Navigating the Unseen: Zero-shot Scene Graph Generation via Capsule-Based Equivariant Features

Supplementary Material

In this supplementary material, we present the following contents for the proposed CAPSGG framework, which effectively models equivariant features for zero-shot SGG: (1) a formulated definition of zero-shot SGG; (2) additional ablation studies examining the Capsule Feature Encoder (CFE) and the influence of hyperparameters m^+ and m^-; (3) qualitative visualizations; (4) a comparison of model size and speed with existing methods; and (5) a discussion of the limitations of the proposed framework. The code will be available upon publication.

6. Zero-Shot SGG Formulation

Our study addresses an advanced variant of Scene Graph Generation (SGG), referred to as zero-shot SGG, as defined in Sec. 3.1. We aim to develop a predicate prediction model that extracts knowledge from seen triplets categories, *i.e.* $t_{ij} = (o_i, p_k, o_j)$, where $o_{i,j} \in \mathcal{C}_o$ and $p_k \in \mathcal{C}_p$, $t_{ij} \in \mathcal{T}_s$. Here, $\mathcal{C}_o$ and $\mathcal{C}_p$ denote the categories of objects and predicates, respectively, $\mathcal{T}_s$ represents the set of all seen triplets, which is utilized to predict any triplet that does not appear (unseen) during the training stage, *i.e.* $t_{nm} = (o_n, p_k, o_m)$, $t_{nm} \notin \mathcal{T}_s$. Now, we can represent the probability of two objects, o_i and o_j with bounding boxes b_i and b_j, having the relationship of the predicate label p_k as shown in Eq.(10):

$$P(p_k|o_i,b_i,o_j,b_j,I)_{(o_i,p_k,o_j)\notin\mathcal{T}_s} \propto softmax\left(\mathcal{F}(o_i,b_i,o_j,b_j,I)\right), \tag{10}$$

where $\mathcal{F}$ denotes the trainable function embodying our SGG model, and the objective is to develop $\mathcal{F}$ into a robust mapping function that minimizes the disparity between unseen and seen triplets.

7. Additional Ablation Studies

We conduct supplementary ablation studies to evaluate the impact of the integrated capsule features and various hyperparameter configurations on the model efficacy.

7.1. Capsule Feature Encoder

In this study, we replace the capsule features in the CFE module with conventional features, while retaining the structural framework of CapsNet in CAPSGG to ensure model consistency. The results of this modified model are detailed in Tab. 7.

Analysis of the table reveals two key findings: 1) the removal of capsule features results in a significant decline in performance, thereby affirming the effectiveness of capsule features; 2) the zero-shot performance is comparable to other models, validating the robustness of our Three-Stream Pipeline.

Table 7. Ablation studies on CFE module

Models	PredCls	SGCls
	zR@50 / 100	zR@50 / 100
UVTransE [11]	16.5 / 18.9	3.3 / 3.9
EBM [32]	16.8 / 20.0	5.3 / 6.2
KGC* [42]	18.8 / 21.9	5.0 / 6.1
RGN* [43]	18.4 / 21.2	6.4 / 7.7
T-CAR [21]	31.9 / 34.9	9.3 / 10.6
CAPSGG (w/o)	21.2 / 30.0	6.8 / 8.0
CAPSGG	**37.7 / 46.1**	**20.0 / 24.3**

7.2. Hyperparameters Analysis

In addition to the hyperparameter λ, we have introduced two hyperparameters m^+ and m^-, which also influence the distinguishability of the model's positive and negative predicate class scores. The ablation studies on these hyperparameters are detailed in Table 8.

Table 8. Ablation studies on Hyperparameters m^+ and m^-

m^+	m^-	zR@K			ng-zR@K		
		20	50	100	20	50	100
0.9	0.1	19.2	33.2	42.1	22.2	42.4	60.4
0.9	0.2	22.0	36.0	44.1	25.1	46.3	62.6
0.8	0.1	20.6	34.4	42.7	23.1	44.3	61.7
0.8	**0.2**	**24.2**	**37.7**	**46.1**	**29.3**	**51.7**	**68.0**
0.7	0.3	19.3	34.5	43.8	21.1	44.0	62.2

A comparison of rows 1 and 2 reveals that increasing m^- enhances the model's performance, as it extends the upper boundary of the negative class scores, thereby diminishing the class distinguishability. Similarly, compared with rows 1 and 3, decreasing m^+ improves performance by lowering the lower boundary of the positive class scores, thereby further reducing class distinguishability. Based upon these observations, row 4 decreases m^+ and increases m^- simultaneously, thereby balancing the score ranges of positive and negative classes and achieving superior results. Fur-

Figure 5. Qualitative comparisons between our CapsNet and T-CAR [21] in the PredCls setting. The green color indicates the unseen triplets in test images. The blue color denotes the correctly classified triplets, and the red × represents the misclassified triplets.

thermore, as indicated in row 5, overly reducing the distinguishability between positive and negative class scores negatively impacts the model's performance.

8. Qualitative Visualization

We visually compare scene graphs generated by CAPSGG and T-CAR in Fig. 5, demonstrating the superior performance of our model on the zero-shot SGG task and its ability to effectively manage transformations. Fig. 5 shows that T-CAR [21] struggles with subject and object transformations under the same predicate, leading to substantial predicate switching. Specifically, the second and third rows illustrate that variations in the subject's posture or spatial configuration result in significant prediction shifts for a given predicate. Similarly, the third and fourth rows show that

changes in the object's posture and spatial configuration, with the same subject and predicate, also result in significant changes in predicate predictions. Our model effectively addresses these intrinsic transformations by encapsulating them within the capsule concept, generating distinct predicate instance parameters for transformed subjects and objects while maintaining consistent lengths across the spatial manifold for the same category. Our approach ensures consistent model responses to such transformations, effectively bridging the gap between seen and unseen domains.

Table 9. Model size and computational speed of CAPSGG compared with other methods

Models	Training		Inference FPS
	#Params(M)	Size(MB)	
Motifs [44]	378.6 (212.7)	1,444	4.2
IMP [40]	320.8 (155.0)	1,224	5.6
VTransE [45]	324.2 (158.3)	1,237	5.6
TDE [35]	382.1 (216.1)	1,458	3.1
EBM [32]	383.8 (217.9)	1,464	2.2
BGNN [22]	354.2 (188.3)	1,351	2.5
T-CAR [21]	344.8 (178.9)	1,315	3.9
CAPSGG	336.8 (170.9)	1,285	3.0

9. Model Size

CapsNet utilizes vectors instead of conventional scalar features, which increases the parameter requirements for equivalent representations and raises the model's parameter count — a notable limitation of CapsNet. To balance the number of parameters with performance, this study employs a single-layer Self-Attention routing mechanism, as referenced in [25], to reduce both the parameter count and computational load. Our experiments were conducted on two NVIDIA GeForce GTX 2080 Ti GPUs, with part of the data sourced from T-CAR [21]. The results, as shown in Tab. 9, demonstrate that our method achieves excellent zero-shot performance with moderate model size and computational efficiency.

10. Limitations

While our research has made significant enhancements in zero-shot SGG by effective modeling of predicates' equivariant features, the innovative application of CapsNet, and the construction of the CAPSGG framework, several limitations deserve discussion. Firstly, our approach, while effective in handling transformations, may have challenges with highly abstract or unconventional predicate relationships that are insufficiently represented in the training data. Secondly, although the computational complexity of CapsNet is alleviated by our single-layer Self-Attention Routing, it may still present scalability challenges for extremely large datasets or real-time applications. Future work will focus on addressing these limitations, by investigating more robust feature extraction techniques and enhancing the computational efficiency of our framework for complex scenarios.