Relationship Fields (CoRF) are introduced to represent the relationships between pairs of objects. The relationship detection head predicts an output map a^p for every relationship category p. For every location (i, j) in the output map and relationship p, CoRF is represented as follows: $a^p_{ij} = [c, x_s, y_s, x_o, y_o, s_s, s_o]^p_{ij}$. For example, assume a predicate r exists between the relationship subject with bounding box $b^s = [x^s_c, y^s_c, w^s, h^s]$ and the relationship object with bounding box $b^o = [x^o_c, y^o_c, w^o, h^o]$ where x_c and y_c are the center locations of the downscaled bounding boxes and w and h are their width and height respectively. For every location (i, j) in the region between the two centers (x^s_c, y^s_c) and (x^o_c, y^o_c) and for predicate r, a^r_{ij} is represented as follows:

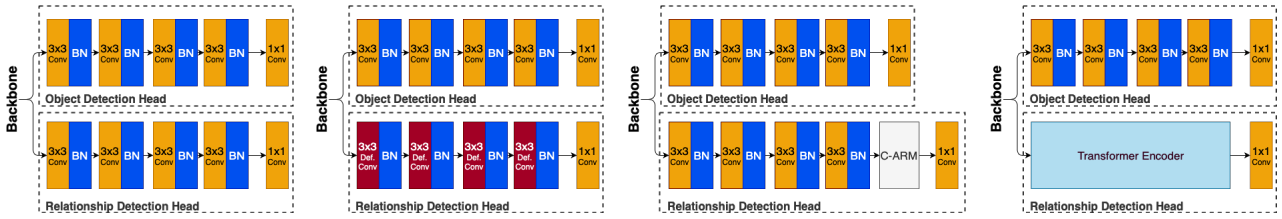
$$\begin{aligned} c = 1, \qquad & x_s = x^s_c - j, & & y_s = y^s_c - i, \\ & x_o = x^o_c - j, & & y_o = y^o_c - i, \\ & s_s = 0.1 \cdot \min(w^s, h^s) & & s_o = 0.1 \cdot \min(w^o, h^o), \end{aligned}$$

Figure 3 in the main paper shows the different components of the CoRF for the predicate *throwing*. The blue vectors represent the vectors $v_s = (x_s, y_s)$ pointing towards the relationship subject and the green vectors represent the vectors $v_o = (x_o, y_o)$ pointing towards the relationship object.

A.3. Implementation and Training Details

We report results with a convolutional backbone: ResNet-50 [20], and a Transformer backbone: Swin-S [48]. These backbones are pretrained on ImageNet [61] and modified to output a feature map of stride 16. For ResNet, we increase the feature map resolution to stride 16 by dilating the C5 stage, as in [1]. For Swin, we first upsample the last feature map of stage 4 to stride 16. Then, we project the last feature map of stage 3 to C channels (with $C = 768$ for Swin-S) and add it to the upsampled stage 4 feature map before applying one 3×3 convolution to the combined feature map, similar to [41].

All models are trained using AdamW [2] for 60 epochs with a learning rate of 10^{-4} for the heads and 10^{-5} for the



(a) CoRF (b) CoRF + Deform (c) CoRF + S.Deform (d) CoRF + T_R

Figure 1: **Illustration of the refinement heads.** Architecture of the different refinement heads used in the ablation study.

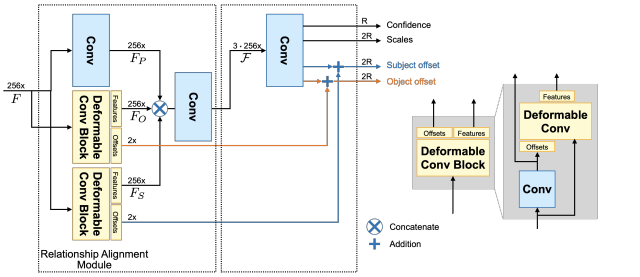


Figure 2: **Illustration of the Relationship Alignment Module (RAM).** RAM is made up of three parallel layers each responsible for a component in a relationship triplet <*subject, predicate, object*>. All convolution layers have a 1x1 kernel. The channel dimension is indicated on the connections, *e.g.*, 256x. R is the number of relationship categories.

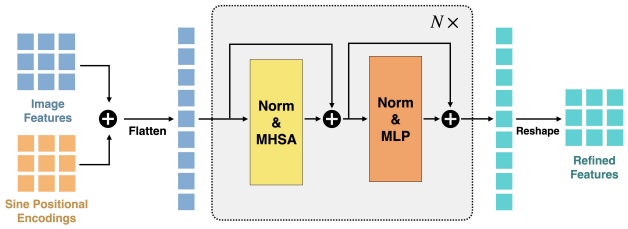


Figure 3: **Illustration of the Transformer Encoder.** The Transformer encoder of each refinement head consists of 6 Transformer blocks. Each Transformer block consists of a multi-head self-attention layer (MHSA) followed by a point-wise multilayer perceptron (MLP), with skip-connections and normalization layers. See Section 3.3 for more details.

backbone [4]. The learning rates are decreased by a factor of 10 at epochs 40 and 50. We train our models on 4 V100 GPUs with 10 images per GPU for a total batch size of 40. We apply standard data augmentations techniques of random scaling, cropping, and horizontal flipping, similar to FCSGG [47]. All models are trained and evaluated on images of resolution 512×512.

A.4. Refinement Heads

We study the effect of different refinement heads to verify that Transformers are suitable for the task of scene graph generation. The different refinement heads are illustrated in Fig. 1. The baseline model (CoRF) has four 3x3 convolution layers each separated by a batch normalization (BN) and a ReLU (not shown in the figure for compactness). We also study the performance of using deformable convolution layers [8] instead of traditional ones for the relationship detection head (CoRF + Deform). Deformable convolution layers were introduced to deal with the fixed receptive fields of traditional convolution layers. In essence, they allow the model to attend to different spatial locations in the scene, making it a suitable attention mechanism.

Since the relationship head needs to predict, at every location, the relationship between a subject and object, we introduce another refinement head that aligns the features of the subject, predicate, and object. The primary motivation behind our proposed refinement is that these three components are highly correlated. It is achieved by adding a *relationship alignment module* (RAM) after the four convolution layer in the relationship head (CoRF + S.Deform). This module consists mainly of *supervised deformable convolutions*. RAM enriches the feature vector responsible for relationship prediction using information from the related subjects and objects at every location. Essentially, RAM applies the following equation to output the refined feature map $\mathcal{F}$ at location x, y:

$$\begin{aligned}\mathcal{F}_{x,y} &= W \cdot (F_P \oplus F_S \oplus F_O) \\ &= W \cdot [\underbrace{(W_r \cdot F_{x,y})}_{\text{predicate}} \\ &\quad \oplus \underbrace{(W_s \cdot F_{x+S^x_{x,y}, y+S^y_{x,y}})}_{\text{subject}} \\ &\quad \oplus \underbrace{(W_o \cdot F_{x+O^x_{x,y}, y+O^y_{x,y}})}_{\text{object}}] \end{aligned} \tag{3}$$

where $F_{x,y}$ is the feature at location (x, y), $S^x_{x,y}$ and $S^y_{x,y}$ are the vector coordinates of the subject relative to location (x, y), and $O^x_{x,y}$ and $O^y_{x,y}$ are the vector coordinates of the objects relative to location (x, y). W_s, W_o, W_r are

the learned subject, object, and predicate transformation matrices to produce F_S, F_O, and F_P respectively. RAM is illustrated in Figure 2. Compared to deformable convolutions, the main modification in our implementation is that we supervise the kernel offsets by adding them directly to the CoRF of the relationship head. This forces the deformable layer to attend specifically to the location of the subject and object rather than learning where to attend to.

Lastly, we also study the performance when using only a Transformer encoder in the relationship head (CoRF + T_R). The Transformer encoder used in both CoRF + T_R and CoRF + T is illustrated in Figure 3.

B. Inference Speed

We report the inference speed of additional top-down SGG methods and compare them to our model (Table 1). All models are tested on the NVIDIA GeForce GTX 1080 Ti GPU. All our models maintain a real-time performance while improving performance compared to previous bottom-up approaches, which shows their efficiency. Thus, our method proves to be computationally efficient without compromising performance. As shown in the table, the inference speed remains almost the same when evaluating BGNN [36] under two different input sizes. This is expected for top-down methods since the main bottleneck is performing relationship prediction on a large number of pairs of detected objects, which is independent of the input size.

	Method	Backbone	Input Size	img/sec
Top-down	VCTree-TDE [67]	RNXt101-FPN	600×1000	1
	KERN [5]	VGG16	592×592	1
	MOTIFS-TDE [67]	RNXt101-FPN	600×1000	1
	MOTIFS-TDE [67]	RNXt-101-FPN	512×512	1.1
	GB-NET-β [90]	VGG16	592×592	2
	MOTIFS [67]	RNXt101-FPN	600×1000	2
	VTransE-TDE [67]	RNXt101-FPN	600×1000	2
	Graph R-CNN [84]	VGG16	800×1024	5
	BGNN [36]	RNXt-101-FPN	600×1000	1.4
	BGNN [36]	RNXt-101-FPN	512×512	1.8
Bottom-up	Px2Grp [55]	Hourglass-104	512×512	0.2
	FCSGG [47]	HRNet-W38	512×512	14
	FCSGG [47]	HRNet-W48	512×512	13
	FCSGG [47]	RN50-FPN$_{\times 2}$	512×512	25
	CoRF	RN50	512×512	30
	CoRF + T	RN50	512×512	22
	CoRF	Swin-S	512×512	20
	CoRF + T	Swin-S	512×512	15

Table 1: **Inference speed for different SGG methods.** All methods are tested on a NVIDIA GeForce GTX 1080 Ti GPU with batch size of 1.

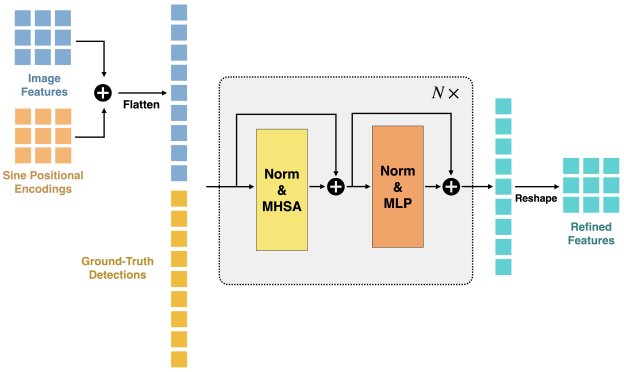


Figure 4: **Illustration of Ground-truth Detections Injection.** The Transformer encoder of each refinement head takes as input the concatenation of the flattened backbone features and ground-truth detection tokens when evaluating PredCls and SGCls.

C. Positional Encodings

As mentioned in the main paper, we study the effect of different positional encodings added to the input of the Transformer. Using learnable 1D or 2D positional encodings did not lead to any significant gain in performance over fixed sinusoidal + absolute position encodings (Table 2).

D. Ground-Truth Detection Injection

In PredCls and SGCls settings, the relationship head should only be evaluated for its ability to classify the relationship and both object and relationship, respectively. To compare with previous work (*e.g.* Px2Graph [55]), we need to provide our model with ground-truth object annotations when evaluating PredCls and SGCls. The annotations are provided as additional input tokens to the Transformer modules of both the detection and relationship head during training and testing following the same hyperparameters reported in the paper. The mechanism is visualized in Figure 4. For the SGCls protocol, a token is created for every ground-truth object that contains the object's location and bounding box size. For PredCls, this vector also contains an embedding indicating the category of the object. Each category has a different embedding that is learned while training. Positional encoding is also added to these tokens to encode their locations, similar to the image features. In addition to the described ground-truth tokens, the relationship head is passed all possible object pair vectors, similar to how top-down methods predict the relationship between all pairs of objects. This is achieved by creating a token for every object pair in the ground-truth annotations, which encodes the location of the corresponding objects. To clarify, for every pair of objects in the ground-truth annotations, a token encoding the locations of the pair of corresponding objects is passed

		PredCls	
	$AP_{0.5}$	R@50	ng-R@50
Learned 1D	20.9	43.9	56.3
Learned 2D	21.1	43.6	56.2
Fixed (Sine + Abs. Pos)	**21.9**	**44.4**	**56.8**

Table 2: **Performance using different positional encodings.** Fixed sinusoidal + absolute position encodings perform slightly better than learnable encodings for both object detection ($AP_{0.5}$) and relationship detection (PredCls). We do not observe any significant difference in performance between learnable 1D and learnable 2D positional encodings

.

to the relationship head. This is done for every object pair, whether there exists a ground-truth relationship or not. It allows our relationship head to focus mainly on classifying the relationship rather than also learning how to predict the object's location. These tokens are then concatenated with the backbone features before passing them to the Transformer modules(Figure 4).

E. Additional Visualizations

E.1. Attention Maps

We provide additional self-attention visualizations in Figure 5. We show visualizations of the last self-attention layer of both the relationship head's Transformer and the detection head's Transformer, with both the ResNet-50 backbone and the Swin-S backbone. The selected images are from the Visual Genome test set. The images selected for the Swin-S backbone are also used for the CoRF visualizations (Appendix E.2).

We observe that the attention maps differ between the heads. For the relationship head, the self-attention heads attend to the cell's surroundings and far-away objects that are likely to be part of a relationship. The model is able to learn, with the supervision of our CoRF connecting different objects, that attending specifically to objects in the scene improves relationship prediction. The attention maps for the object detection head are much less interpretable, as some self-attention heads attend to the corners and center of the image while others attend to locations throughout the image. Such differences indicate that the Transformer of the object detection head needs to attend to different details compared to the Transformer of the relationship head, motivating the need for two separate encoders for each of these tasks.

We note that this difference in the attention maps is not surprising; as for the relationship head, each cell of the feature map has to identify surrounding objects and the relationships between them, while for the object detection head, each cell only has to detect if it is at the center of an object and, if so, the object's class, height, and width.

E.2. Composite Relationship Fields

We show in Figures 6, 7, 8 qualitative results for different sample images in the Visual Genome [31] test set. These images were extracted by following the SGDet protocol [81]. The model used is CoRF + T, with Swin-S as the backbone. We report the ground-truth object detection, ground-truth scene graph, predicted object detections, and the composite relationship fields for different predicates. In Figure 6d, the green vectors with the red boxes point to the subject, and the blue vectors with the green boxes point towards the object of the relationship triplet <*subject, predicate, object*>.

In Figure 6, we can observe that our detector is able to detect additional objects compared to the ground-truth annotations, such as the building, lights, and windows. Moreover, although the ground-truth scene graph contains relationships between three objects in the scene, our method was able to extract more relationships shown by the relationship fields for every predicate. For example, our method is able to identify that the 'bus' is 'parked on' the 'street' (6d). This shows that our method perceives the image globally and generalizes to other predicates. Another interesting observation is that the relationship field for predicates 'has' and 'of' are similar but with their object and subject vectors swapped. This indicates that our model is able to deduce that a relationship triplet <*object A, has, object B*> is equivalent to <*object B, of, object A*>.

Similarly, Figures 7 and 8 demonstrate the generalizability of our model, *i.e.* we are able to extract many more relationships than the ones found in the ground-truth. These figures also highlight an issue in the Visual Genome dataset: object and predicate ambiguity. For example, in Figure 7b, our object detector detected the three people in the scene as 'people', 'person', or 'guy' as opposed to the ground-truth 'man', 'girl', and 'skier'. Although both labels can be correct, our model is penalized for such predictions. Another common issue in the Visual Genome dataset is predicate ambiguity. It occurs because the training dataset has similar predicates, such as 'wears' and 'wearing'.

F. Additional Results

As mentioned in Section 5, we report the detailed performance of previous methods. Tables 3, 4, and 5 show the Recall performance at K=20, K=50, and K=100, respectively for both graph and no-graph constraint. These tables are divided into top-down and bottom-up approaches. Furthermore, the top-down methods are further divided into methods that use external knowledge (top-part) such as linguistic priors, knowledge bases, and statistical correlations between objects, and methods that only use visual information (bottom-part). As observed, our method outperforms previ-

ous bottom-up approaches, especially at K=20 and K=50, indicating that correct relationships are ranked higher than incorrect ones. Our method also achieves competitive results compared to top-down visual-only methods (CoRF+T$^+$). Other methods use external knowledge that increases performance and especially helps deal with relationships with few instances. We focus in our work on improving visual-only bottom-up approaches using Composite Relationship Fields and our Transformer-based refinement head.

Similarly, Tables 8, 6 and 7 show the the mean-Recall at K=20, K=50, and K=100, respectively. We observe that our method is still able to outperform previous bottom-up methods indicating that our model can generalize and deal with the long-tail distribution of Visual Genome [31]. It is important to note that the top-down methods mentioned in the tables apply techniques such as sampling strategies, external knowledge, etc., to specifically improve mean recall. These strategies aid in distinguishing between relationships that are close visually, such as *laying on* and *lying on*. These relationships are challenging for methods relying on visual-only input, similar to our method and FCSGG [47].

We also report the zero-shot performance at K=50 and K=100 (Table 9 and 10). This also verifies the benefits of CoRF and the refinement head in helping the model generalize to unseen relationships. Previous top-down methods reporting zero-shot performance apply a debiasing technique (TDE [67]) to improve their method's zero-shot metric. Our method only relies on visual information.

G. Limitations

Our method is highly limited by the performance of the detection task. For a fair comparison with previous work, we have used the same popular method [99] which is not necessarily competitive with top-down methods. Hence, future work should address this limitation by potentially leveraging other detection datasets to improve the performance of bottom-up detectors. Moreover, the Visual Genome dataset suffers from many issues, such as missing or similar annotations and highly biased distribution, which can negatively affect training. The effect also differs between top-down and bottom-up methods. In addition, our current performances should be further improved if our method is to be used for safety-critical applications such as autonomous navigation.

Societal impact. Any SGG method can be used in surveillance to visually understand an environment. However, our method does not use or store any identifying information.

References

[1] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2359–2367, 2017.

[2] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

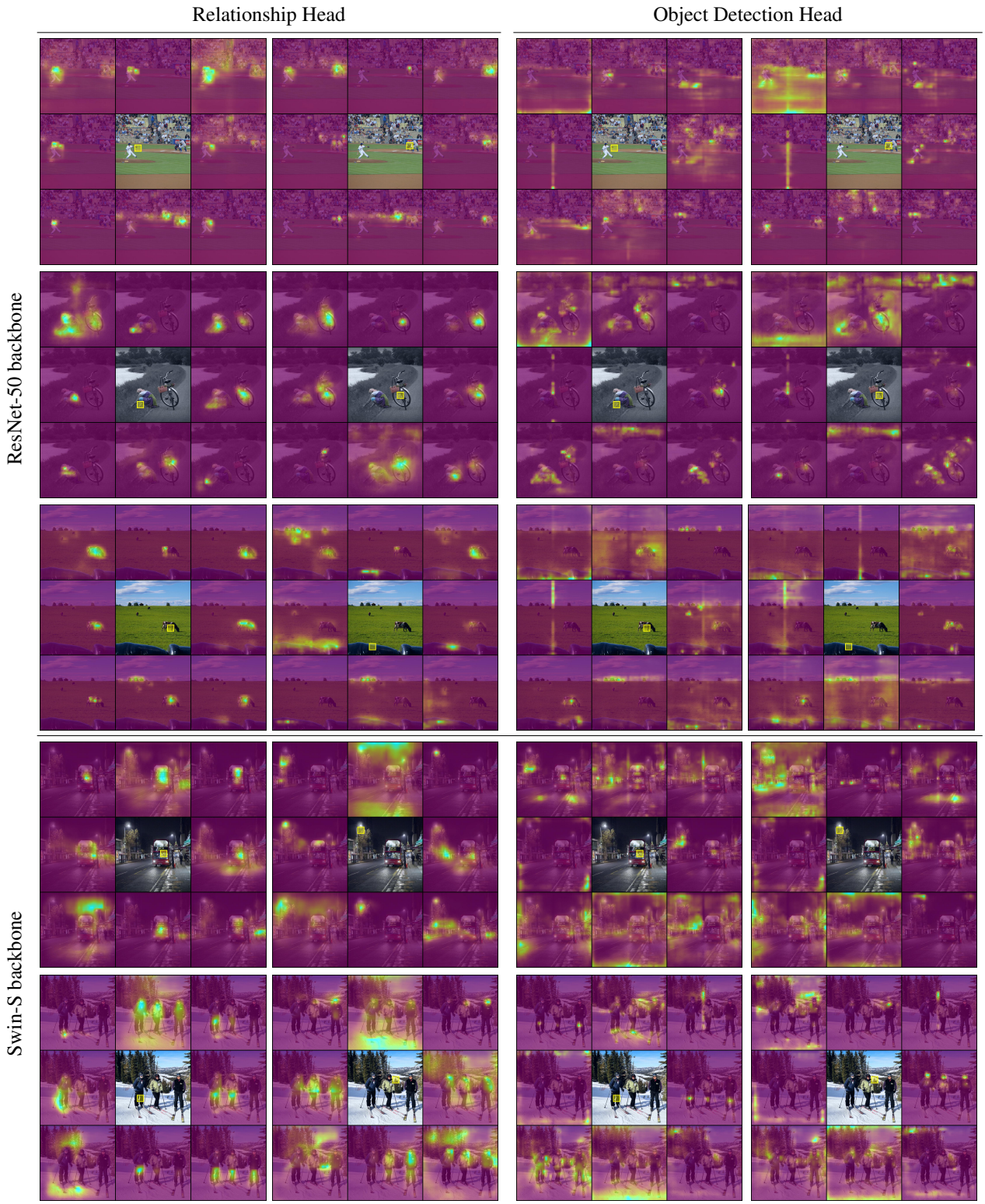
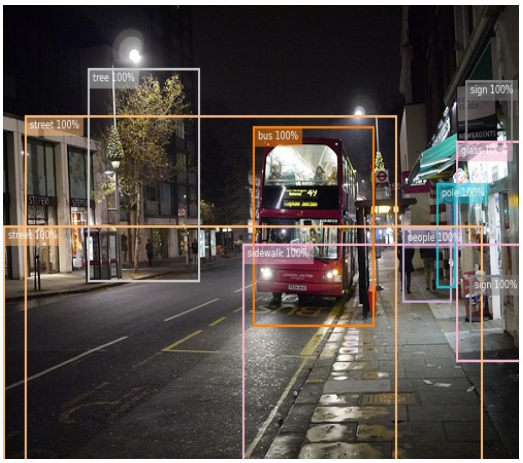
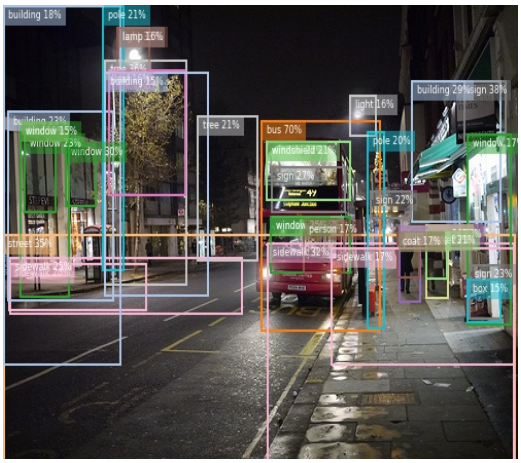


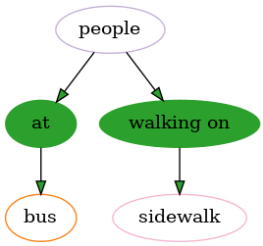
Figure 5: **Attention maps from the relationship head and object detection head.** For a given reference point, the attention maps from all heads of the last self-attention layer of the Transformer are shown. We show attention maps for both backbones: ResNet-50 and Swin-S, from both the relationship head's Transformer and the object detection head's Transformer.



(a) GT detections



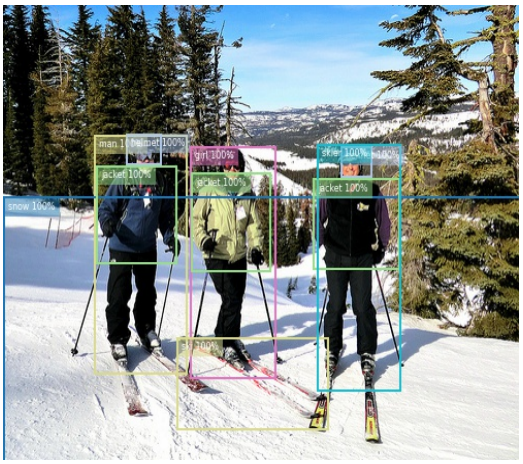
(b) Predicted detections



(c) GT Scene Graph

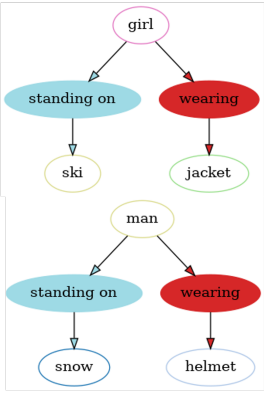
(d) Composite Relationship Fields for different predicates

Figure 6: **Qualitative results for image *2343440.jpg*.** Our model is able to detect the relationship 'parked on' between the bus and the street. It also predicted that the bus 'has' windows. It was even able to detect the relationship 'wearing' between the pedestrians and their pants. GT= Ground-Truth. (Images are enlarged for better viewing)



(a) GT detections

(b) Predicted detections

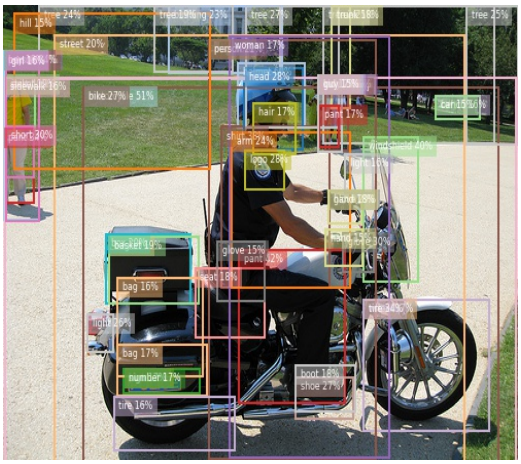


(c) GT Scene Graph

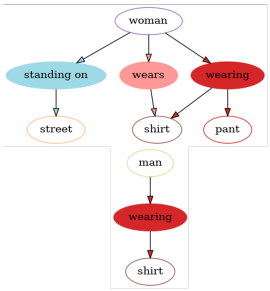
(d) Composite Relationship Fields for different predicates

Figure 7: **Qualitative results for image *2334989.jpg*.** Our model is able to analyze the scene and detect more relationships between objects such as 'holding', 'has', and 'behind'. GT= Ground-Truth. (Images are enlarged for better viewing)

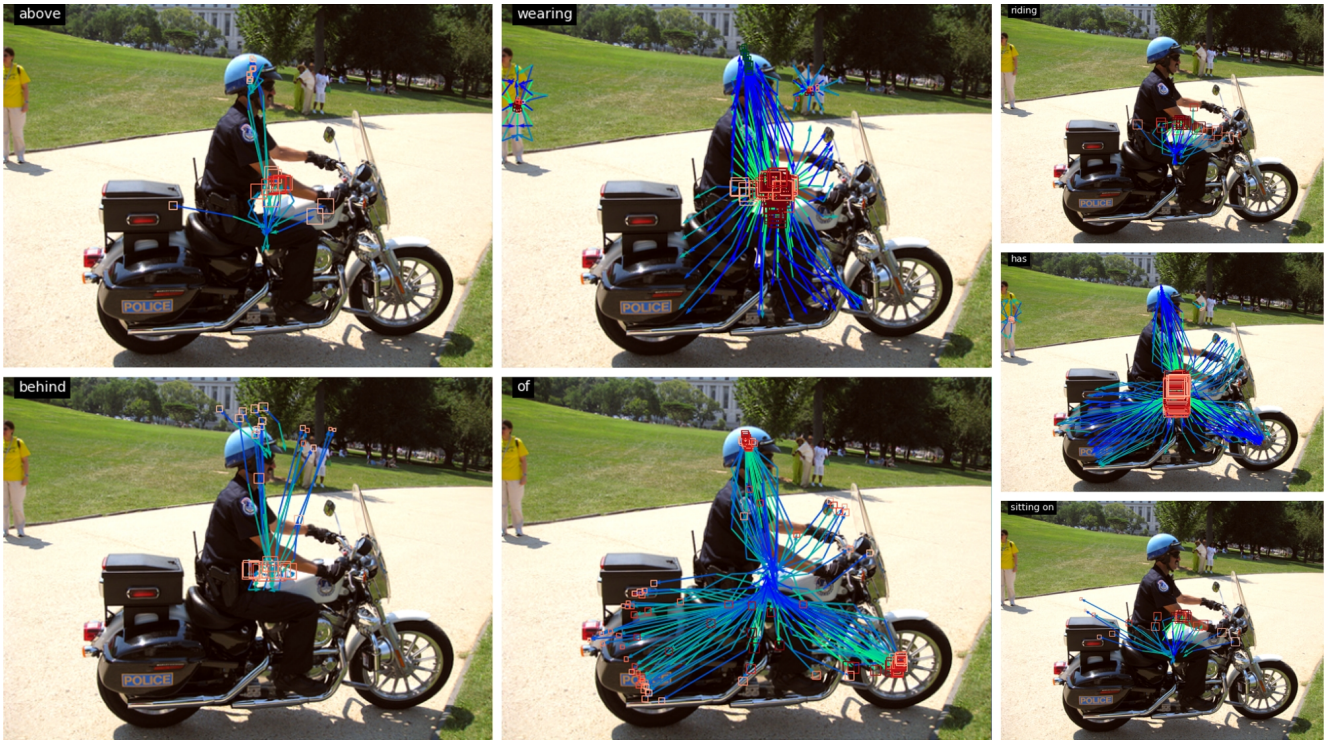
(a) GT detections



(b) Predicted detections



(c) GT Scene Graph



(d) Composite Relationship Fields for different predicates

Figure 8: **Qualitative results for image *2343342.jpg*.** Our model is able to analyze the scene and detect more relationships between objects such as 'above', 'riding', 'sitting on', and 'behind'. GT= Ground-Truth. (Images are enlarged for better viewing)

	Method	Backbone	$AP_{0.5}$	PredCls		SGCls		SGDet	
				R@20	ng-R@20	R@20	ng-R@20	R@20	ng-R@20
Top-down	VCTree [68]	VGG16 [64]	–	60.1	–	35.2	–	22.0	–
	GPS-Net [43]	VGG16 [64]	–	67.6	–	41.8	–	22.3	–
	RelDN [96]	VGG16 [64]	25.5	66.9	–	36.1	–	21.1	–
	MotifsTDE [67]	RNXt101-FPN	28.1	33.6	–	21.7	–	12.4	–
	MotifNet [93]	VGG16 [64]	20.0	58.5	–	32.9	–	21.4	–
	NLS [98]	RNXt101-FPN	–	58.7	–	36.5	–	24.6	–
	CISC [77]	VGG16 [64]	–	42.1	–	23.3	–	7.7	–
	LinkNet [78]	VGG16 [64]	–	61.8	–	38.3	–	22.3	–
	Seq2Seq [50]	VGG16 [64]	–	60.3	–	34.5	–	22.1	–
Bottom-up	Px2Graph* [55]	Hourglass-104 [56]	–	*47.9*	–	*18.2*	–	*6.5*	–
	FCSGG [47]	HRNet-W32 [74]	21.6	27.6	32.2	12.3	13.5	11.0	12.4
	FCSGG [47]	HRNet-W48 [74]	25.0	24.2	28.1	13.6	14.2	11.5	12.7
	FCSGG [47]	RN50-$FPN_{\times 2}$	23.0	28.0	31.6	13.9	14.8	11.4	12.2
	CoRF [Ours]	RN50	19.6	36.0	40.6	13.6	13.7	11.6	13.9
	CoRF + T [Ours]	RN50	21.9	38.2	43.4	15.9	16.0	13.3	15.9
	CoRF [Ours]	Swin-S	23.8	38.5	43.5	16.1	16.4	14.6	17.4
	CoRF + T [Ours]	Swin-S	24.7	39.3	44.9	17.3	17.7	15.3	18.0
	CoRF + T^+ [Ours]	RN50	21.9	56.7	67.9	27.7	32.9	13.3	15.9
	CoRF + T^+ [Ours]	Swin-S	24.7	**56.5**	**67.7**	**29.6**	**35.2**	**15.3**	**18.0**

Table 3: **Recall@20 for graph (R@20) and no-graph constraint (ng-R@20) on Visual Genome [31].** $\star$ indicates that [55] trained a different model for each metric. RN50 = ResNet50[20]. RNXt101 = ResNeXt-101 [80]. FPN = Feature Pyramid Network [41]. $FPN_{\times 2}$ indicates that a separate FPN is used for detection and relationship. Our proposed CoRF has convolutions in both heads (Sec. 6), while CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls to compare with Px2Graph.

	Method	Backbone	$AP_{0.5}$	PredCls		SGCls		SGDet	
				R@50	ng-R@50	R@50	ng-R@50	R@50	ng-R@50
Top-down	VRD [49]	VGG16 [64]	–	27.9	–	11.8	–	0.3	–
	VCTree [68]	VGG16 [64]	–	66.4	–	38.1	–	27.9	–
	KERN [5]	VGG16 [64]	–	65.8	81.9	36.7	45.9	27.1	30.9
	GPS-Net [43]	VGG16 [64]	–	69.7	–	42.3	–	28.9	–
	RelDN [96]	VGG16 [64]	25.5	68.4	93.8	36.8	48.9	28.3	30.4
	MotifsTDE [67]	RNXt101-FPN	28.1	46.2	–	27.7	–	16.9	–
	MotifNet [93]	VGG16 [64]	20.0	65.2	–	35.8	–	27.2	–
	GB-NET-β [90]	VGG16 [64]	–	66.6	83.5	37.3	46.9	26.3	29.3
	PI-SG [21]	–	–	65.1	80.8	36.5	45.5	–	–
	NLS [98]	RNXt101-FPN	–	65.6	–	40.0	–	31.8	–
Bottom-up	IMP [81]	VGG16 [64]	–	44.8	–	21.7	–	3.4	–
	Graph R-CNN [84]	VGG16 [64]	23.0	54.2	–	29.6	–	11.4	–
	VRF [12]	RN50	–	56.7	–	23.7	–	13.2	–
	CISC [77]	VGG16 [64]	–	53.2	–	27.8	–	11.4	–
	LinkNet [78]	VGG16 [64]	–	67.0	–	41	–	27.4	–
	Seq2Seq [50]	VGG16 [64]	–	66.4	83.6	38.3	46.9	30.9	30.9
	BGNN [36]	RNXt101-FPN	–	59.2	–	37.4	–	31.0	–
	Px2Graph* [55]	Hourglass-104 [56]	–	*54.1*	*68.0*	*21.8*	*26.5*	*8.1*	*9.7*
	FCSGG [47]	HRNet-W48 [74]	25.0	31.0	40.3	17.1	19.6	15.5	18.3
	FCSGG [47]	HRNet-W32 [74]	21.6	34.9	46.3	15.5	19.3	15.1	18.2
	FCSGG [47]	RN50-$FPN_{\times 2}$	23.0	35.8	44.7	17.7	20.6	15.7	18.0
	CoRF [Ours]	RN50	19.6	42.3	53.9	14.8	18.3	14.5	17.6
	CoRF + T [Ours]	RN50	21.9	44.4	56.8	17.2	21.3	16.5	20.2
	CoRF [Ours]	Swin-S	23.8	44.8	56.9	17.5	21.6	17.9	22.0
	CoRF + T [Ours]	Swin-S	24.7	45.4	58.1	18.7	23.4	18.6	22.9
	CoRF + T^+ [Ours]	RN50	21.9	60.5	78.8	28.6	36.1	16.5	20.2
	CoRF + T^+ [Ours]	Swin-S	24.7	**60.2**	**78.5**	**30.5**	**38.8**	**18.6**	**22.9**

Table 4: **Recall@50 for graph (R@50) and no-graph constraint (ng-R@50) on Visual Genome [31].** $\star$ indicates that [55] trained a different model for each metric. RN50 = ResNet50[20]. RNXt101 = ResNeXt-101 [80]. FPN = Feature Pyramid Network [41]. $FPN_{\times 2}$ indicates that a separate FPN is used for detection and relationship. Our proposed CoRF has convolutions in both heads (Sec. 6), while CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls to compare with Px2Graph.

	Method	Backbone	$AP_{0.5}$	PredCls		SGCls		SGDet	
				R@100	ng-R@100	R@100	ng-R@100	R@100	ng-R@100
Top-down	VRD [49]	VGG16 [64]	–	35.0	–	14.1	–	0.5	–
	VCTree [68]	VGG16 [64]	–	68.1	–	38.8	–	31.3	–
	KERN [5]	VGG16 [64]	–	67.6	88.9	37.4	49.0	29.8	35.8
	GPS-Net [43]	VGG16 [64]	–	69.7	–	42.3	–	33.2	–
	RelDN [96]	VGG16 [64]	25.5	68.4	97.8	36.8	50.8	32.7	36.7
	MotifsTDE [67]	RNXt101-FPN	28.1	51.4	–	29.9	–	20.3	–
	MotifNet [93]	VGG16 [64]	20.0	67.1	–	36.5	–	30.3	–
	GB-NET-β [90]	VGG16 [64]	–	68.2	90.3	38.0	50.3	29.9	35.0
	PI-SG [21]	–	–	66.9	88.2	38.8	50.8	–	–
	NLS [98]	RNXt101-FPN	–	67.4	–	40.8	–	36.3	–
Bottom-up	IMP [81]	VGG16 [64]	–	53.8	–	24.4	–	4.2	–
	Graph R-CNN [84]	VGG16 [64]	23.0	59.1	–	31.6	–	13.7	–
	VRF [12]	VGG16 [64]	–	57.2	–	24.7	–	13.5	–
	CISC [77]	VGG16 [64]	–	57.9	–	29.5	–	13.9	–
	LinkNet [78]	VGG16 [64]	–	68.5	–	41.7	–	30.1	–
	BGNN [36]	RNXt101-FPN	–	61.3	–	38.5	–	35.8	–
	Seq2Seq [50]	VGG16 [64]	–	68.5	90.8	39.0	51.2	34.4	37.0
	Px2Graph* [55]	Hourglass-104 [56]	–	*55.4*	*75.2*	*22.6*	*30.0*	*8.2*	*11.3*
	FCSGG [47]	HRNet-W32 [74]	21.6	38.5	56.6	17.2	23.6	18.1	23.0
	FCSGG [47]	HRNet-W48 [74]	25.0	34.6	50.0	18.8	24.0	18.4	23.0
	FCSGG [47]	RN50-$FPN_{\times 2}$	23.0	40.2	54.8	19.6	25.0	19.0	22.8
	CoRF [Ours]	RN50	19.6	44.1	62.4	14.9	20.6	15.9	19.9
	CoRF + T [Ours]	RN50	21.9	46.0	65.1	17.4	23.6	18.1	22.6
	CoRF [Ours]	Swin-S	23.8	46.7	65.2	17.7	23.9	19.4	24.5
	CoRF + T [Ours]	Swin-S	24.7	47.1	66.3	18.9	25.8	20.0	25.4
	CoRF + T^+ [Ours]	RN50	21.9	61.1	83.5	28.7	37.3	18.1	22.6
	CoRF + T^+ [Ours]	Swin-S	24.7	**60.8**	**83.2**	**30.6**	**40.0**	**20.0**	**25.4**

Table 5: **Recall@100 for graph (R@100) and no-graph constraint (ng-R@100) on Visual Genome [31].** $\star$ indicates that [55] trained a different model for each metric. RN50 = ResNet50[20]. RNXt101 = ResNeXt-101 [80]. FPN = Feature Pyramid Network [41]. $FPN_{\times 2}$ indicates that a separate FPN is used for detection and relationship. Our proposed CoRF has convolutions in both heads (Sec. 6), while CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls to compare with Px2Graph.

	Method	Backbone	PredCls mR/ng-mR	SGCls mR/ng-mR	SGDet mR/ng-mR
Top-down	KERN [5]	VGG16	17.7/–	9.4/–	6.4/–
	VCTree [68]	VGG16	17.9/–	10.1/–	6.9/–
	GB-NET-β [90]	VGG16	22.1/–	12.7/–	7.1/–
	NLS [98]	RNXt101-FPN	17.7/–	10.4/–	7.3/–
	ResCAGCN [83]	VGG16	20.2/–	11.9/–	7.7/–
	VCTree-Seg [26]	RNXt101-FPN	19.2/–	11.6/–	8.1/–
	MOTIFS-TDE [67]	RNXt101-FPN	25.5/–	13.1/–	8.2/–
	VCTree-TDE-EB [65]	RNXt101-FPN	26.7/–	18.2/–	9.7/–
	Seq2Seq [50]	VGG16	26.1/–	14.7/–	9.6/–
	BGNN [36]	RNXt101-FPN	30.4/–	14.3/–	10.7/–
	BA-SGG [19]	RNXt101-FPN	31.9/–	18.5/ –	14.8/–
Bottom-up	FCSGG	HRNet-W32	5.5/9.7	2.5/4.4	2.4/3.6
	FCSGG	HRNet-W48	5.2/9.5	2.9/6.3	2.6/4.7
	FCSGG	RN50-FPN$_{\times 2}$	5.7/11.3	2.9/6.0	3.2/5.7
	CoRF	RN50	8.1/17.0	2.7/5.4	2.7/5.8
	CoRF + T	RN50	9.5/20.0	3.4/6.8	3.5/7.6
	CoRF	Swin-S	9.3/19.2	3.3/6.9	3.5/7.9
	CoRF + T	Swin-S	10.1/21.7	3.9/8.3	3.9/9.2
	CoRF + T$^+$	RN50	13.5/34.9	5.0/11.7	3.5/7.6
	CoRF + T$^+$	Swin-S	**14.2/35.6**	**5.6/13.3**	**3.9/9.2**

Table 6: **Mean recall performance (@50).** We compare mean recall@50 for both graph (mR) and no-graph (ng-mR) constraint on Visual Genome [31]. CoRF has convolutions in both heads (Sec. 6), CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls

	Method	Backbone	PredCls mR/ng-mR	SGCls mR/ng-mR	SGDet mR/ng-mR
Top-down	KERN [5]	VGG16	19.4/–	10.0/–	7.3/–
	VCTree [68]	VGG16	19.4/–	11.8/–	8.0/–
	GB-NET-β [90]	VGG16	24.0/–	13.4/–	8.5/–
	NLS [98]	RNXt101-FPN	19.5/–	11.1/–	8.7/–
	ResCAGCN [83]	VGG16	22.0/–	12.8/–	8.9/–
	VCTree-Seg [26]	RNXt101-FPN	21.1/–	12.3/–	9.0/–
	GPS-Net [43]	VGG16	22.8/–	12.6/–	9.8/–
	MOTIFS-TDE [67]	RNXt101-FPN	29.1/–	14.9/–	9.8/–
	VCTree-TDE-EB [65]	RNXt101-FPN	30.0/–	20.5/–	11.6/–
	Seq2Seq [50]	VGG16	30.5/–	16.2/–	12.1/–
	BGNN [36]	RNXt101-FPN	32.9/–	16.5/–	12.6/–
	BA-SGG [19]	RNXt101-FPN	34.2/–	19.4/ –	17.1/–
Bottom-up	FCSGG	HRNet-W32	6.3/13.6	2.8/6.2	2.9/4.9
	FCSGG	HRNet-W48	6.1/14.7	3.4/9.4	3.1/6.9
	FCSGG	RN50-FPN$_{\times 2}$	6.7/16.6	3.3/8.3	3.3/6.8
	CoRF	RN50	9.1/24.5	2.8/7.2	3.1/7.2
	CoRF + T	RN50	10.5/28.1	3.5/8.8	4.1/9.3
	CoRF	Swin-S	10.4/27.5	3.4/9.1	4.0/9.7
	CoRF + T	Swin-S	11.3/29.9	4.0/10.3	4.5/10.9
	CoRF + T$^+$	RN50	14.4/46.1	5.1/13.4	4.1/9.3
	CoRF + T$^+$	Swin-S	**14.9/46.6**	**5.1/15.4**	**4.5/10.9**

Table 7: **Mean recall performance (@100).** We compare mean recall@100 for both graph (mR) and no-graph (ng-mR) constraint on Visual Genome [31]. CoRF has convolutions in both heads (Sec. 6), CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls

	Method	Backbone	PredCls mR/ng-m	SGCls mR/ng-m	SGDet mR/ng-mR
Top-down	VCTree [68]	VGG16	14.0/–	8.2/–	5.2/–
	NLS [98]	RNXt101-FPN	13.3/–	8.3/–	5.3/–
	MOTIFS-TDE [67]	RNXt101-FPN	18.5/–	9.8/–	5.8/–
	VCTree-Seg [26]	RNXt101-FPN	15.0/–	9.3/–	6.3/–
	VCTree-TDE-EB [65]	RNXt101-FPN	19.9/–	13.9/–	7.1/–
	Seq2Seq [50]	VGG16	21.3/–	11.9/–	7.5/–
	BA-SGG [19]	RNXt101-FPN	26.7/–	15.7/ –	11.4/–
Bottom-up	FCSGG	HRNet-W32	4.0/5.4	1.9/2.7	1.7/2.2
	FCSGG	HRNet-W48	3.7/5.2	2.2/3.5	1.8/2.7
	FCSGG	RN50-FPN$_{\times 2}$	4.2/6.5	2.2/3.6	1.9/3.0
	CoRF	RN50	6.0/9.5	2.3/3.3	1.9/3.8
	CoRF + T	RN50	7.2/11.4	2.9/4.1	2.5/5.2
	CoRF	Swin-S	7.0/11.1	2.8/4.1	2.5/5.4
	CoRF + T	Swin-S	7.8/12.5	3.3/5.0	2.9/6.3
	CoRF + T$^+$	RN50	11.2/21.3	4.6/8.8	2.5/5.2
	CoRF + T$^+$	Swin-S	**11.9/22.2**	**5.2/9.9**	**2.9/6.03**

Table 8: **Mean recall performance(@20).** We compare mean recall@20 for both graph (mR) and no-graph (ng-mR) constraint on Visual Genome [31]. CoRF has convolutions in both heads (Sec. 6), CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls

	Method	Backbone	PredCls zsR/ng-zsR	SGCls zsR/ng-zsR	SGDet zsR/ng-zsR
Top-down	VTransE-TDE [67]	RNXt101-FPN	13.3/–	2.9/–	2.0/–
	Motifs-TDE [67]	RNXt101-FPN	14.4/–	3.4/–	2.3/–
	VCTree-TDE [67]	RNXt101-FPN	14.3/–	3.2/–	2.6/–
	VCTree-TDE-EB [65]	RNXt101-FPN	15.1/–	6.4/–	2.7/–
Bottom-up	FCSGG	RN50-FPN$_{\times 2}$	8.2/11.7	1.3/2.4	0.8/1.0
	FCSGG	HRNet-W32	8.3/12.9	1.0/2.3	0.6 /1.2
	FCSGG	HRNet-W48	8.6/12.8	1.7/ 2.9	1.0/1.8
	CoRF	RN50	10.5/16.3	1.5/3.2	0.4/1.1
	CoRF + T	RN50	11.6/18.2	1.8/4.0	0.8/1.4
	CoRF	Swin-S	11.1/18.0	1.9/3.5	1.1/2.2
	CoRF + T	Swin-S	11.3/18.8	1.9/3.8	1.2/2.6
	CoRF + T$^+$	RN50	14.8/28.7	1.9/4.5	0.8/1.4
	CoRF + T$^+$	Swin-S	**14.5/28.5**	**2.5/5.8**	**1.2/2.6**

Table 9: **Zero-shot performance (@50).** We compare zero-shot recall@50 for both graph (zR) and no-graph (ng-zR) constraint on Visual Genome [31]. CoRF has convolutions in both heads (Sec. 6), CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls

	Method	Backbone	PredCls zsR/ng-zsR	SGCls zsR/ng-zsR	SGDet zsR/ng-zsR
Top-down	VCTree-Seg [26]	RNXt101-FPN	10.6/–	2.5/–	1.5/–
	VTransE-TDE [67]	RNXt101-FPN	17.6/–	3.8/–	2.7/–
	Motifs-TDE [67]	RNXt101-FPN	18.2/–	4.5/–	2.9/–
	VCTree-TDE [67]	RNXt101-FPN	17.6/–	4.0/–	3.2/–
Bottom-up	FCSGG	RN50-FPN$_{\times 2}$	8.2/11.7	1.3/2.4	0.8/1.0
	FCSGG	HRNet-W32	8.3/12.9	1.0/2.3	0.6 /1.2
	FCSGG	HRNet-W48	8.6/12.8	1.7/ 2.9	1.0/1.8
	CoRF	RN50	12.2/24.9	1.5/4.2	0.7/1.6
	CoRF + T	RN50	13.3/26.9	1.9/5.2	1.1/2.2
	CoRF	Swin-S	12.7/27.0	1.9/4.6	1.4/3.2
	CoRF + T	Swin-S	12.9/28.1	2.0/5.2	1.6/3.5
	CoRF + T$^+$	RN50	16.0/39.0	2.0/5.7	1.1/2.2
	CoRF + T$^+$	Swin-S	**15.6/38.0**	**2.6/7.2**	**1.6/3.5**

Table 10: **Zero-shot performance (@100).** We compare zero-shot recall@100 for both graph (zR) and no-graph (ng-zR) constraint on Visual Genome [31]. CoRF has convolutions in both heads (Sec. 6), CoRF + T has a Transformer encoder in both heads. $^+$: GT detections given to Transformer in PredCls and SGCls