RepSGG: Novel Representations of Entities and Relationships for Scene Graph Generation

Hengyue Liu, Bir Bhanu *Life Fellow, IEEE*

Abstract—Scene Graph Generation (SGG) has achieved significant progress recently. However, most previous works rely heavily on fixed-size entity representations based on bounding box proposals, anchors, or learnable queries. As each representation's cardinality has different trade-offs between performance and computation overhead, extracting highly representative features efficiently and dynamically is both challenging and crucial for SGG. In this work, a novel architecture called RepSGG is proposed to address the aforementioned challenges, formulating a subject as queries, an object as keys, and their relationship as the maximum attention weight between pairwise queries and keys. With more fine-grained and flexible representation power for entities and relationships, RepSGG learns to sample semantically discriminative and representative points for relationship inference. Moreover, the long-tailed distribution also poses a significant challenge for generalization of SGG. A run-time performance-guided logit adjustment (PGLA) strategy is proposed such that the relationship logits are modified via affine transformations based on run-time performance during training. This strategy encourages a more balanced performance between dominant and rare classes. Experimental results show that RepSGG achieves the state-of-the-art or comparable performance on the Visual Genome and Open Images V6 datasets with fast inference speed, demonstrating the efficacy and efficiency of the proposed methods.

Index Terms—Scene Graph Generation, Visual Relationship Detection, Long-tailed Learning, Human-Object Interaction

1 Introduction

To understand a scene, it is important to infer underlying properties of entities and the relationships between them. For a computer vision system to explicitly represent and reason about the detailed semantics, Johnson [1] *et al*. adopt and formalize scene graphs from computer graphics community. A scene graph is an explicit graph representation for modeling a visual scene, where entities are the nodes, and pairwise relationships are represented as edges. A relationship between two entities is denoted as a triplet of `<subject, predicate, object>`. In the context of this paper, the term "entity" denotes an instance of an object, while the term "object" specifically indicates an entity with semantic significance. Serving as a powerful representation, scene graph enables many down-stream high-level reasoning tasks such as image captioning [2], [3], image retrieval [1], [4], Visual Question answering [5], [6], [7] and image generation [4], [8]. Since SGG is built upon object detection where many off-the-shelf detectors [9], [10], [11], [12], [13], [14] can be used, limited attention has been directed towards the investigation of better feature representations for entities and relationships. Currently, there are mainly three types of entity visual feature representations:

1) *Box-based*: spatially-pooled features extracted from bounding boxes. Most SGG methods [15], [16] are based on R-CNN detectors [9], [10], [12] which use RoIPooling [9] or RoIAlign [12] for feature extraction.
2) *Point-based*: single-pixel features extracted from bounding box centers. These methods [17], [18], [19] utilize anchor-free detectors [11], [13], [14] to ground entities and relationships in a regression fashion.
3) *Query-based*: fixed-size learnable embeddings. These methods [20], [21], [22] build upon DETR [23] where message passing and matching are performed between entity and relationship embeddings.

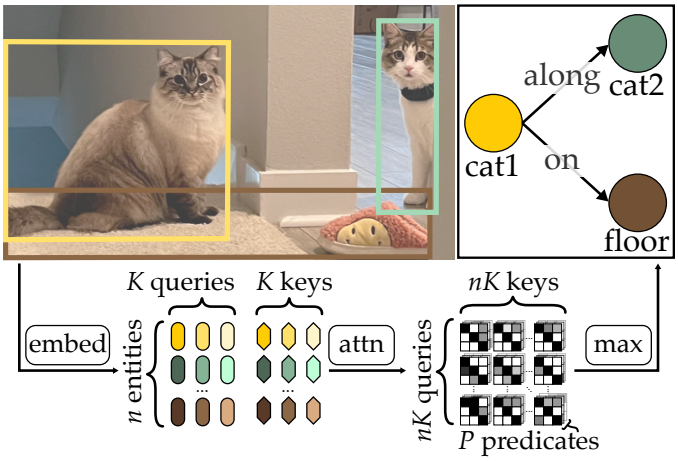


Fig. 1. Illustration of RepSGG. For n detected entities, each entity is represented by K subject queries and K object keys. The attention weights between queries and keys are projected as the predicate classification scores in the shape of $P \times nK \times nK$, where P is the number of predicates in a dataset. The final predicate classification is reduced to the shape of $P \times n \times n$ by max-pooling, and top predictions are collected as the scene graph. $K = 3$ and $n = 3$ in the example.

Each type has its own merits and drawbacks. While preserving the spatial appearance of entities, box-based features are computationally expensive and use more memory. Point-based features are often regressed and extracted at the

• *Hengyue Liu and Bir Bhanu are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA, 92521, USA. E-mails: hliu087@ucr.edu, bhanu@ee.ucr.edu.*

Manuscript received XXX; revised August XXX.

center of an entity's bounding box (referred to as entity center for the rest of the paper). Although such models achieve fast inference speed, their performance is relatively low due to the fact that point-based features are less semantically meaningful with limited cardinality. Query-based representation relies on a fixed number of learnable embeddings for decoding entities or relations, which considerably simplifies the SGG tasks. However, it suffers from training difficulties and lower performance compared to box-based methods.

Another issue of box-based, point-based and query-based entity representations lies in their predetermined granularities. Box-based features could be coarse and redundant, while point-based and query-based features are insufficient to represent entities with different semantics. For example, if there are two relation for the same entity `person`, `<person, eating, pizza>` and `<person, on, street>`, it is important to capture the context around the mouth and hands of the person for the first triplet, and around the feet and street for the second one. Box-based representation keeps the information of the entity `person`, but loses the fine-grained features around the mouth, hands and feet due to the low pooling resolution (*e.g.* 7×7). The same issue will arise for point-based and query-based representations. Increasing the pooling resolution, feature dimension, or number of queries could help, but the computation complexity will increase dramatically.

Regarding the relation representations, most works use compositional contextual features to perform predicate classification. Few researchers [18], [19] have explored different ways to represent relations in a regression fashion. These methods represent entities as keypoints (*e.g.*, entity centers), and relationships as points [17] or vectors [18], [19]. By regressing and grounding entities and relations geometrically, these methods achieve faster inference speed which is useful for down-stream tasks. However, regression with hand-crafted targets [18], [19] is deficient, especially on sparsely-annotated datasets [24]. Consequently, the performance of regression is inferior compared to predicate classification.

In this work, we seek new insights into alternative representations of entities and relationships for scene graph generation. We propose a novel architecture called RepSGG that is built upon the FCOS [14] entity detector with a specialised transformer-based [23], [25], [26] relationship decoder. As shown in Fig. 1, an entity class is represented with a set of learnable subject and object embeddings, named rep-embeddings, to learn diverse semantics. The subject and object embeddings are progressively augmented by dynamically sampled visual features and inter-embedding attentions through decoder layers. First, each embedding is updated by visual features sampled at semantic-dependent representative points (rep-points); then, a two-way cross-attention (subject-to-object and vice-versa) is performed to update both subject and object embeddings. For relationship prediction, subject and object embeddings are treated as queries and keys, with relationships quantified as the projected attention weights between these queries and keys.

Besides the proposed architecture, we also investigate the long-tailed problem in SGG. Inspired by logit adjustment [27], we propose a run-time performance-guided logit adjustment (PGLA) to achieve per-instance label-dependent loss modification. We measure and update the performance (*e.g.*, recall or precision) of predicate predictions per mini-batch, per iteration during training. Instead of adding a bias term to logits as in [27], we perform the affine transformation on the logits. We also measure run-time logit differences among predicates, named confusion logits, to further enlarge inter-class logit margins. For each predicate, the among of the adjustment will be determined by its frequency in the training set, run-time performance, and confusion logits.

The contributions of this paper are:

1) Significantly different from most existing SGG approaches, RepSGG introduces a novel SGG paradigm in which entities are expressed as queries and keys, and relationships are represented as their attention weights. It explores a natural approach of capturing visual and semantic features progressively, and encapsulating relationships as attention weights which encode the edge confidence and directionality effectively.
2) A run-time performance-guided logit adjustment (PGLA) strategy is proposed to mitigate the long-tailed problem. PGLA is a simple, yet effective, model-agnostic, and cost-free solution that achieves a more balanced performance on unbalanced data. The choice of loss (*e.g.*, binary cross-entropy or cross-entropy) and performance metric (*e.g.*, recall or precision) are task-dependent, and this adaptability can be extended to a range of settings and tasks.
3) Extensive experiments on the Visual Genome and Open Images V6 datasets demonstrate the effectiveness of the proposed approach. Beyond standard SGG metrics, we also report the zero-shot mean recall (zs-mR) on our method and several state-of-the-art methods. RepSGG exhibits superior robustness and generalization capabilities on out-of-distribution data.

The remainder of the paper is structured as follows: Section 2 provides a literature review of related work in scene graph generation. Section 3 outlines the proposed approach. Section 4 discusses experimental results and ablation studies. Section 5 presents the limitations and future work. Section 6 concludes this paper.

2 Related Work

The task of scene graph generation involves numerous aspects, and our focus lies on entity and relation representations, as well as the long-tailed problem.

2.1 Feature Representations

As discussed in Section 1, the entity representation is either box-based, point-based, or query-based. Traditional SGG methods [15], [28], [29] utilize a pre-trained detector [10], [12] to extract a set of entity bounding boxes and their corresponding feature maps via feature pooling (RoIPool [10] or RoIAlign [12]). The visual features for each entity, commonly referred to as appearance features, are represented as a tensor of shape $c \times h \times w$, where c is the number of channels of the feature, and $h \times w$ is the feature spatial size. Those features are used to construct visual context for predicate classification. To incorporate the relative position between entities, researchers use the geometric layout encoding [30],

union of bounding boxes mask encoding [31], or geometric constraints [32] for a better visual representation.

One-stage anchor-free entity detectors [13], [14], [33] have recently gained popularity due to their simplicity and efficiency. In these works, entities are directly regressed at pixels in feature maps, where the entity center or corners are selected as the ground-truth targets. Instead of pooling features of various shapes, extracting features at multiple pixels is much faster and consumes less memory. Several works [17], [18], [19] explore such point-based entity representation for SGG. Pixel2Graph [17] grounds edges at the midpoints between the bounding box centers of subjects and objects (referred to as subject and object centers for the rest of the paper). FCSGG [18] uses relation affinity fields to encode the relations as 2D vectors "flow" from the subject to object centers. CoRF [19] extends the concept of fields by composing more regression targets per pixel.

Recent works in scene graph generation have explored transformer-based models to improve the performance. Several works [16], [34], [35] start to replace the RNN-based context decoders [28], [29] with multi-head self-attention [25]. Subsequently, other works [36], [37], [38] explore ways to construct subject, object and predicate queries with variants of transformers. With the success of DETR [23], more works [20], [21], [22] study the query-based representations of entities and relations. For DETR-like approaches, there are a fixed number of learnable entity and predicate queries, which will be decoded as output triplets in an end-to-end manner.

The strengths and weaknesses of different types of entity representation vary based on their granularity and flexibility. For example, box-based features are extracted via RoIAlign [12] with a fixed shape of $d \times h \times w$, such as $256 \times 7 \times 7$ for SGG tasks. Although preserving entities' spatial configuration, box-based features may lose semantic details due to the pooling operation. Furthermore, the fact that features are pooled into the same shape regardless of actual sizes of entities may result in the loss of semantic details in relationship inference. Another drawback of the box-based representation is that it is computationally expensive to compute $\mathcal{O}(n^2)$ relationships for n entity proposals. Sampling candidate entities and relationships is commonly used during training. On the other hand, the point-based methods significantly reduce the computational cost by using features of shape $d \times 1$. By reformulating the SGG in a per-pixel regression fashion, point-based methods [18], [19] achieve much faster inference speed. However, the performance is relatively lower due to the coarse entity representations and handcrafted relationship targets. The query-based entity representation provides a way to perform object detection and SGG in an end-to-end manner. It exhibits greater capability in capturing semantics compared to convolutional regression, achieving better performance than point-based methods. Nevertheless, challenges arise for query-based methods due to factors like feature cardinality, the constraint of a fixed number of learnable queries, and increased complexity in both model design and post-processing. It is also difficult for query-based methods to perform predicate classification and scene graph classification due to the end-to-end prediction manner.

This work proposes a novel entity representation by using a set of semantically representative embeddings, which are augmented by visually representative points (rep-points) [39], [40] progressively. Different sets of rep-points are sampled dynamically w.r.t. entity reference points (centers) to update subject and object embeddings, respectively. Cross-attention between subject and object embeddings are also performed to achieve message passing. This approach allows for each entity instance to be represented as distinct queries and keys, which is more flexible than box-based representation and more fine-grained than point-based and query-based representations. Additionally, the proposed method eliminates the need for composite or predicate queries [20], [22], [36], as the predicate of a relationship can be computed as the multi-head attention weights between the subjects (as queries) and objects (as keys). Unlike triplet classification, modeling relationships as attention weights preserves the directional information among subjects and objects, and captures more semantics.

2.2 Long-tailed Distributions

Long-tailed data distribution has been a key challenge in visual recognition [41], and it has been addressed in the recent literature on SGG [42]. In order to tackle this problem, various approaches have been proposed, such as data re-sampling [43], [44], [45], [46], de-biasing [16], [47], [48], [49], [50], and loss modification [51], [52], [53], [54], [55], [56], [57]. De-biasing methods require pre-trained biased models for initialization and then finetune the model. Loss modification methods generally assign a weight vector to the cross-entropy loss for predicate classification, with higher weights to tail classes and lower weights to head classes.

In this work, instead of re-weighting the loss function, another type of approach is applied by directly modifying the classification logits [27], [58], [59], [60]. A run-time performance-guided logit adjustment strategy is presented which offers a dynamic and effective control over the relative contributions of labels in the loss.

3 Technical Approach

In this section, we first provide the preliminaries of modeling entity detection in a per-pixel prediction fashion. Subsequently, we introduce the RepSGG architecture, consisting of an entity detector, an entity encoder, a relationship encoder, and a relationship output layer. An illustration of RepSGG is shown in Fig. 2. Finally, we present several training strategies, including PGLA, addressing the challenges posed by long-tailed distributions and sparsely annotated data.

3.1 Entity Detection

Our model is built upon a one-stage anchor-free detector, namely FCOS [14]. Different from commonly used anchor-based R-CNN approaches for generating object proposals and features, entity detections are decoded from regressed dense features. More specifically, an input image $\mathbf{I} \in \mathbb{R}^{H^0 \times W^0 \times 3}$ will go through a backbone CNN (*e.g.*, ResNet-50 [61]) followed by a feature pyramid network (FPN) [62] to generate 5 scale levels of visual feature maps. Feature maps of different levels have different down-sampled spatial size

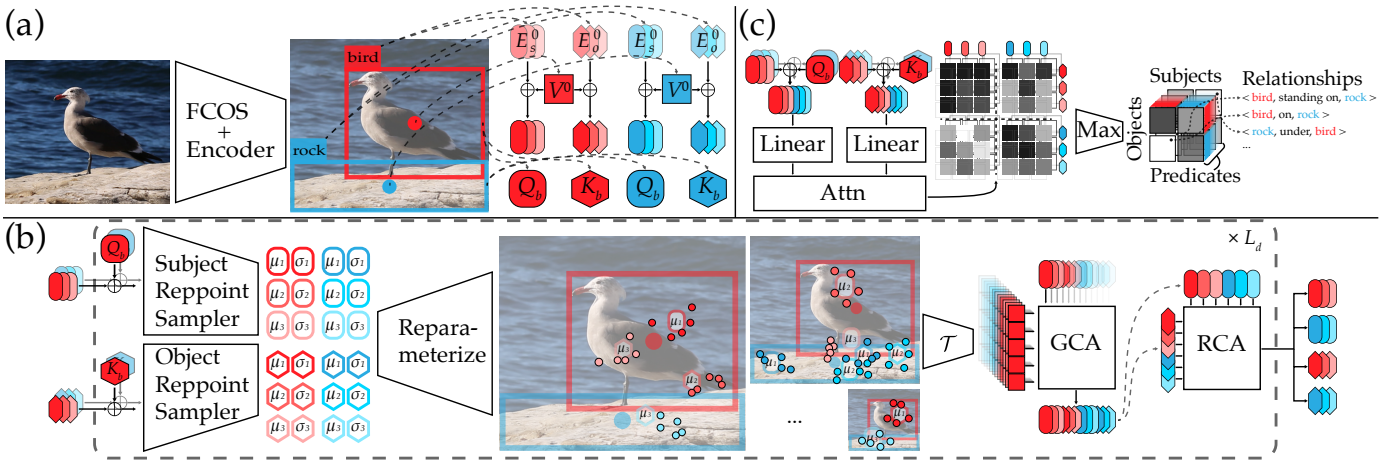


Fig. 2. An illustration of RepSGG. (a) Entity detection and encoder: firstly, the FCOS entity detector detects a `bird` (colored in red) and `rock` (colored in blue). For each entity, K subject and object rep-embeddings ($K = 3$ in this illustration) are retrieved based on the entity's class label as $\mathbf{E}_s^0$ (shaped as a rounded rectangle) and $\mathbf{E}_o^0$ (shaped as a hexagon), respectively. The entity encoder generates entities' visual features as $\mathbf{V}^0$, and semantic-specific bounding box embeddings $\mathbf{Q}_b$ and $\mathbf{K}_b$. (b) Relationship encoder: the initial *queries* $\mathbf{Q}^0$ (representing subjects) are generated by adding $\mathbf{V}^0$ and $\mathbf{E}_s^0$ with $\mathbf{Q}_b$ acting as positional embeddings, and likewise for the *keys* $\mathbf{K}^0$ (representing objects). Semantic-specific visual features are sampled around entities dynamically based on input queries and keys via rep-point samplers (5 samples per rep-embedding in this illustration), and then utilized to update queries and keys via a GCA layer to gather more visual context. Subsequently, the cross-attention between queries and keys are performed via a RCA layer to further capture semantic features. (c) Relationship output layer: the pair-wise relationship scores are computed as the sigmoid activation of raw attention weights between the linear projections of queries and keys. The group-wise maximum scores are then taken as the predicate classification scores.

w.r.t. the original image size and thus are used for detecting entities of different sizes. To generate entity outputs, 3 fully convolutional detection heads shared between 5 scale levels are used, generating dense feature maps that provide entity classification, bounding box regression, and center-ness scores, respectively. Specifically, at each spatial location on a level of feature maps, FCOS directly predicts the entity category and the relative offsets from the four sides of the bounding box to the location. Locations with a final score weighted by the classification and center-ness scores over 0.2, followed by a non-maximum suppression (NMS) operation, are considered as positive detections. The entity detections are gathered as $\mathcal{B} : \{b^i\}_{i=1}^n$, where $b^i = (x_0^i, y_0^i, x_1^i, y_1^i, z^i, c^i)$, (x_0^i, y_0^i) and (x_1^i, y_1^i) denote the coordinates of the top-left and bottom-right corners of the bounding box, $z^i \in \{0, 1, 2, 3, 4\}$ is the scale level at which the detection is decoded, $c^i \in \{0, ..., C-1\}$ is the predicted entity label for a dataset containing C classes, and n is the number of detected entities. For SGG tasks, the top 100 detections are kept. Since the positive training targets are defined around the center of bounding boxes, the center is considered as the reference point of an entity in this paper. We maintain the FCOS architecture and training/testing settings unchanged, focusing instead on grounding entity features in a compact form to enable efficient relationship inference.

3.2 Entity Encoder

To further encourage information exchange among different scales and prepare for the relationship encoder, a deformable transformer encoder [26], consisting of L_e encoder layers, is applied on the FPN features without changing their shapes. At each location, the deformable attention is performed by querying the FPN features at that location to features dynamically sampled around the location across all 5 scale levels. Such mechanism allows efficient multi-level feature aggregation, which is important for relationship inference as it requires more spatial context around entities. The output features of the deformable transformer encoder are considered as the visual features of the image. Following [26], the 2D positional embeddings [25] are generated and added with a learnable scale-level embedding. Note that the visual features and positional embeddings are multi-scale features. To merge levels of features, they are resized via bilinear interpolation to the shape of the largest feature map respectively. The interpolated features are then stacked along the level dimension, producing the visual features $\mathbf{V} \in \mathbb{R}^{5\times H\times W\times d}$ and positional embeddings $\mathbf{PE} \in \mathbb{R}^{5\times H\times W\times d}$, where $H = \lfloor H^0/8 \rfloor$, $W = \lfloor W^0/8 \rfloor$ and $d = 256$.

Entities can have different characteristics and roles depending on the context. For instance, detecting the relationship `<man, on, street>` requires the visual context around the feet of the `man` to determine the predicate `on`. While for `<man, holding, apple>`, it relies on the visual context surrounding the hands of the `man`. To account for such contextual variations, we initiate the representation of an entity with a distinct set of representative embeddings (rep-embeddings) in the semantic space, which we subsequently employ for relationship inference. Concretely, we construct subject rep-embeddings $\mathbf{E}_s$ and object rep-embeddings $\mathbf{E}_o$ of shape $C \times K \times d$, where K is the number of entity's class-specific embeddings. $\mathbf{E}_s$ and $\mathbf{E}_o$ are fixed-size learnable parameters initialized randomly, and are learnt through training. For every entity class, there are K distinct subject embeddings and K distinct object embeddings, respectively. These class-specific and semantic-specific embeddings serve as feature prototypes to characterize an entity from a particular class being a subject or object. By having distinct embeddings for entities being

subjects and objects, our method can better capture the nuanced relationships and contextual information present in complex scenes. We conduct an ablation study using identical rep-embeddings for subjects and objects ($\mathbf{E}_s = \mathbf{E}_o$) in Section 4.5, where we show having distinct rep-embeddings achieves better results compared with identical ones.

3.3 Relationship Encoder

In this section, we describe the process of constructing semantic-specific entity features based on rep-embeddings ($\mathbf{E}_s$ and $\mathbf{E}_o$), visual features $\mathbf{V}$, positional embeddings $\mathbf{PE}$, and entity detections $\mathcal{B}$. For each detected entity, the relationship encoder generates a subject and object feature representations, respectively. Then, it employs the attention mechanism [25] to aggregate local visual features and capture dependencies between all subjects and objects.

The relationship encoder is composed of a stack of L_d encoder layers, where the initial inputs are formed by fusing visual features and bounding box embeddings with subject rep-embeddings and object rep-embeddings, respectively. The outputs are encoded subject and object features of the same shape as inputs. Each layer has a rep-point sampler, two group cross-attention layers, and a two-way relational cross-attention layer. To simplify the terms and make them compatible with the attention mechanism, we refer to the subject embeddings as *queries*, and object embeddings as *keys* for the rest of the paper.

3.3.1 Initial Queries and Keys

Rep-embeddings only provide identities of entity classes and semantics, without considering visual and spatial contexts. To integrate such information, the entity-specific visual and spatial features are sampled from the entity reference points, and fused with rep-embeddings to create subject and object features as the initial queries and keys to the relationship encoder.

Let the reference point $\boldsymbol{p} \in [0,1]^3$ be the normalized coordinates, where $(0,0,0)$ and $(1,1,1)$ indicate the top-left corner at the lowest scale level, and bottom-right corner at the highest scale level of the features, then for $n \times m$ reference points $\mathbf{P} = \{\boldsymbol{p}^{i,1}, \ldots, \boldsymbol{p}^{i,m}\}_{i=1}^{n}$, the point sampler function is defined as

$$\mathcal{T}(\,\cdot\,, \mathbf{P}) : \mathbb{R}^{5\times h\times w\times d} \times \mathbb{R}^{n\times m\times 3} \to \mathbb{R}^{n\times m\times d}, \tag{1}$$

which is achieved by bilinear interpolation. To prepare semantic-agnostic entity features, the normalized centers of bounding boxes $\mathbf{P}^0 \in \mathbb{R}^{n\times 1\times 3}$ are used as the reference points derived from the entity detections $\mathcal{B}$. The point sampler is applied on $\mathbf{V}$ and $\mathbf{PE}$ to get the corresponding features at the entities' reference points as:

$$\begin{aligned} \mathbf{V}^0 &= \mathcal{T}(\mathbf{V}, \mathbf{P}^0) \\ \mathbf{PE}^0 &= \mathcal{T}(\mathbf{PE}, \mathbf{P}^0) \\ \mathbf{P}^0 &= \left\{ \left(\frac{x_0^i + x_1^i}{2W}, \frac{y_0^i + y_1^i}{2H}, \frac{z^i}{4} \right) \right\}_{i=1}^{n}. \end{aligned} \tag{2}$$

For assigning semantics to entities, the subject and object embeddings are gathered from the corresponding indices of predicted entity labels $\{c^i\}_{i=1}^n$ as $\mathbf{E}_s^0 \in \mathbb{R}^{n\times K\times d}$ and $\mathbf{E}_o^0 \in \mathbb{R}^{n\times K\times d}$. The subject and object embeddings are class-dependent learnable parameters which capture the semantics of an entity class being the subject and object. Subject queries $\mathbf{Q}^0$ and object keys $\mathbf{K}^0$ are then constructed by adding the corresponding embeddings with the visual features $\mathbf{V}^0$:

$$\begin{aligned} \mathbf{Q}^0 &= \mathbf{E}_s^0 + \mathbf{V}^0 \\ \mathbf{K}^0 &= \mathbf{E}_o^0 + \mathbf{V}^0. \end{aligned} \tag{3}$$

By merging instance-specific visual features with class-specific embeddings, the queries (or keys) retain semantic similarities within their entity classes while also diversifying the instance-wise entity representations.

The bounding box also plays a crucial role in determining the spatial relationship between entities. Hence, the bounding box coordinates are mapped into embeddings. First, the positional embeddings $\mathbf{PE}$ of the top-left and bottom-right corners are sampled via (1). Two learnable embeddings indicating "top-left corner" and "bottom-right corner" are added with the corresponding corners' positional embeddings, respectively. The two corner embeddings are then concatenated and fed to a fully-connected layer, resulting in the box embeddings. Lastly, two learnable embeddings are added with the box embeddings to construct subject and object box embeddings respectively, denoted as $\mathbf{Q}_b \in \mathbb{R}^{n\times d}$ and $\mathbf{K}_b \in \mathbb{R}^{n\times d}$.

3.3.2 Rep-point Sampler

For rep-embeddings to capture more visual context, we propose a dynamic approach for sampling features from representative points (rep-points) and message passing via attentions. Two rep-point samplers are implemented to sample subject and object rep-points, since queries and keys serve distinct roles in conveying subject and object semantics respectively. Specifically, a multi-layer perceptron (MLP) is used as the sampler to predict the points of interest for each entity. To achieve semantic-specific sampling, the MLP weights are split into K groups (in practice, group 1D convolution is used), and k-th group is applied on the k-th slice of the inputs (queries or keys) along the K dimension of rep-embeddings. To further increase the diversity of sampling and prevent overfitting on few points, instead of directly predicting the coordinates, the distribution parameters of offsets w.r.t. the reference points are predicted, following the variational autoencoder (VAE) and reparameterization trick [63]. Let the input queries to the l-th layer be $\mathbf{Q}^{l-1}$, the subject rep-point offsets are defined as

$$\begin{aligned} \Delta\mathbf{P}_s^l &= \boldsymbol{\mu}_s^l + \boldsymbol{\sigma}_s^l \odot \boldsymbol{\epsilon}, \\ \boldsymbol{\mu}_s^l, \boldsymbol{\sigma}_s^l &= \mathrm{MLP}(\mathbf{Q}^{l-1} + \mathbf{Q}_b) \end{aligned} \tag{4}$$

where means $\boldsymbol{\mu}_s^l \in \mathbb{R}^{n\times K\times 3}$ and standard deviations (stds) $\boldsymbol{\sigma}_s^l \in \mathbb{R}^{n\times K\times 3}$ are the outputs of the subject rep-point sampler, $\odot$ is the element-wise product, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_3)$. To estimate robust parameters, m points are randomly sampled per parameter during training, namely $\boldsymbol{\epsilon} \in \mathbb{R}^{n\times K\times m\times 3}$. Similarly, the object rep-point offsets are obtained as $\Delta\mathbf{P}_o^l$. By adding the sampled rep-points offsets to the reference points, the sampled points are obtained as

$$\begin{aligned} \mathbf{P}_s^l &= \mathbf{P}_s^{l-1} + \Delta\mathbf{P}_s^l,\ l = 1 \ldots L \\ \mathbf{P}_o^l &= \mathbf{P}_o^{l-1} + \Delta\mathbf{P}_o^l,\ l = 1 \ldots L, \end{aligned} \tag{5}$$

where $\mathbf{P}_s^0 = \mathbf{P}_o^0 = \mathbf{P}^0$ with expanded shapes of $n \times 1 \times 1 \times 3$. The offsets are accumulated through encoder layers from the original entity centers $\mathbf{P}^0$, so relevant features can be aggregated as the encoder layer goes deeper. Accordingly, rep-point visual features and positional embeddings are sampled via (1):

$$\begin{aligned} \mathbf{V}_s^l &= \mathcal{T}(\mathbf{V}, \mathbf{P}_s^l),\ \mathbf{PE}_s^l = \mathcal{T}(\mathbf{PE}, \mathbf{P}_s^l), \\ \mathbf{V}_o^l &= \mathcal{T}(\mathbf{V}, \mathbf{P}_o^l),\ \mathbf{PE}_o^l = \mathcal{T}(\mathbf{PE}, \mathbf{P}_o^l). \end{aligned} \tag{6}$$

The rep-point sampler provides a probabilistic mapping from visual cues to the semantic space, which helps finding visually relevant features with semantic significance. Since our task is prediction rather than generation, to achieve deterministic SGG evaluation results, the stochastic behavior of rep-point samplers is transformed into a deterministic one by sampling within a range with a fixed step size during inference. In particular, the noise vector $\boldsymbol{\epsilon}$ for each offset is replaced by a set $\{-\xi, -\xi+1, ..., 0, ..., \xi\}^3$, where ξ is the range of consideration which is set to 3 by default. In other words, rep-points are sampled within the "3σ" range with a step size of "σ" along the width, height, and scale dimensions in a combinatorial manner. For a subject rep-point sampler for the k-th rep-embedding, at layer l, the rep-points are derived the Cartesian product $\prod_{dim=1}^{3}(\boldsymbol{\mu}_s^{l,k,dim} - 3\boldsymbol{\sigma}_s^{l,k,dim}, \cdots, \boldsymbol{\mu}_s^{l,k,dim} + 3\boldsymbol{\sigma}_s^{l,k,dim})$. In total, there are $7^3 = 343$ rep-points sampled per mean. Comparisons of performance and inference speed w.r.t. ξ are provided in Section 4.

3.3.3 Group Cross-Attention

The group cross-attention (GCA) captures the visual features that correspond to each subject and object rep-embedding by computing their attention scores, respectively. The application of GCA involves performing separate interactions between queries and subject rep-point features, as well as between keys and object rep-point features. The subject GCA for the i-th entity is defined as

$$\begin{aligned} \mathbf{Q}^{l,i} &= \text{GCA}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}) \\ &= \text{softmax}(\boldsymbol{q}\boldsymbol{k}^T/\sqrt{d_G})\ \boldsymbol{v} \\ \boldsymbol{q} &= \text{Linear}(\mathbf{Q}^{l-1,i} + \mathbf{Q}_b^i) \in \mathbb{R}^{h_G \times K \times d_G} \\ \boldsymbol{k} &= \text{Linear}(\mathbf{V}_s^{l,i} + \mathbf{PE}_s^{l,i}) \in \mathbb{R}^{h_G \times K \times m \times d_G} \\ \boldsymbol{v} &= \text{Linear}(\mathbf{V}_s^{l,i}) \in \mathbb{R}^{h_G \times K \times m \times d_G}, \end{aligned} \tag{7}$$

where i indexes the entity, $\text{Linear}(\cdot)$ is a fully-connected layer ($\boldsymbol{q}$, $\boldsymbol{k}$, and $\boldsymbol{v}$ are projected with different parameters), h_G is the number of attention heads, and d_G is the dimension of each head. GCA is performed independently among groups in parallel, where the cross-attention between a rep-embedding and its corresponding sampled rep-point features are performed. Following the transformer architecture [25], multi-head outputs are concatenated and projected with a fully-connected layer, and a residual connection [61] with layer normalization [64] is added. Likewise, another GCA layer with different parameters is performed for keys as $\mathbf{K}^l = \text{GCA}\left(\text{Linear}(\mathbf{K}^{l-1} + \mathbf{K}_b), \text{Linear}(\mathbf{V}_o^l + \mathbf{PE}_o^l), \text{Linear}(\mathbf{V}_o^l)\right)$. GCA allows each rep-embedding to focus on different visual features, carrying the relevant ones along the way.

3.3.4 Two-Way Relational Cross-Attention

In GCA, queries and keys are updated by their corresponding sampled features. In a two-way relational cross-attention (RCA) layer, queries are updated by keys, and vice versa. Firstly, the raw attention weights $\mathbf{A}^l$ are computed between flattened $\mathbf{Q}^l$ and $\mathbf{K}^l$. Since the projections of queries and keys are different, the attention weights are not symmetric and can be normalized along different dimensions. Softmax is then applied on $\mathbf{A}^l$ along the dimension of keys, and along the dimension of queries to obtain two-way attention weights. Finally, queries and keys are updated by multiplying the corresponding attention weights with values. The two-way relational cross-attention is formally defined as following:

$$\begin{aligned} \mathbf{Q}^l, \mathbf{K}^l &= \text{RCA}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}_q, \boldsymbol{v}_k) \\ \boldsymbol{q} &= \text{Linear}(\mathbf{Q}^l + \mathbf{Q}_b) \\ \boldsymbol{k} &= \text{Linear}(\mathbf{K}^l + \mathbf{K}_b) \\ \boldsymbol{v}_q &= \text{Linear}(\mathbf{Q}^l) \\ \boldsymbol{v}_k &= \text{Linear}(\mathbf{K}^l) \\ \mathbf{A}^l &= \boldsymbol{q}\boldsymbol{k}^T/\sqrt{d_R} \in \mathbb{R}^{h_R \times n_q \times n_k} \\ \mathbf{A}_k^l &= \text{softmax}(\mathbf{A}^l)\ s.t.\ \textstyle\sum_{j=1}^{n_k} \mathbf{A}_k^{l,*,*,j} = 1 \\ \mathbf{A}_q^l &= \text{softmax}(\mathbf{A}^l)\ s.t.\ \textstyle\sum_{i=1}^{n_q} \mathbf{A}_q^{l,*,i,*} = 1 \\ \mathbf{Q}^l &= \mathbf{A}_k^l \boldsymbol{v}_k,\ \mathbf{K}^l = (\mathbf{A}_q^l)^T \boldsymbol{v}_q, \end{aligned} \tag{8}$$

where $n_q = n_k = n \times K$, h_R is the number of attention heads, and d_R is the dimension of each head, "$*$" denotes any index along the specific dimension. Additionally, two MLPs are used for projecting the output queries and keys respectively, following the feed-forward network design in [25]. Without abuse of notation, the notations of output queries $\mathbf{Q}^l$ and keys $\mathbf{K}^l$ of RCA layers remain the same. After L_d relationship encoder layers, the outputs $\mathbf{Q}^{L_d}$ and $\mathbf{K}^{L_d}$ are obtained which are used for predicate prediction.

3.4 Relationships as Attention Weights

For most SGG methods using either box-based, or query-based representation, predicate classification is performed on triplet features in different forms of feature fusions. For example, box-based methods use the concatenation of pairwise entity features and their union-box features, and query-based methods use learnable triplet embeddings to perform classification directly. Neither of them can capture the directional information of scene graphs explicitly which could cause learning bias and overfitting on dominant visual configurations. A very simple case is that there is a common triplet `<man, on, street>` in the dataset, the learnt model will likely predict `on` if it detects `man` and `street` concurrently. Conversely, triplets such as `<man, standing on, street>` and `<street, under, man>` are considered as incorrect predictions and are treated as negative examples during training, despite being semantically valid triplets. The presence of rare, bidirectional [65], and unannotated relationships [46] hinders the learning of representative semantics for SGG methods.

Similar to the RCA discussed in Section 3.3.4, $\mathbf{Q}^{L_d}$ and $\mathbf{K}^{L_d}$ are projected with h_A heads, and each head has

d_A dimensions. Unnormalized attention weights $\mathbf{A}^{L_d} \in \mathbb{R}^{h_A \times n_q \times n_k}$ are computed between projected queries and keys. Different from RCA, the multi-head attention weights are not multiplied by projected values. The asymmetric nature of dot-product attention serves as a natural metric for quantifying the relationship between subjects and objects. Moreover, attention weights of each head captures distinct semantics, similar to the way feature channels operate. Therefore, the attention weights can be mapped to the predicate classification $\mathbf{Y} \in \mathbb{R}^{P \times n_q \times n_k}$ via a fully-connected layer, where P is the number of predicate classes of a dataset. In parallel, a binary relation mask $\mathbf{H} \in \mathbb{R}^{n_q \times n_k}$ is also predicted to classify if a relationship exists between a pair of rep-embeddings. The relation mask is used for suppressing low-quality predictions. Formally, $\mathbf{Y}$ and $\mathbf{H}$ are obtained as

$$\begin{aligned} \mathbf{Y} &= \text{Linear}(\mathbf{A}^{L_d}) \\ \mathbf{H} &= \text{Linear}(\mathbf{A}^{L_d}) \\ \mathbf{A}^{L_d} &= \boldsymbol{q}\boldsymbol{k}^T/\sqrt{d_A} + \boldsymbol{b}_A \\ \boldsymbol{q} &= \text{Linear}(\mathbf{Q}^L + \mathbf{Q}_b) \\ \boldsymbol{k} &= \text{Linear}(\mathbf{K}^L + \mathbf{K}_b), \end{aligned} \tag{9}$$

where $\boldsymbol{b}_A$ is an added bias term. $\mathbf{Y}$ represents relationships between subject and object rep-embeddings instead of subjects and objects. To get pairwise relationships between entities, $\mathbf{Y}$ is re-arranged to $\mathbb{R}^{P \times n \times n \times K^2}$. During training, Gumbel-Softmax [66] is applied over the last dimension of $\mathbf{Y}$ to sample the rep-embedding pairs with the largest logits, and $\mathbf{Y}$ is reduced to the shape of $\mathbb{R}^{P \times n \times n}$. An annealing schedule is applied that changes the Gumbel-Softmax temperature from 10 to 0.5 gradually through the first 30% iterations. During inference, the maximum logits over the last dimension are chosen. The relation mask $\mathbf{H}$ undergoes the same re-arrangement operation, followed by selecting the maximum value for both training and inference. The final predicate classification score is defined as $(\sigma(\mathbf{H}) \cdot \sigma(\mathbf{Y}))^\zeta$, where $\sigma(\cdot)$ is the sigmoid function, and ζ is a hyper-parameter to balance between predicate scores and entity detection scores. During evaluation, the triplet score is computed by multiplying the predicate classification score, subject detection score, and object detection score. We empirically set $\zeta = 0.5$ for evaluating on the Visual Genome [24] dataset where recall is the primary metric. For the Open Images [67] dataset where precision is the main metric, we find that assigning more weights on entity detection scores yields improved results, effectively lowering the ranking of false positive detections. We set $\zeta = 0.1$ for evaluation on the Open Images dataset.

3.5 Training

In this section, we discuss the training losses and strategies to address the challenges posed by long-tailed sparsely-annotated data. The hyper-parameters and losses for entity detection remain the same as in FCOS [14].

3.5.1 Losses

Due to the potential existence of multiple relationships (directional or bidirectional) between two entities, the scene graph frequently exhibits a multi-graph structure. The predicate classification is considered as a multi-label multi-class classification problem. Consequently, we use the binary cross-entropy (BCE) instead of softmax to supervise the predicate classification and relation mask. To balance well-learned and hard examples, the focal loss (FL) [68] for BCE is used. Specifically, given the predicted logits $\hat{\mathbf{Y}}$ and ground-truth labels $\mathbf{Y} \in \{0,1\}^{P \times n \times n}$, the focal BCE is defined as

$$\text{FL}(\hat{\mathbf{Y}}, \mathbf{Y}) = \begin{cases} -\frac{\alpha}{N_{pos}} \sum_{p,i,j} (1 - \sigma(\hat{\mathbf{Y}}^{p,i,j}))^\gamma \log(\sigma(\hat{\mathbf{Y}}^{p,i,j})), & \mathbf{Y}^{p,i,j} = 1 \\ -\frac{1-\alpha}{N_{pos}} \sum_{p,i,j} \sigma(\hat{\mathbf{Y}}^{p,i,j})^\gamma \log(1 - \sigma(\hat{\mathbf{Y}}^{p,i,j})), & \mathbf{Y}^{p,i,j} = 0, \end{cases} \tag{10}$$

where α is a class-balance weighting factor, γ is the focal factor, p indexes the predicate classes, i indexes subjects, j indexes objects, and $N_{pos} = \sum_{p,i,j} \mathbf{Y}^{p,i,j}$ is the number of ground-truth triplets. As the predicate prediction forms a fully-connected graph among n entities, the ground-truth $\mathbf{Y}$ is inherently sparse with few ones. We select $\alpha = 0.75$

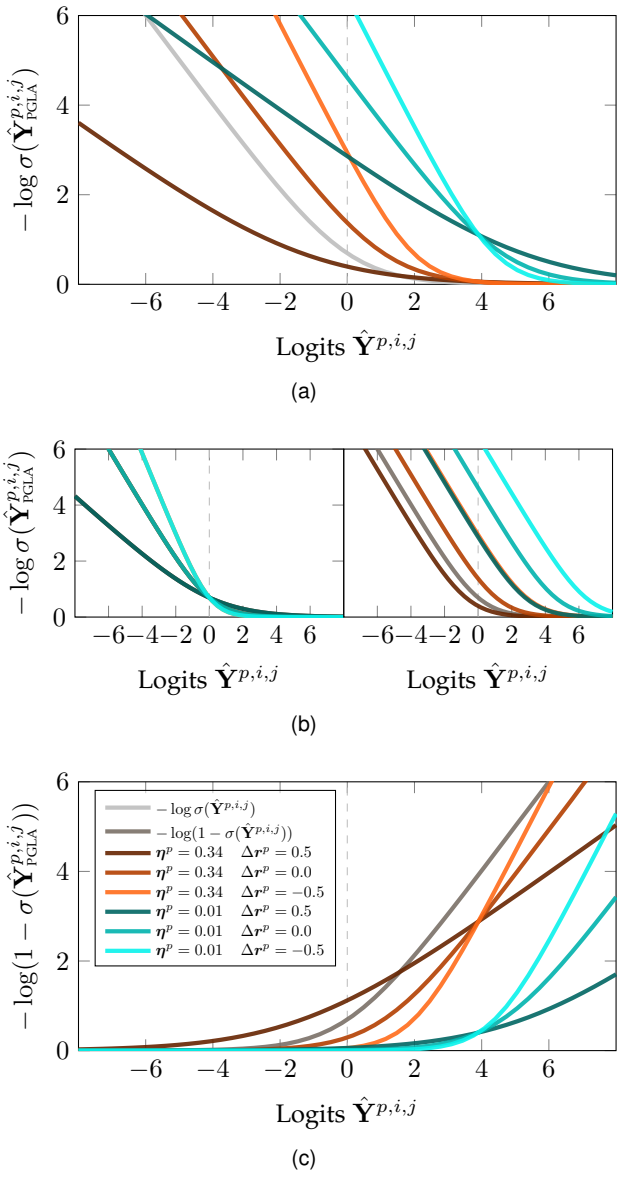


Fig. 3. BCE loss on PGLA-adjusted logits. (a) Loss on a positive predicate p where $\mathbf{Y}^{p,i,j} = 1$. (b) Loss on a positive predicate p with $\mathbf{B} = \mathbf{0}$ (left), and with $\mathbf{W} = \mathbf{1}$ (right). Legend follows (c) Loss on a negative predicate p where $\mathbf{Y}^{p,i,j} = 0$. The legend is shared across (a) - (c), and gray lines in figures represent the BCE loss on original logits.

to prioritize generating larger logits for positive predicates, rather than penalizing negative predicates. For the focal factor γ, we set it differently based on the predicate frequency of training data. Let the predicate priors be $\boldsymbol{\eta}$, *e.g.*, the empirical predicate class frequencies in the training dataset, we compute a predicate-specific $\boldsymbol{\gamma}$ to replace γ in (10) as

$$\boldsymbol{\gamma} = \gamma \cdot \frac{\boldsymbol{\eta} - \min(\boldsymbol{\eta})}{\max(\boldsymbol{\eta}) - \min(\boldsymbol{\eta})}, \tag{11}$$

where $\gamma = 2$, $\min(\cdot)$ and $\max(\cdot)$ are operations to get the minimum and maximum value respectively. For the tail predicates, $\boldsymbol{\gamma}^p$ is small so that it encourages the logits to be larger. For the head predicates, $\boldsymbol{\gamma}^p$ becomes larger and down-weights the loss. For supervising the relation mask, we empirically select $\alpha = 0.75$ and $\gamma = 2$.

As discussed in [17], [18], [46], the sparsity of data annotations for relations, coupled with the presence of numerous unannotated ones, leads to semantic ambiguity and poses challenges during training. Therefore, simply considering triplets without ground-truth annotations as negative is not optimal. For training the relation mask $\hat{\mathbf{H}} \in \mathbb{R}^{n \times n}$ among n entities, we sub-sample the negative triplets with a ratio of 10:1 in proportion to the number of ground-truth triplets. Additional, we employ a margin ranking loss for predicted relation classification $\hat{\mathbf{Y}}$. Instead of supervising negative samples (where $\mathbf{Y}^{p,i,j} = 0, \forall p$) with labels of zero in BCE, per-predicate margins are calculated and used as the upper bounds for negative samples' logits. The margin ranking loss $\mathcal{L}_\eta$ is defined as

$$\begin{aligned} \mathcal{L}_\eta(\hat{\mathbf{Y}}, \mathbf{Y}) &= \frac{1}{N_{neg}} \sum_{p,i,j} \max(\sigma(\hat{\mathbf{Y}}^{p,i,j}) - \eta^p),\ 0), \\ \eta^p &= \min\{\sigma(\hat{\mathbf{Y}}^{p,i,j}) \mid \mathbf{Y}^{p,i,j} = 1,\ \forall\ i, j\}, \end{aligned} \tag{12}$$

where N_{neg} is the number of negative samples, and η^p is the margin for predicate p between positive and negative samples. To ensure a normalized effect across different scenes (images), the margins are computed on a per-image basis. By employing this loss, unannotated triplets are neither excessively penalized, which could lead to training ambiguity, nor overly encouraged, which could result in lower ranks for positive triplets.

For sampling useful subject and object rep-points, margin ranking losses are applied to the subject and object offset means $\boldsymbol{\mu}_s^l$ and $\boldsymbol{\mu}_o^l$, for $l = 1, ..., L$. For the rep-point coordinates reparameterized by the mean, their margins are the top-left and bottom-right corner coordinates of union bounding boxes of triplets involving the entity. Consequently, the loss is nonzero when a mean rep-point is outside of a corresponding union bounding box.

3.5.2 Performance-Guided Logit Adjustment

Building upon the concept of the logit adjustment (LA) [27], we introduce the run-time performance-guided logit adjustment (PGLA) as a novel approach to enhance the performance on long-tail problems. In logit adjusted softmax cross-entropy [27], the logits from a classifier are added with a bias term $\log \boldsymbol{\eta}$ to create pairwise margins between classes. Due to the intricacies involved in entity detection and scene graph generation, employing a fixed bias throughout the training process is suboptimal. Hence, we extend the logit adjustment to a more general form of affine transformation by adding a weight term, and leveraging the training statistics to more accurately quantify the class margins.

In the context of the scene graph generation task, recall is used to guide the strength of adjustment, and PGLA is only applied to logits of positive relations. For the pair of subject i and object j, the predicted logits are adjusted as

$$\hat{\mathbf{Y}}^{:,i,j}_{\text{PGLA}} = \begin{cases} \mathbf{W} \odot \hat{\mathbf{Y}}^{:,i,j} + \mathbf{B} & (\exists\ p)\ \mathbf{Y}^{p,i,j} = 1 \\ \hat{\mathbf{Y}}^{:,i,j} & \text{otherwise,} \end{cases} \tag{13}$$

where $\mathbf{W} \in \mathbb{R}^P$ and $\mathbf{B} \in \mathbb{R}^P$ are the weight and bias factors, respectively, and the operation denoted by ":" selects all elements along the specified dimension. By setting $\mathbf{W} = \mathbf{1}$ and $\mathbf{B} = \log \boldsymbol{\eta}$, (13) yields logit adjustment [27]. The weight and bias factors for (13) are calculated as

$$\begin{aligned} \mathbf{W} &= -\tanh(\Delta \boldsymbol{r}) + 1, \\ \mathbf{B} &= -\tanh(\Delta \boldsymbol{r} / \lambda) \cdot \log(P^{-1}) + \log \boldsymbol{\eta}, \end{aligned} \tag{14}$$

where $\Delta \boldsymbol{r} = \boldsymbol{r} - \bar{\boldsymbol{r}}$, $\boldsymbol{r}$ is the measured recall, $\bar{\boldsymbol{r}}$ is the mean of the recall, λ is a hyper-parameter set to 1 by default, and $\log(P^{-1})$ is the log probability of the uniform distribution over P predicates. The hyper-parameter λ controls the sensitivity of PGLA w.r.t. the recall differences. A smaller value of λ increases the sensitivity, and more losses are enforced for predicates with low recall. The $\tanh(\cdot)$ function limits $\Delta \boldsymbol{r}$ or $\Delta \boldsymbol{r} / \lambda$ within the range of [-1, 1], and there exist alternative functions that serve the same purpose.

The effect of the long-tailed problem on losses can be described as follows: tail predicates play the role of negative classes when training head predicates, and constantly receive losses for being classified as negatives during training; on the other hand, head predicates receive more losses for being positive and less losses for being classified as negatives during training. To address this effect, there are various considerations to be taken into account regarding $\mathbf{W}$ and $\mathbf{B}$. Overall, the impacts of $\mathbf{W}$ and $\mathbf{B}$ w.r.t. the BCE loss with adjusted logits $\hat{\mathbf{Y}}_{\text{PGLA}}$ are illustrated in Fig. 3. In the case where a predicate p achieves a relatively higher recall ($\Delta \boldsymbol{r}^p > 0$), we decrease the value of $\mathbf{W}^p$ to encourage the network to generate larger logits when p is positive ($\mathbf{Y}^{p,*,*} = 1$), and smaller logits when p is negative ($\mathbf{Y}^{p,*,*} = 0$). This adjustment of $\mathbf{W}$ tries to push the predictions towards the saturation regions of the sigmoid function so that it is easier to distinguish between positive and negative classes. Simultaneously, as the recall $\boldsymbol{r}^p$ increases relatively, $\mathbf{B}^p$ increases and less loss is assigned to predicate p, allowing us to focus on other predicates with lower recall. In addition to utilizing the recall, the bias factor per predicate will be adjusted based on its prior distribution as well. A tail predicate is assigned with a smaller bias factor, and receive larger losses when being positive and smaller loss when being negative.

Despite achieving the goal of assigning class margins dynamically to address the long-tailed problem, it is critical to note that the similarities between predicates have not been taken into account. Therefore, we introduce a training statistic named "confusion logits", denoted as $\mathbf{D} \in \mathbb{R}^{P \times P}$, which tracks pairwise predicate logit differences if the predictions are incorrect. The confusion logit between the

ground-truth predicate p and an arbitrary predicate $\hat{p}$ is computed as

$$\begin{aligned} \mathbf{D}^{p,\hat{p}} &= \underset{(p,i,j)\in\Omega}{\text{mean}}\{\boldsymbol{d}^{i,j}_{logits}\cdot\boldsymbol{d}^{p,\hat{p}}_{\eta}\} \\ \Omega &\coloneqq \{(p,i,j)\mid \mathbf{Y}^{p,i,j}=1\} \\ \boldsymbol{d}^{i,j}_{logits} &= \text{ReLU}(\hat{\mathbf{Y}}^{\hat{p},i,j}-\hat{\mathbf{Y}}^{p,i,j}) \\ \boldsymbol{d}^{p,\hat{p}}_{\eta} &= \tanh(\text{ReLU}(\log\boldsymbol{\eta}^{\hat{p}}-\log\boldsymbol{\eta}^{p})), \end{aligned} \tag{15}$$

where $\text{ReLU}(\cdot)$ [69] is used for selecting positive values only. The confusion logits for a GT predicate p are computed only with respect to incorrectly-predicted predicates ($\hat{\mathbf{Y}}^{\hat{p},i,j} > \hat{\mathbf{Y}}^{p,i,j}$) with larger priors ($\boldsymbol{d}^{p,\hat{p}}_{\eta} > 0$ if $\boldsymbol{\eta}^{\hat{p}} > \boldsymbol{\eta}^{p}$). Large value of $\mathbf{D}^{p,\hat{p}}$ means that p is often mis-classified as $\hat{p}$. The confusion logits not only quantify the effects of long-tailed data, but also the semantic similarities between predicates. For similar tail predicates like `across` and `along`, their confusion logits can be large as well. During training, per-instance PGLA is applied instead by utilizing the corresponding confusion logits of a specific ground-truth predicate p, and (13) is modified as

$$\hat{\mathbf{Y}}^{:,i,j}_{\text{PGLA}} = \begin{cases} \mathbf{W}\odot\hat{\mathbf{Y}}^{:,i,j}+\mathbf{B}+\mathbf{D}^{p,:} & (\forall\, p)\ \mathbf{Y}^{p,i,j}=1 \\ \hat{\mathbf{Y}}^{:,i,j} & \text{otherwise.} \end{cases} \tag{16}$$

Ideally, recall of each predicate should be close to the mean recall for a balanced classifier. However, head classes tend to exhibit significantly higher recall compared to tail classes due to long-tailed training data. Therefore, we evaluate the recall per predicate differently. The exponential moving average (EMA) of predicate recall per mini-batch is computed to estimate the performance change over time. For each image in the mini-batch, the histogram of ground-truth relationships per predicate is obtained as $\boldsymbol{n}$. Next, based on the top $\text{sum}(\boldsymbol{n})$ triplets and predicate priors, ranking targets $\boldsymbol{\kappa}$ are set differently per predicate. For a predicate p, the top-$\boldsymbol{\kappa}^p$ triplets is used for computing the recall, where tail predicates are assigned with smaller $\boldsymbol{\kappa}^p$ while head predicates are assigned with larger values. As an illustration, the ranking target $\boldsymbol{\kappa}^{p_0}$ for the rarest predicate p_0 is $\boldsymbol{n}^{p_0}$, where the predicate should be ranked within the top-$\boldsymbol{n}^{p_0}$ in a mini-batch. For the second rarest predicate p_1, it should be ranked within the top-$(\boldsymbol{n}^{p_0}+\boldsymbol{n}^{p_1})$, and so forth. By setting distinct ranking targets for each predicate, it forces tail predicates to rank higher than head ones. Finally, the per-predicate recall $\boldsymbol{r}$ is calculated. A EMA momentum specific to each predicate is assigned as $\boldsymbol{\rho} = 0.9999^{-\log\boldsymbol{\eta}}$. Assume the batch size is 1, the process of calculating recall at iteration t is detailed in Algorithm 1. The confusion logits $\mathbf{D}_t$ is calculated and updated per iteration via EMA as well. Consequently, the PGLA can be performed given the runtime values of $\mathbf{W}_t$, $\mathbf{B}_t$, and $\mathbf{D}_t$ in (16).

Algorithm 1 Recall Calculation at training iteration t

Input: $\hat{\mathbf{Y}}$, $\mathbf{Y}$, $\boldsymbol{r}_{t-1}$, $\boldsymbol{\eta}$
Output: $\boldsymbol{r}_t$

$\boldsymbol{r}_t \leftarrow \mathbf{0}_P$
$\boldsymbol{\nu} \leftarrow \arg\text{sort}(\boldsymbol{\eta})$ ▷ indices of sorted predicate
$\boldsymbol{n} \leftarrow \sum_{i,j}\mathbf{Y}^{:,i,j}$ ▷ No. of GTs per predicate
$\boldsymbol{\kappa} \leftarrow \text{cumsum}(\boldsymbol{n}[\boldsymbol{\nu}])$ ▷ cumulative sum of No. of GTs
for $p \leftarrow \boldsymbol{\nu}^1$ to $\boldsymbol{\nu}^P$ **do**
  $\hat{R}_{\text{top}-\boldsymbol{\kappa}^p} \leftarrow \arg\max_{\boldsymbol{\kappa}^p}(\hat{\mathbf{Y}})$ ▷ top $\boldsymbol{\kappa}^p$ triplets
  $R^p \leftarrow \text{nonzero}(\mathbf{Y}^p)$ ▷ GT triplets
  $\boldsymbol{r}^p_t \leftarrow \boldsymbol{r}^p_t + \text{match}(\hat{R}_{\text{top}-\boldsymbol{\kappa}^p}, R^p)$ ▷ accumulate matches
end for
$\boldsymbol{r}_t \leftarrow \boldsymbol{r}_t \oslash \boldsymbol{n}$ ▷ element-wise division
$\boldsymbol{r}_t \leftarrow (1-\boldsymbol{\rho})\cdot\boldsymbol{r}_t + \boldsymbol{\rho}\cdot\boldsymbol{r}_{t-1}$ ▷ EMA
return $\boldsymbol{r}_t$

4 Experiments

4.1 Datasets and Evaluation

Datasets. We evaluate our methods on the Visual Genome (VG) [24] dataset. We follow the protocols for the widely-used pre-processed subset VG150 [15], [28] which contains the most frequent 150 entities (C = 150) and 50 predicates (P = 50). The dataset contains approximately 108k images, with 70% for training and 30% for testing. We also evaluate on the Open Images V6 dataset (OIV6) [43], [67], which contains 126k training images, 5k testing images, 301 entities and 30 predicates. For OIV6, as some entity and predicate classes are absent from the testing set, we exclusively train on the classes that are actually present. Consequently, we use 212 entities and 21 predicates that remain in the training set.

Evaluation. We evaluate our methods following three standard evaluation tasks: 1) predicate classification (PredCls): predict predicates given ground-truth entity classes and bounding boxes; 2) scene graph classification (SGCls): predict predicates and entity classes given ground-truth entity bounding boxes; 3) scene graph detection (SGDet): predict predicates, entity classes, and entity bounding boxes. For VG150, we report results of recall@K (R@K) [72], mean recall@K (mR@K) [29], [73], zero-shot recall@K (zs-R@K) [72] for all the three evaluation tasks. In addition, we further evaluate zero-shot mean recall@K (zs-mR@K) to assess methods' ability to generalize to unseen long-tailed testing distributions. For OIV6, following previous works [43], [74], we report results of R@K, weighted mean AP of relationship detection (wmAP$_{rel}$), weighted mean AP of phrase detection (wmAP$_{phr}$), and the weighted score as score$_{wtd}$=0.2 × R@50 + 0.4 × wmAP$_{rel}$ + 0.4 × wmAP$_{phr}$. Considering the down-weighting effect of these metrics on tailed predicates, we also provide mean recall@K results.

4.2 Implementation Details

ResNet-50 [61] is used as the backbone network and the same hyper-parameters are used following [14], [75]. The entity detector is initialized with the weights pre-trained on COCO dataset [76], [77]. Specifically, the pre-trained weights are trained for 90k iterations with a batch size of 16, an initial learning rate of 0.01 which is decreased at the 60k-th and 80k-th iteration by a factor of 0.1 sequentially, and the weight decay of 0.0001. Images are resized such that their shorter edge is sampled from [640, 800] with a step of 32, and their longer edge does not exceed 1333 pixels. Random horizontal flip with a probability of 0.5 and random crop for are used for data augmentations. Specifically, a relative random ratio is selected from [0.9, 1] to crop along each axis respectively. To train RepSGG, the same multi-scale

TABLE 1
Comparisons of R@K and mR@K results on VG150 between the proposed methods and SOTA methods. Methods are grouped from top to bottom as: point-based, query-based, and box-based methods. FCSGG [18] uses HRNet [70] as backbone, and CoRF [19] uses Swin-S [71]. The best results are bold, and the second-best results are underlined.

	Predicate Classification						Scene Graph Classification						Scene Graph Detection					
	R@20/50/100			mR@20/50/100			R@20/50/100			mR@20/50/100			R@20/50/100			mR@20/50/100		
FCSGG [18]	33.4	41.0	45.0	4.9	6.3	7.1	19.0	23.5	25.7	2.9	3.7	4.1	16.1	21.3	25.1	2.7	3.6	4.2
CoRF [19]	-	45.4	-	-	10.1	-	-	18.7	-	-	3.9	-	-	18.6	-	-	3.9	-
RelTR [22] ‡	**63.1**	64.2	-	20.0	21.2	-	29.0	36.6	-	7.7	11.4	-	21.2	27.5	-	6.8	10.8	-
TraCQ [21] ‡	-	-	-	-	-	-	-	-	-	-	-	-	19.7	28.3	35.7	12.0	13.8	14.6
SGTR [20] §	-	-	-	-	-	-	-	-	-	-	-	-	-	24.6	28.4	-	12.0	15.2
SGTR [20], [43] §*	-	-	-	-	-	-	-	-	-	-	-	-	-	20.6	25.0	-	15.8	20.1
VCTree [29] ‖	60.1	**66.4**	**68.1**	-	-	-	**35.2**	38.1	38.8	-	-	-	22.0	27.9	31.3	-	-	-
BGNN [43] †*	-	59.2	61.3	-	30.4	32.9	-	37.4	38.5	-	14.3	16.5	-	31.0	35.8	-	10.7	12.6
PPDL [56] †*	-	41.6	43.6	-	33.3	36.2	-	24.8	26.2	-	20.2	22.0	-	13.6	16.5	-	12.2	14.4
PCPL [54] ‖*	-	50.8	52.6	-	35.2	37.8	-	27.6	28.4	-	18.6	19.6	-	14.6	18.6	-	9.5	11.7
RTPB [60] †*	-	45.6	47.5	30.3	36.2	38.1	-	24.5	25.5	19.1	21.8	22.8	-	19.7	23.4	12.7	16.5	19.0
DT2-ACBS [44] §*	-	23.3	25.6	27.4	35.9	39.7	-	16.2	17.6	18.7	24.8	27.5	-	15.0	16.3	**16.7**	**22.0**	**24.4**
IETrans [46] †*	-	48.0	49.9	-	37.0	39.7	-	30.0	30.9	-	19.9	21.8	-	23.6	27.8	-	12.0	14.9
FGPL [57] †*	-	-	-	**30.8**	37.5	40.2	-	-	-	**21.9**	**26.2**	27.6	-	-	-	11.9	16.2	19.1
RepSGG ‡	55.2	62.7	65.0	16.8	22.2	24.4	34.7	**44.0**	**49.9**	10.5	14.5	17.3	**23.6**	**31.1**	**36.3**	7.2	10.0	12.3
RepSGG$_{\text{PGLA, }\lambda=0.1}$ ‡*	40.3	46.7	48.7	23.1	29.8	33.1	23.5	30.6	35.0	12.6	17.5	21.5	16.1	21.8	26.0	9.4	13.1	16.1
RepSGG$_{\text{PGLA}}$ ‡*	24.3	27.8	28.8	29.2	**39.7**	**43.7**	13.8	17.9	20.3	16.2	22.3	**27.7**	8.9	12.2	14.6	10.9	15.4	18.7

Backbone network: † ResNeXt-101-FPN § ResNet-101 ‖ VGG-16 ‡ ResNet-50 * debiasing technique is used

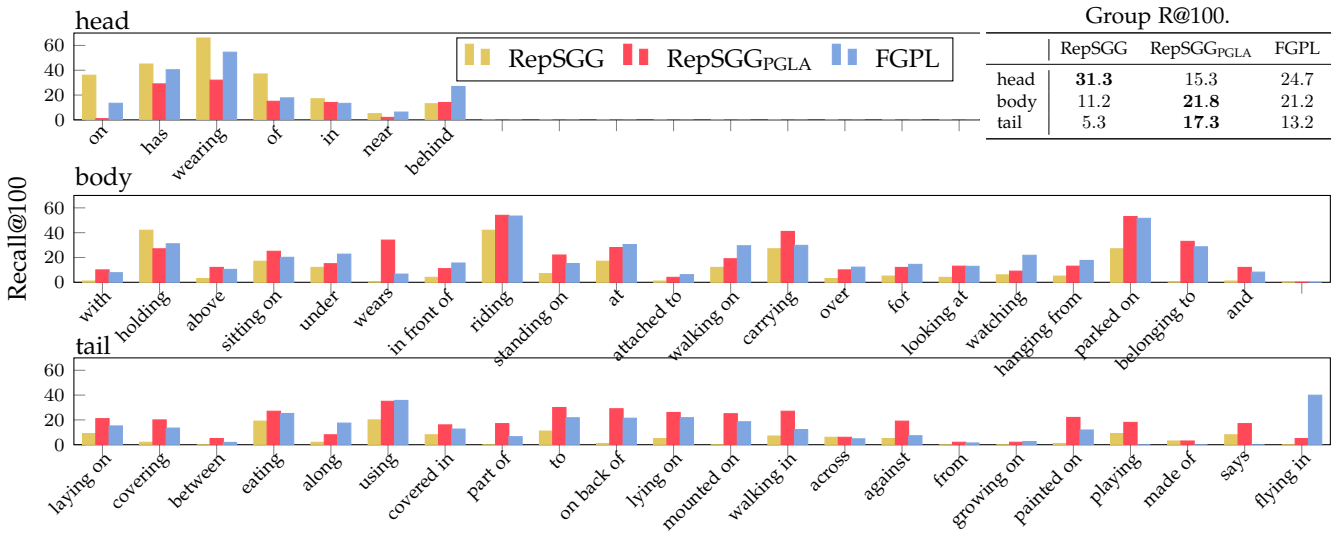
training is adopted following FCOS [14], except that we set the shorter edge range as [480, 800], and random crop ratio range as [0.8, 1]. The repeat factor sampling [78] with the factor of 0.02 is applied to sample more images that contain tail predicates. We first train the FCOS detector on VG150 or OIV6 for 90k iterations. The whole architecture is then trained for additional 90k iterations while freezing the backbone and entity detection heads. Finally, the entire model is jointly trained for 10k iterations, and this procedure is referred to as fine-tuning throughout the rest of the paper. Training is performed on 4 Nvidia A100 GPUs with a batch size of 32. The AdamW [79] optimizer is used with a initial learning rate of 10^{-5} which is decayed at the 80k-th iteration by a factor of 0.1, and the weight decay of 10^{-4}. The learning rates of the backbone and rep-point samplers are multiplied by a factor of 0.1. A single model is trained for all tasks, rather than separate models for each task. For the encoder, L_e is set to 1 with the same hyper-parameter setting as used in [26]. The relationship encoder is configured with $L_d = 1$, $K = 4$, $h_G = h_R = 8$, $h_A = 128$, $d_G = d_R = 32$, and $d_A = 64$ as the default settings. For the rep-point samplers, m is uniformly sampled from [1, 100].

4.3 Quantitative Results

4.3.1 Visual Genome

To compare with methods using different entity representations and those using debiasing techniques respectively, we train one model without PGLA (RepSGG) and another with PGLA (RepSGG$_{\text{PGLA}}$). R@K is the main metric when comparing methods without debiasing, while mR@K is the main one for debiasing methods. In addition, since RepSGG and RepSGG$_{\text{PGLA}}$ are two extreme cases of using debiasing methods, we also add a 3rd model RepSGG$_{\text{PGLA, }\lambda=0.1}$ with balanced R@K and mR@K results for comparison. We compare our methods with the state-of-the-art (SOTA) scene graph generation models as shown in Table 1. We divide the methods for comparison into 3 groups in Table 1 from top to bottom: point-based, query-based, and box-based, regardless of whether long-tailed techniques are used. Notably, point-based and query-based methods seek fast inference speed, and often do not involve debiasing techniques. Query-based methods inherently possess a certain level of debiasing capability due to the direct triplet prediction design. However, they are typically limited to performing only scene graph detection task. Box-based methods focus on designing debiasing methods with off-the-shelf entity detectors.

First of all, RepSGG and RepSGG$_{\text{PGLA}}$ attain state-of-the-art performance across 8 out of 18 metrics, surpassing other methods (VCTree, DT2-ACBS, FGPL) that achieve results over only 3 metrics. Solely comparing methods that have different entity or predicate representations, RepSGG outperforms point-based methods FCSGG [18] and CoRF [19] on all metrics by a large margin. Comparing with query-based methods, RepSGG outperforms RelTR [22], TraCQ [21], and SGTR [20] on most R@K metrics across all 3 tasks, while RepSGG$_{\text{PGLA}}$ also outperforms RelTR and TraCQ on mR@K metrics. As query-based methods like SGTR and TraCQ directly predict triplets consisting of entity and predicate predictions, they typically perform better on the SGDet task. Nevertheless, RepSGG$_{\text{PGLA, }\lambda=0.1}$ achieves comparable performance on R@K and mR@K w.r.t. SGTR, with higher mR@K and slightly lower R@K. Under the condition of no debiasing, the performance improvements on R@K for RepSGG indicates that the proposed entity and predicate representations are superior to point-based and query-based methods. Compared with box-based VCTree [29], RepSGG achieves higher R@K on SGCls and SGDet tasks with slightly lower recall on PredCls. In terms of debiasing methods, RepSGG$_{\text{PGLA}}$ outperforms the SOTA



Group R@100.

	RepSGG	RepSGG$_{PGLA}$	FGPL
head	**31.3**	15.3	24.7
body	11.2	**21.8**	21.2
tail	5.3	**17.3**	13.2

Fig. 4. Per-predicate SGDet R@100 comparison between RepSGG, RepSGG$_{PGLA}$, and FGPL on VG150 dataset. RepSGG$_{PGLA}$ performs better on body and tail groups. The overall standard deviation of R@100 is 14.6 (RepSGG), 12.3 (RepSGG$_{PGLA}$), and 13.6 (FGPL) respectively, which also implies that RepSGG$_{PGLA}$ achieves a more balanced performance.

methods on PredCls mR@50, PredCls mR@100, and SGCls mR@100, while achieving comparable results on other metrics. RepSGG achieves 43.7 mR@100 on PredCls, which is 3.5 higher than the box-based FGPL [57]. FGPL is a complicated long-tail learning technique which requires a biased model with several fine-tuned hyper-parameters, while PGLA is much simpler yet effective for mitigating the long-tailed problem.

We further compare the per-predicate and group R@100 with FGPL for the SGDet task as shown in Fig. 4. Predicates are sorted in descending order based on their frequency in the training set, and divided into 3 groups following [43]. RepSGG$_{PGLA}$ achieves higher recall on body and tail classes, resulting in a more balanced performance over FGPL. In VG150 testing set, there are only 29, 37, and 12 triplets involving `playing`, `made of`, and `says`, respectively. While FGPL failed retrieving these triplets, our method achieves significantly better results, even thought these triplets are extremely rare both during training and testing. Moreover, we observe considerate improvements on fine-grained predicates with similar semantics. RepSGG$_{PGLA}$ achieves better recall over predicates `sitting on`, `standing on`, `parked on`, `laying on`, `lying on`, and `painted on`. It reveals the discriminatory capability of our model among hard-to-distinguish predicates. We conjecture that the reppoint samplers and the GCA layers capture more visual context compared with box-based methods. Furthermore, by comparing the results of RepSGG$_{PGLA}$ and FGPL on a pair of visually indistinguishable predicates, `wearing` and `wears`, it becomes evident that FGPL still exhibits the bias towards the more frequent predicate `wearing`, resulting in significantly lower performance on `wears`. In contrast, RepSGG$_{PGLA}$ achieves comparable results on `wearing` and `wears`, indicating that the proposed PGLA strategy offers a more effective and balanced learning process. Notably, without debiasing techniques, RepSGG achieves excellent performance on those tail predicates (`across`, `playing`, `made of`, and `says`) as well. It demonstrates that the proposed entity and relationship representations inherently capture more informative semantics.

TABLE 2
PredCls results of zero-shot mean recall (zs-mR@K) and zero-shot recall (zs-R@K) on VG150 compared to state-of-the-art methods. The best results are bold, and the second-best results are underlined.

	PredCls Zero Shot Relationship Retrieval mR@20/50/100			R@20/50/100		
BGNN [43]	1.9	3.2	4.9	2.0	3.5	4.6
BA-SGG [49]	3.1	5.3	6.7	3.0	6.0	8.0
Motifs-TDE [16]	5.3	9.3	11.4	8.3	14.3	**18.0**
FGPL [57]	**11.0**	14.3	15.9	**9.4**	13.0	14.6
IETrans [46]	**11.0**	**14.5**	17.0	6.5	10.0	12.0
RepSGG	4.1	7.1	8.9	8.9	**14.6**	**18.0**
RepSGG$_{PGLA}$	9.9	**17.2**	**20.0**	6.1	9.2	11.1

We further conduct analysis on zero-shot performance. In this setting, the objective is to retrieve triplets that are not encountered during training, but are present during testing. The zero-shot performance in scene graph generation is essential for achieving generalizability, robustness, adaptability, and cost-effectiveness in real-world applications, while also serving as a key metric for model evaluation and benchmarking. In Table 2, we report the zero-shot recall and zero-shot mean recall results on the PredCls task and compare with the SOTA methods. We collect the results by implementing the zs-mR@K evaluation following [16]. RepSGG$_{PGLA}$ outperform the SOTA methods on mR@50 and mR@100 by a large margin. Among methods for comparison, IETrans [46] achieves good results by relabeling predictions and labeling unannotated samples from biased models for training. Without extra data for training, RepSGG$_{PGLA}$ outperforms IETrans by 3.0 on zs-mR@100. RepSGG also achieves the SOTA zero-shot recall performance over zs-R@50 and zs-R@100. Motifs-TDE employs a debiasing technique to achieve a zs-R@100 of 18.0, whereas

TABLE 3
Comparisons with the state-of-the-art methods on OI V6. R@50 in the table is micro-Recall@50 [80]. The best results are bold, and the second-best results are underlined.

	mR@50	R@50	wmAP_{rel}	wmAP_{phr}	score_{wtd}
Motifs [28]	32.68	71.63	29.91	31.59	38.93
RelDN [74]	33.98	73.08	32.16	33.39	40.84
VCTree [29]	33.91	74.08	34.16	33.11	40.21
G-RCNN [81]	34.04	74.51	33.15	34.21	41.84
GPS-Net [53]	35.26	74.81	32.85	33.98	41.69
BGNN [43]	40.45	74.98	33.51	34.15	42.06
SGTR [20]	42.61	59.91	38.73	36.98	42.28
CSL [82]	41.72	75.44	34.30	35.38	42.86
SS R-CNN [37]	50.73	75.70	**41.14**	**43.24**	**48.89**
RepSGG	62.68	77.70	30.01	29.58	39.38
$\text{RepSGG}_{\text{PGLA/R}}$	**64.26**	76.40	31.64	31.27	40.44
$\text{RepSGG}_{\text{PGLA/P}}$	53.87	70.25	32.53	32.47	40.05
$\text{RepSGG}_{\text{X101}}$	56.32	**77.83**	36.73	36.61	44.90

RepSGG achieves the same results without using debiasing techniques. It demonstrates that RepSGG generalizes significantly better to compositions of entities and relationships in unseen contexts.

4.3.2 Open Images

Since the precision is one of the evaluation metrics for OIV6, we conduct experiments using precision as the PGLA metric besides recall. The model trained with precision-guided logit adjustment is denoted as $\text{RepSGG}_{\text{PGLA/P}}$, and the model trained with recall-guided logit adjustment is renamed to $\text{RepSGG}_{\text{PGLA/R}}$. We also train a model with the ResNeXt-101-32×8d [83] backbone, denoted by $\text{RepSGG}_{\text{X101}}$ without using PGLA. The experimental results on OIV6 [67] are shown in Table 3. We observe that all RepSGG models outperforms other methods on mR@50 by a large margin. $\text{RepSGG}_{\text{PGLA/R}}$ achieves a mR@50 of 64.26, marking a 13.53-point increase, or a 26.7% improvement over the previous SOTA method SS R-CNN [37]. $\text{RepSGG}_{\text{PGLA/P}}$ achieves better results on wmAP_{rel} and wmAP_{phr} in comparison to RepSGG and $\text{RepSGG}_{\text{PGLA/R}}$, highlighting the effectiveness of precision-guided PGLA for precision-oriented tasks. As a compromise, the recall performance is lower compared with PGLA/R. With a larger backbone network, $\text{RepSGG}_{\text{X101}}$ achieves the highest R@50 performance of 77.83, and the second-best score_{wtd} of 44.90, which is a 2.62-point improvement over SGTR. Our methods achieves lower precision-oriented metrics like wmAP_{rel} and wmAP_{phr}. This is because OIV6 has very sparse annotations, while our model has great generalization power as shown in Table 2. OIV6 has 2.76 relationship annotations per image on average in the training set, while VG150 has 5.97. As a result, most detections will be considered as false positives, leading to lower precision. We then collect the results on other SGG tasks to further analyze the performance as shown in Table 4. The results on R@50, mR@50, and wmAP_{rel} show significant improvements on PredCls and SGCls tasks. In the PredCls setting, all RepSGG models achieve R@50 and wmAP_{rel} over 90. Both Table 1 and 3 demonstrate that our model's primary bottleneck lies in entity detection rather than predicate prediction. Consequently, there are fewer improvements observed in SGDet and SGCls tasks compared to PredCls.

TABLE 4
Comparisons of RepSGG models without PGLA, with recall-guided LA, and with precision-guided LA on PredCls and SGCls tasks.

	PredCls			SGCls		
	mR@50	R@50	wmAP_{rel}	mR@50	R@50	wmAP_{rel}
RepSGG	69.40	97.33	93.19	66.59	**90.56**	54.21
$\text{RepSGG}_{\text{PGLA/R}}$	**80.33**	96.69	89.08	**71.95**	90.06	**55.46**
$\text{RepSGG}_{\text{PGLA/P}}$	73.44	95.36	92.51	55.09	74.99	51.30
$\text{RepSGG}_{\text{X101}}$	66.97	**97.48**	**93.60**	63.04	85.81	52.09

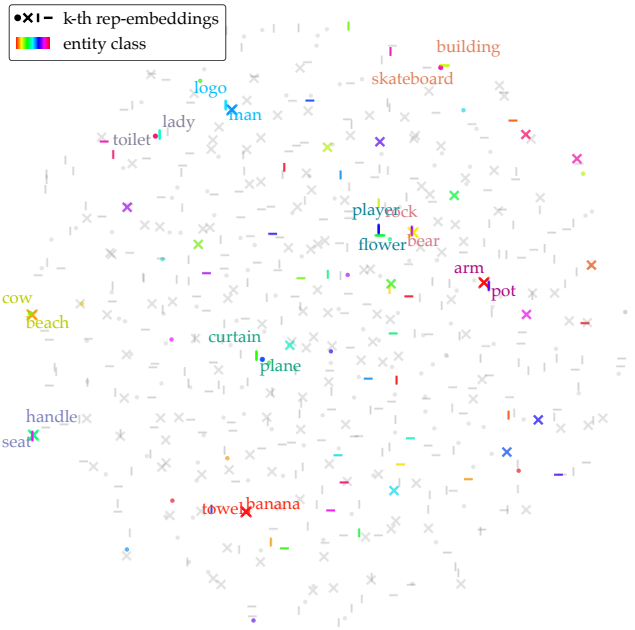


Fig. 5. The t-SNE visualization of subject rep-embeddings $\mathbf{E}_s$, projected to $\mathbb{R}^{C\times K\times 2}$ with pairwise cosine similarity. There are $C\times K = 150\times 4 = 600$ points in total, and each point represents a subject rep-embedding in the projected 2D space. The top-10 similar pairs are labeled. Rep-embeddings of the same entity class share a color. Only the entity classes involved in the top-10 pairs are colored, while the others are displayed in gray.

4.4 Qualitative Analysis

We want to qualitatively examine what the model learns from the data, especially on the rep-embeddings and rep-points. We visualize the weights of subject rep-embeddings $\mathbf{E}_s$ of the trained $\text{RepSGG}_{\text{PGLA}}$ (with $K = 4$, $L_e = 1$, and $L_d = 1$) via t-SNE [84] as shown in Fig. 5. The top 10 pairs of similar rep-embeddings are highlighted, while the remaining rep-embeddings belonging to the involved entities are colored. We use the pairwise cosine similarity as the distance metric for t-SNE where distances represent the similarities between rep-embeddings. It reveals that the rep-embeddings between different entity classes are well separated. Rep-embeddings of the same entity class are distinctly separated as well. Interestingly, most pairs do not exhibit explicit semantic part affinities. We hypothesize that the initial rep-embeddings serve as "anchors" which are evenly distributed in the embedding space, and capture more semantic information as they progress through the relationship encoder.

We further validate the hypothesis by visualizing the

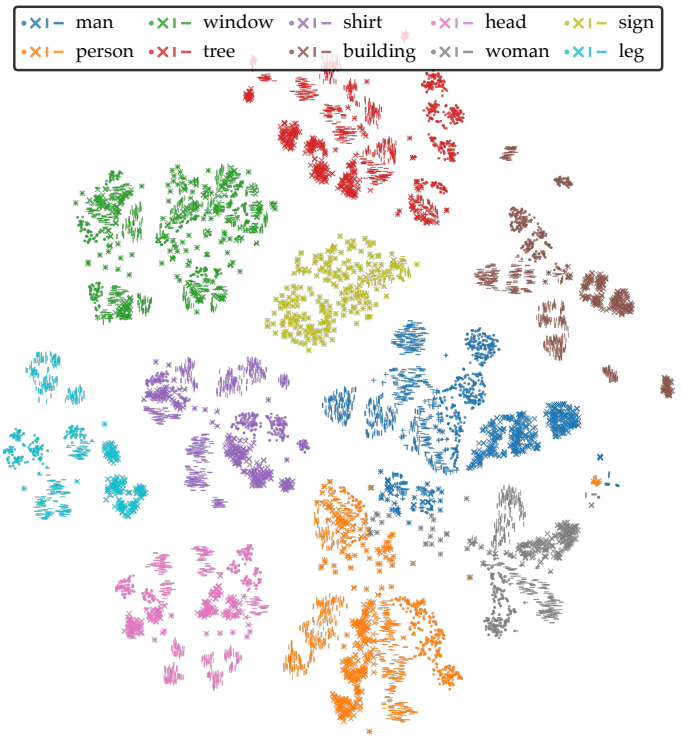


Fig. 6. The t-SNE visualization results on the output subject queries of the relationship encoder ($\mathbf{Q}^{L_d}$) for 10 frequent entity classes.

output queries $\mathbf{Q}^{L_d}$. To eliminate entity mis-classification, we collect the output queries on the PredCls task from the first 1000 testing images. As shown in Fig. 6, the inter-class rep-embeddings are distinguishable. Entities with similar semantics, such as `woman`, `man`, and `person`, form more compact clusters. The compactness and distinctness of rep-embeddings differ within an entity class. For `building`, 4 types of rep-embeddings are well-separated with high variance where each type is responsible for a different semantic concept. For `sign`, the distribution of rep-embeddings is more compact, indicating that the semantics and relationships associated with `sign` are mostly homogeneous.

We also collect the subject and object rep-point offset parameters $\boldsymbol{\mu}_s^1$, $\boldsymbol{\sigma}_s^1$, $\boldsymbol{\mu}_o^1$, and $\boldsymbol{\sigma}_o^1$, which are shown in Fig. 7. The variation in parameters is evident, primarily between different entity classes, and between subject and object offsets. The distribution of parameters implies the locations of relevant semantic features to a certain extent. It is likely that the spatial means follow a bivariate Gaussian distribution especially for `man` and `sign`, which suggests that rep-points are sampled around the bounding box centers. Entities with larger location variances within bounding boxes, such as `banana`, `bird`, and `railing`, exhibit more dispersed distributions of spatial mean offsets. The distributions of subject and object mean offsets for `man` are noticeably distinct. The subject means are more tightly clustered around centers, indicating that when `man` is a subject, the features representing the entirety of a `man` are more significant. Conversely, when `man` is an object, the features representing semantic parts become more important as object means are much more distributed. The scale mean offsets also reveal different patterns among entities. For entities with large bounding boxes like `room`, the scale means are primarily negative, indicating that the sampled rep-points are from lower scales of features. For entities in medium and small sizes, the sampled rep-points are mostly from the same or higher scales of features. In terms of the standard deviations (stds) of offsets, we observe the spatial collinearity, which is a natural occurrence in training data with diverse entity sizes. It is notable that `railing` has larger spatial and scale stds, reflecting the uncertainty associated with the varying lengths of railings. The scale stds normally remain within a narrower range (0.07 to 0.09) compared to the spatial stds (0.103 to 0.115).

TABLE 5
Ablation studies of number of rep-embeddings K, number of encoder layers L_e, and number of decoder layers L_d in RepSGG$_{\text{PGLA}}$. Results on mR@100 and zs-mR@100 are collected for three SGG tasks.

K	L_e	L_d	PredCls @100		SGCls @100		SGDet @100	
			mR	zs-mR	mR	zs-mR	mR	zs-mR
1	1	1	**42.5**	18.5	20.0	6.5	15.7	5.5
4	1	1	42.0	**20.6**	19.9	6.0	**15.9**	5.4
7	1	1	41.1	18.8	19.6	6.2	15.7	5.6
10	1	1	40.7	20.0	19.1	6.6	15.4	**5.9**
4	0	0	39.1	16.2	18.0	5.9	13.8	5.0
4	1	0	39.8	18.2	19.1	6.3	15.0	5.1
4	0	1	41.0	16.8	19.4	6.1	15.7	5.5
4	2	2	40.7	18.2	19.5	5.9	15.2	5.4
10	3	3	41.0	19.6	**20.0**	**7.1**	15.1	5.5

TABLE 6
Ablation studies of GCA and RCA.

GCA	RCA	PredCls @100		SGCls @100		SGDet @100	
		mR	zs-mR	mR	zs-mR	mR	zs-mR
		39.8	18.2	19.1	6.3	15.0	5.1
✓		40.3	17.3	19.0	5.5	14.4	4.8
	✓	41.8	18.7	**19.9**	**6.6**	15.7	**5.7**
✓	✓	**42.0**	**20.6**	**19.9**	6.0	**15.9**	5.4

4.5 Ablation Studies

To further investigate the proposed methods, we perform ablation studies on VG150 dataset. Unless specified, we use the same hyper-parameters as discussed in Section 4.2, and PGLA is applied but fine-tuning is not applied.

4.5.1 *Analysis of RepSGG*

We investigate the effects of different numbers of rep-embeddings, deformable encoder layers, and relationship encoder layers. The results on mR@100 and zs-mR@100 are listed in Table 5. First, adding the deformable encoder or relationship encoder improves the overall performance. It is also important to note that similar behaviors can be observed when increasing the values of K, L_e, or L_d, as the performance improves and then plateaus at certain values of K, L_e, or L_d, respectively. The combination of $K = 4$, $L_e = 1$, and $L_d = 1$ achieves a balanced trade-off between performance and inference speed. Based on our experiments, four rep-embeddings per entity class ($K = 4$) are sufficient to capture possible relationships, and using $K = 10$ potentially causes overfitting on few re-embeddings. Likewise,

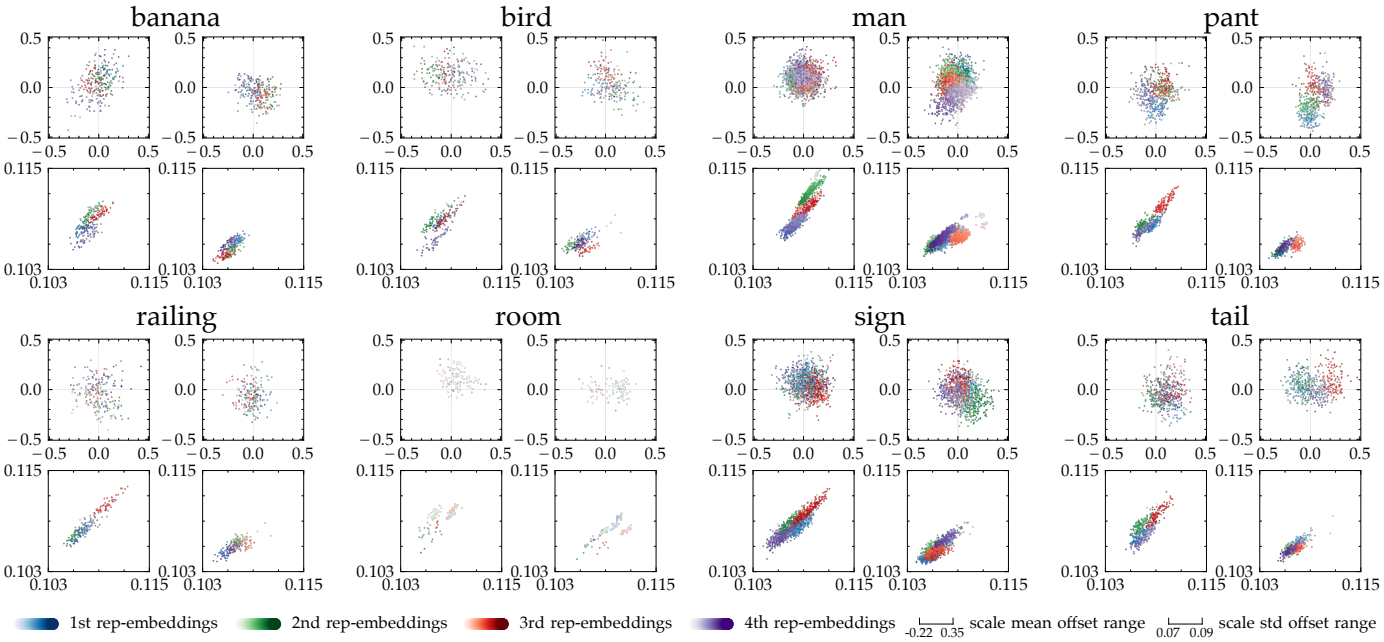


Fig. 7. Visualizations on predicted subject and object rep-point mean and std offsets. For each entity class, the offsets of subject means, object means, subject stds, and object stds are shown on the top-left, top-right, bottom-left, and bottom-right of the sub-figure. The coordinates represent spatial offsets w.r.t. the ground-truth centers, while the color saturation denotes the scale offsets w.r.t. the entity feature scale. The more saturated the color is, the larger the scale offset is, and vice versa.

TABLE 7
Ablation studies on using separate subject and object rep-embeddings ($\mathbf{E}_s \neq \mathbf{E}_o$) vs. identical rep-embeddings ($\mathbf{E}_s = \mathbf{E}_o$).

L_e	L_d	Separate	PredCls @100		SGCls @100		SGDet @100	
			mR	zs-mR	mR	zs-mR	mR	zs-mR
0	0		39.1	15.5	**18.2**	5.5	13.8	**5.1**
0	0	✓	39.1	**16.2**	18.0	**5.9**	13.8	5.0
1	1		40.2	17.0	19.1	4.6	14.4	5.0
1	1	✓	**42.0**	**20.6**	**19.9**	**6.0**	**15.9**	**5.4**

increasing the number of deformable encoder or relationship encoder layers does not lead to significant performance improvement. When both deformable encoder and relationship encoder are removed, ReSGG$_{\text{PGLA}}$ still achieves a high PredCls mR@100 of 39.3, which outperforms many state-of-the-art box-based methods listed in Table 1. This suggests the advantage of our proposed relationship representation over traditional softmax classification. Treating relationships as attention weights naturally incorporates more semantic information, effectively capturing the distinction between subject and object entities.

We further conduct the analysis of GCA and RCA modules in the relationship encoder. We use the model with $K = 4$, $L_e = 1$, and $L_d = 1$. The model without either GCA or RCA is equivalent to the one with hyper-parameters $K = 4$, $L_e = 1$, and $L_d = 0$ as shown in Table 5. We further train 3 separate models, one with GCA only, one with RCA only, and one with both modules. When GCA is not equipped, we use the visual features sampled at entity centers as the inputs to RCA, without using rep-point samplers. The ablation results are shown in Table 6. No significant performance drop is observed when removing GCA or RCA. RCA is more important than GCA based on the results, as the RCA-only model achieves better performance on most metrics. We conjecture that RCA exchanges semantic information among rep-embeddings, while GCA only exchanges visual features locally. When both GCA and RCA are applied, we obtain the best results on most metrics except on SGDet zs-mR@100.

We also investigate the effectiveness of using separate rep-embeddings to represent subjects and objects. Specifically, we keep $K = 4$ for this experiment, and conduct 2 groups of ablation studies, with either $L_e = L_d = 0$ or $L_e = L_d = 1$, and train with separate rep-embeddings ($\mathbf{E}_s \neq \mathbf{E}_o$) or identical rep-embeddings ($\mathbf{E}_s = \mathbf{E}_o$). The results are shown in Table 7. In the absence of both the entity encoder and relationship encoder, there is minimal performance difference observed between the two settings. It is difficult for the model to predict relationships without the entity encoder and relationship encoder, which are responsible for capturing more visual and semantic context. However, upon adding a single layer of both the entity encoder and relationship encoder, significant performance improvements are observed across all metrics and tasks. This clearly illustrates the benefits of using separate rep-embeddings for subjects and objects in distinguishing semantics in relationship inference. We hypothesize that predicting certain relationships becomes challenging for the model when it lacks information about which entities serve as subjects or objects.

4.5.2 *Analysis of PGLA*

To investigate the effects different loss-related configurations proposed in the paper, we conduct the ablation experiments on Logit Adjustment [27], the proposed PGLA, the margin ranking loss $\mathcal{L}_\eta$ proposed in Section 3.5.1, and

TABLE 8
Ablation studies of loss and training configurations.

LA	PGLA	$\mathcal{L}_\eta$	finetune	PredCls @100 mR	PredCls @100 zs-mR	SGCls @100 mR	SGCls @100 zs-mR	SGDet @100 mR	SGDet @100 zs-mR
				27.1	9.7	15.9	5.1	11.4	4.1
✓				39.4	16.2	19.4	5.5	13.7	4.4
	✓			40.3	18.4	21.4	5.8	15.0	5.1
	✓	✓		41.2	**20.1**	21.4	6.6	15.3	5.6
	✓	✓	✓	**43.7**	**20.1**	**27.7**	**7.1**	**18.7**	**5.9**

TABLE 9
Effects of $\mathbf{W}$, $\mathbf{B}$, and $\mathbf{D}$ in PGLA.

$\mathbf{W}$	$\mathbf{B}$	$\mathbf{D}$	PredCls @100 mR	PredCls @100 zs-mR	SGCls @100 mR	SGCls @100 zs-mR	SGDet @100 mR	SGDet @100 zs-mR
			27.1	9.7	15.9	5.0	11.4	4.3
✓			28.5	9.4	17.4	5.4	12.8	4.5
	✓		**42.0**	17.7	21.3	6.5	15.2	5.8
		✓	30.5	10.0	18.8	5.7	13.6	4.8
	✓	✓	40.5	16.8	20.5	6.9	15.3	5.7
✓		✓	30.5	10.2	17.8	5.9	13.5	4.7
✓	✓		41.6	18.7	**21.4**	7.0	**16.0**	**5.9**
✓	✓	✓	41.2	**20.1**	**21.4**	**7.1**	15.3	5.8

fine-tuning. As shown in Table 8, RepSGG$_{\text{PGLA}}$ achieves higher mean recall on all metrics compared with RepSGG trained with LA, which confirms the effectiveness of PGLA. RepSGG$_{\text{PGLA}}$ also achieves a more significant improvement on zs-mR@100, providing evidence that PGLA is more resilient to under-fitting or over-fitting. Using $\mathcal{L}_\eta$ with PGLA brings more improvements, as it effectively suppresses unlikely relationships and avoids excessively penalizing potentially unannotated ones. Finally, with additional fine-tuning the entire model, we manage to further increase the performance on mean recall.

We study the individual and combinatorial effects of $\mathbf{W}$, $\mathbf{B}$, and $\mathbf{D}$ in (16). We simply set $\mathbf{W} = \mathbf{1}$ when $\mathbf{W}$ is not applied, and $\mathbf{B}$ or $\mathbf{D}$ to zero when they are not applied. As shown in Table 9, introducing either component improves the performance on mean recall. The bias term $\mathbf{B}$ has the most significant effect over the results with the largest improvement compared with individual application of $\mathbf{W}$ or $\mathbf{D}$. This can be attributed to the fact that class margins are primarily determined by the bias. Similarly, $\mathbf{D}$ serves as an additional adjustment for class margins, which has more effect over $\mathbf{W}$ as a result. Applying $\mathbf{B}$ alone already yields the highest mR, but considering zs-mR is also crucial for real applications of scene graphs. Combing $\mathbf{W}$ and $\mathbf{B}$ enhances the zs-mR performance, and combing all 3 components further improves the zs-mR performance with a slightly decrease on mR.

The hyper-parameter λ in (14) allows for adjusting the sensitivity to run-time recall, and we explore its effects on recall and mean recall. We evaluate on all 3 SGG tasks with different values of λ, and the results are shown in Fig. 8. As anticipated, increasing λ results in higher mR@100 but lower R@100, whereas decreasing λ leads to lower mR@100 but higher R@100. Although better trade-offs may exist, exploring them falls beyond the scope of this paper.

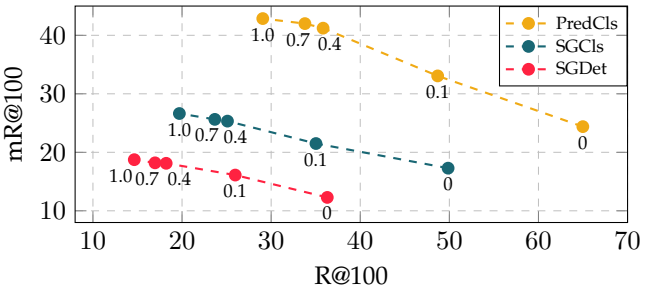


Fig. 8. Effects of λ in (14) on R@100 and mR@100. The value of λ used for each model is annotated, where $\lambda = 0$ denotes "PGLA is not applied", and $\lambda = 1$ denotes the default PGLA.

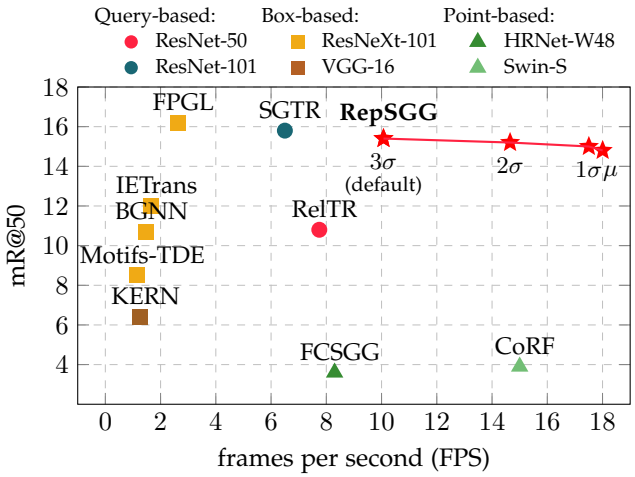


Fig. 9. Inference speed and mR@50 benchmark on the SGDet task.

4.5.3 *Analysis of Inference Speed*

Beside performance improvements, another set of benefits of RepSGG are its fast inference speed and flexible inference configurations. In Section 3.3.2, the details of inference are discussed where we sample rep-points within "3σ". We have the option to perform inference with fewer samples by changing the hyper-parameter ξ during inference. We conduct experiments using samples within 3σ, 2σ, σ, and just the samples at the means μ. We also benchmark several methods discussed in Section 2 as comparisons. We measure the average of frames per second (FPS) over all testing images for all selected models, which is evaluated on a single Nvidia GTX 1080 Ti GPU with a batch size of 1. The settings from different models follow their original papers and implementations, so the inference speed depends on not only the architectures, but also inference configurations such as image size, mixed precision, *etc*.

The results of mR@50 and FPS are collected on the SGDet task as shown in Fig. 9. Remarkably, the proposed methods achieve better trade-offs between performance and speed. No crucial performance drop is observed when less rep-points are sampled. By sampling within 3σ, our model achieves competitive results compared to the state-of-the-art box-based FPGL and query-based SGTR methods, while being approximately 4.1× and 1.6× faster, with only 4.9% and 2.5% mR@50 performance drop, respectively. By only sampling at the means, RepSGG$_{\text{PGLA}}$ achieves 14.8 mR@50

with a nearly real-time inference speed of 18 FPS, a speedup of approximately 6.8× and 2.7×, with only 8.6% and 6.3% mR@50 performance drop, compared to FPGL and SGTR, respectively.

5 Limitations and Future Work

RepSGG excels on PredCls and SGCls tasks, showcasing an advantage over query-based methods [20], [21], [37]. Nevertheless, its performance on SGDet does not outperform that of the state-of-the-art methods, because it does not fully exploit the ground-truth information as effectively as box-based methods. Giving the GT bounding boxes, we must map the corner coordinates to appropriate feature levels and identify the best-matched pixels responsible for the detections. However, the best-matched features do not correspond as accurately to actual ground-truth features as achieved by RoIAlign [12]. A hybrid representation of box, point, and query features could help improve the performance. Additionally, it is important to note that our model is exploratory and primarily focuses on visual features. The integration of multi-modal features, such as depth maps [85], language [72], video [86], and knowledge graph [87], can be seamlessly incorporated into our RepSGG architecture as additional queries and keys. The architecture also offers potential for extension to other relationship-related tasks, such as human-object interaction (HOI) detection [88], video-based HOI detection [89], video SGG [90], panoptic SGG [91], and panoptic video SGG [92].

The proposed PGLA serves as a general approach for leveraging performance evaluation to attain a more balanced performance. It can also be adapted for addressing other long-tailed problems (*e.g.*, in visual recognition), utilizing different metrics (such as accuracy), and incorporating other loss functions (such as cross-entropy) with minor adjustments.

6 Conclusions

We have explored novel representations of entities and relationships for scene graph generation, and introduced a performance-guided logit adjustment strategy for long-tailed learning. The proposed RepSGG architecture models entities as subject queries and object keys, and relationships as the attention weights between subjects and objects. The proposed PGLA significantly mitigates the long-tailed problem in SGG. Our experiments demonstrate that RepSGG trained with PGLA compares favourably against box-based, query-based, and point-based SGG models with considerably less design complexity. Our methods also achieve the state-of-the-art performance with fast inference speed. Due to its effectiveness and efficiency, we envision RepSGG to serve as a strong and simple alternative to current mainstream SGG methods.

Acknowledgments

This work was supported in part by Bourns Endowment funds, National Science Foundation (NSF) awards CNS-1730158, ACI-1540112, ACI-1541349, OAC-1826967, OAC-2112167, CNS-2100237, CNS-2120019, the University of California Office of the President, and the University of California San Diego's California Institute for Telecommunications and Information Technology/Qualcomm Institute. Thanks to CENIC for the 100Gbps networks.

References

[1] J. Johnson, R. Krishna, M. Stark, L.-J. Li, D. Shamma, M. Bernstein, and L. Fei-Fei, "Image retrieval using scene graphs," in *CVPR*, 2015, pp. 3668–3678.
[2] X. Yang, K. Tang, H. Zhang, and J. Cai, "Auto-encoding scene graphs for image captioning," in *CVPR*, 2019, pp. 10 685–10 694.
[3] T. Yao, Y. Pan, Y. Li, and T. Mei, "Exploring visual relationship for image captioning," in *ECCV*, 2018, pp. 684–699.
[4] J. Johnson, A. Gupta, and L. Fei-Fei, "Image generation from scene graphs," in *CVPR*, 2018, pp. 1219–1228.
[5] D. A. Hudson and C. D. Manning, "GQA: A new dataset for real-world visual reasoning and compositional question answering," in *CVPR*, 2019, pp. 6700–6709.
[6] D. Teney, L. Liu, and A. van Den Hengel, "Graph-structured representations for visual question answering," in *CVPR*, 2017, pp. 1–9.
[7] Q. Li, F. Xiao, B. Bhanu, B. Sheng, and R. Hong, "Inner knowledge-based img2doc scheme for visual question answering," *ACM TOMM*, vol. 18, no. 3, pp. 1–21, 2022.
[8] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks," in *CVPR*, 2018, pp. 1316–1324.
[9] R. Girshick, "Fast R-CNN," in *ICCV*, 2015, pp. 1440–1448.
[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NeurIPS*, 2015, pp. 91–99.
[11] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*. Springer, 2016, pp. 483–499.
[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *ICCV*, 2017, pp. 2961–2969.
[13] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.
[14] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: A simple and strong anchor-free object detector," *IEEE TPAMI*, vol. 44, no. 4, pp. 1922–1933, 2020.
[15] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei, "Scene graph generation by iterative message passing," in *CVPR*, 2017, pp. 5410–5419.
[16] K. Tang, Y. Niu, J. Huang, J. Shi, and H. Zhang, "Unbiased scene graph generation from biased training," in *CVPR*, 2020, pp. 3716–3725.
[17] A. Newell and J. Deng, "Pixels to graphs by associative embedding," in *NeurIPS*, 2017, pp. 2171–2180.
[18] H. Liu, N. Yan, M. Mortazavi, and B. Bhanu, "Fully convolutional scene graph generation," in *CVPR*, 2021, pp. 11 546–11 556.
[19] G. Adaimi, D. Mizrahi, and A. Alahi, "Composite relationship fields with transformers for scene graph generation," in *WACV*, January 2023, pp. 52–64.
[20] R. Li, S. Zhang, and X. He, "SGTR: End-to-end scene graph generation with transformer," in *CVPR*, 2022, pp. 19 486–19 496.
[21] A. Desai, T.-Y. Wu, S. Tripathi, and N. Vasconcelos, "Single-stage visual relationship learning using conditional queries," *NeurIPS*, vol. 35, pp. 13 064–13 077, 2022.
[22] Y. Cong, M. Y. Yang, and B. Rosenhahn, "RelTR: Relation transformer for scene graph generation," *IEEE TPAMI*, pp. 1–16, 2023.
[23] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *ECCV*. Springer, 2020, pp. 213–229.
[24] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma *et al.*, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *IJCV*, vol. 123, no. 1, pp. 32–73, 2017.
[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *NeurIPS*, vol. 30, 2017.
[26] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable transformers for end-to-end object detection," in *ICLR*, 2021. [Online]. Available: https://openreview.net/forum?id=gZ9hCDWe6ke
[27] A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar, "Long-tail learning via logit adjustment," in *ICLR*, 2021. [Online]. Available: https://openreview.net/forum?id=37nvvqkCo5

[28] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi, "Neural motifs: Scene graph parsing with global context," in *CVPR*, 2018, pp. 5831–5840.

[29] K. Tang, H. Zhang, B. Wu, W. Luo, and W. Liu, "Learning to compose dynamic tree structures for visual contexts," in *CVPR*, 2019, pp. 6619–6628.

[30] S. Woo, D. Kim, D. Cho, and I. S. Kweon, "Linknet: Relational embedding for scene graph," in *NeurIPS*, 2018, pp. 560–570.

[31] B. Dai, Y. Zhang, and D. Lin, "Detecting visual relationships with deep relational networks," in *CVPR*, 2017, pp. 3076–3086.

[32] W. Wang, R. Wang, S. Shan, and X. Chen, "Exploring context and visual pattern of relationship for scene graph generation," in *CVPR*, 2019, pp. 8188–8197.

[33] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *ECCV*, 2018, pp. 734–750.

[34] Y. Lu, H. Rai, J. Chang, B. Knyazev, G. Yu, S. Shekhar, G. W. Taylor, and M. Volkovs, "Context-aware scene graph generation with seq2seq transformers," in *ICCV*, 2021, pp. 15 931–15 941.

[35] N. Dhingra, F. Ritter, and A. Kunz, "BGT-Net: Bidirectional GRU transformer network for scene graph generation," in *CVPR*, 2021, pp. 2150–2159.

[36] Q. Dong, Z. Tu, H. Liao, Y. Zhang, V. Mahadevan, and S. Soatto, "Visual relationship detection using part-and-sum transformers with composite queries," in *ICCV*, 2021, pp. 3550–3559.

[37] Y. Teng and L. Wang, "Structured sparse R-CNN for direct scene graph generation," in *CVPR*, 2022, pp. 19 437–19 446.

[38] J. Chen, A. Agarwal, S. Abdelkarim, D. Zhu, and M. Elhoseiny, "RelTransformer: A transformer-based long-tail visual relationship recognition," in *CVPR*, 2022, pp. 19 507–19 517.

[39] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *ICCV*, 2017, pp. 764–773.

[40] Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin, "Reppoints: Point set representation for object detection," in *ICCV*, 2019, pp. 9657–9666.

[41] Y. Zhang, B. Kang, B. Hooi, S. Yan, and J. Feng, "Deep long-tailed learning: A survey," *IEEE TPAMI*, 2023.

[42] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, and A. Hauptmann, "A comprehensive survey of scene graphs: Generation and application," *IEEE TPAMI*, vol. 45, no. 1, pp. 1–26, 2021.

[43] R. Li, S. Zhang, B. Wan, and X. He, "Bipartite graph network with adaptive message passing for unbiased scene graph generation," in *CVPR*, 2021, pp. 11 109–11 119.

[44] A. Desai, T.-Y. Wu, S. Tripathi, and N. Vasconcelos, "Learning of visual relations: The devil is in the tails," in *ICCV*, 2021, pp. 15 404–15 413.

[45] L. Li, L. Chen, Y. Huang, Z. Zhang, S. Zhang, and J. Xiao, "The devil is in the labels: Noisy label correction for robust scene graph generation," in *CVPR*, 2022, pp. 18 869–18 878.

[46] A. Zhang, Y. Yao, Q. Chen, W. Ji, Z. Liu, M. Sun, and T.-S. Chua, "Fine-grained scene graph generation with data transfer," in *ECCV*. Springer, 2022, pp. 409–424.

[47] T.-J. J. Wang, S. Pehlivan, and J. Laaksonen, "Tackling the unannotated: Scene graph generation with bias-reduced models," in *BMVC*. BMVA, 2020.

[48] M.-J. Chiou, H. Ding, H. Yan, C. Wang, R. Zimmermann, and J. Feng, "Recovering the unbiased scene graphs from the biased ones," in *ACM MM*, 2021, pp. 1581–1590.

[49] Y. Guo, L. Gao, X. Wang, Y. Hu, X. Xu, X. Lu, H. T. Shen, and J. Song, "From general to specific: Informative scene graph generation via balance adjustment," in *ICCV*, 2021, pp. 16 383–16 392.

[50] T. He, L. Gao, J. Song, J. Cai, and Y.-F. Li, "Learning from the scene and borrowing from the rich: tackling the long tail in scene graph generation," in *IJCAI*, 2021, pp. 587–593.

[51] N. Gkanatsios, V. Pitsikalis, and P. Maragos, "From saturation to zero-shot visual relationship detection using local context." in *BMVC*, 2020.

[52] B. Knyazev, H. de Vries, C. Cangea, G. W. Taylor, A. Courville, and E. Belilovsky, "Graph density-aware losses for novel compositions in scene graph generation," in *BMVC*, 2020.

[53] X. Lin, C. Ding, J. Zeng, and D. Tao, "GPS-Net: Graph property sensing network for scene graph generation," in *CVPR*, 2020, pp. 3746–3753.

[54] S. Yan, C. Shen, Z. Jin, J. Huang, R. Jiang, Y. Chen, and X.-S. Hua, "PCPL: Predicate-correlation perception learning for unbiased scene graph generation," in *ACM MM*, 2020, pp. 265–273.

[55] M. Suhail, A. Mittal, B. Siddiquie, C. Broaddus, J. Eledath, G. Medioni, and L. Sigal, "Energy-based learning for scene graph generation," in *CVPR*, 2021, pp. 13 936–13 945.

[56] W. Li, H. Zhang, Q. Bai, G. Zhao, N. Jiang, and X. Yuan, "PPDL: Predicate probability distribution based loss for unbiased scene graph generation," in *CVPR*, 2022, pp. 19 447–19 456.

[57] X. Lyu, L. Gao, Y. Guo, Z. Zhao, H. Huang, H. T. Shen, and J. Song, "Fine-grained predicates learning for scene graph generation," in *CVPR*, 2022, pp. 19 467–19 475.

[58] S. Zhang, Z. Li, S. Yan, X. He, and J. Sun, "Distribution alignment: A unified framework for long-tail visual recognition," in *CVPR*, June 2021, pp. 2361–2370.

[59] H. Wei, R. Xie, H. Cheng, L. Feng, B. An, and Y. Li, "Mitigating neural network overconfidence with logit normalization," in *ICML*. PMLR, 2022, pp. 23 631–23 644.

[60] C. Chen, Y. Zhan, B. Yu, L. Liu, Y. Luo, and B. Du, "Resistance training using prior bias: toward unbiased scene graph generation," in *AAAI*, vol. 36, no. 1, 2022, pp. 212–220.

[61] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[62] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *CVPR*, 2017, pp. 2117–2125.

[63] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[64] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[65] D. Abou Chacra and J. Zelek, "The topology and language of relationships in the visual genome dataset," in *CVPRW*. IEEE, 2022, pp. 4859–4867.

[66] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *ICLR*. OpenReview.net, 2017. [Online]. Available: https://openreview.net/forum?id=rkE3y85ee

[67] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov *et al.*, "The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale," *IJCV*, vol. 128, no. 7, pp. 1956–1981, 2020.

[68] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *ICCV*, 2017, pp. 2980–2988.

[69] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010, pp. 807–814.

[70] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE TPAMI*, vol. 43, no. 10, pp. 3349–3364, 2020.

[71] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021, pp. 10 012–10 022.

[72] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei, "Visual relationship detection with language priors," in *ECCV*. Springer, 2016, pp. 852–869.

[73] T. Chen, W. Yu, R. Chen, and L. Lin, "Knowledge-embedded routing network for scene graph generation," in *CVPR*, 2019, pp. 6163–6171.

[74] J. Zhang, K. J. Shih, A. Elgammal, A. Tao, and B. Catanzaro, "Graphical contrastive losses for scene graph parsing," in *CVPR*, 2019, pp. 11 535–11 543.

[75] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[76] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *ECCV*. Springer, 2014, pp. 740–755.

[77] Z. Tian, H. Chen, X. Wang, Y. Liu, and C. Shen, "AdelaiDet: A toolbox for instance-level recognition tasks," https://git.io/adelaidet, 2019.

[78] A. Gupta, P. Dollar, and R. Girshick, "LVIS: A dataset for large vocabulary instance segmentation," in *CVPR*, 2019, pp. 5356–5364.

[79] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *ICLR*, 2018.

[80] N. Gkanatsios, V. Pitsikalis, P. Koutras, and P. Maragos, "Attention-translation-relation network for scalable scene graph generation," in *ICCVW*, 2019, pp. 0–0.

[81] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh, "Graph R-CNN for scene graph generation," in *ECCV*, 2018, pp. 670–685.

[82] D. Liu, M. Bober, and J. Kittler, "Constrained structure learning for scene graph generation," *IEEE TPAMI*, 2023.

[83] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *CVPR*, 2017, pp. 1492–1500.
[84] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *JMLR*, vol. 9, no. 11, 2008.
[85] S. Sharifzadeh, S. M. Baharlou, M. Berrendorf, R. Koner, and V. Tresp, "Improving visual relation detection using depth maps," in *ICPR*. IEEE, 2021, pp. 3597–3604.
[86] J. Ji, R. Krishna, L. Fei-Fei, and J. C. Niebles, "Action genome: Actions as compositions of spatio-temporal scene graphs," in *CVPR*, 2020, pp. 10 236–10 247.
[87] A. Zareian, S. Karaman, and S.-F. Chang, "Bridging knowledge graphs to generate scene graphs," in *ECCV*. Springer, 2020, pp. 606–623.
[88] Y.-W. Chao, Y. Liu, X. Liu, H. Zeng, and J. Deng, "Learning to detect human-object interactions," in *WACV*. IEEE, 2018, pp. 381–389.
[89] M.-J. Chiou, C.-Y. Liao, L.-W. Wang, R. Zimmermann, and J. Feng, "ST-HOI: A spatial-temporal baseline for human-object interaction detection in videos," in *ICDARW*, 2021, pp. 9–17.
[90] X. Shang, T. Ren, J. Guo, H. Zhang, and T.-S. Chua, "Video visual relation detection," in *ACM MM*, 2017, pp. 1300–1308.
[91] J. Yang, Y. Z. Ang, Z. Guo, K. Zhou, W. Zhang, and Z. Liu, "Panoptic scene graph generation," in *ECCV*. Springer, 2022, pp. 178–196.
[92] J. Yang, W. Peng, X. Li, Z. Guo, L. Chen, B. Li, Z. Ma, K. Zhou, W. Zhang, C. C. Loy *et al.*, "Panoptic video scene graph generation," in *CVPR*, 2023, pp. 18 675–18 685.

Bir Bhanu (F'95, LF'17) received B.S. (with Hons.) from IIT-BHU; M.E (with Distinction) from BITS (Pilani); S.M. and E.E. in electrical engineering and computer science from Massachusetts Institute of Technology, Cambridge, MA; Ph.D. in electrical engineering from the University of Southern California, Los Angeles, CA and M.B.A. from the University of California, Irvine, CA. He is the Bourns endowed University of California Presidential Chair in Engineering, the Distinguished Professor of electrical and computer engineering and the Founding Director of the interdisciplinary Center for Research in Intelligent Systems (1998-2019) and the Visualization and Intelligent Systems Laboratory (1991-) at the University of California, Riverside (UCR), CA. He is the first Founding faculty of Bourns College of Engineering and served as the Founding Professor and Chair (1991-94) of electrical engineering with UCR. He has been the cooperative Professor of computer science and engineering (since 1991), bioengineering (since 2006) and mechanical engineering (since 2008). Recently he served as the Interim Chair of the Department of Bioengineering from 2014-16. He also served as the Director of the National Science Foundation graduate research and training program in video bioinformatics with UCR. Prior to joining UCR in 1991, he was a Senior Honeywell Fellow with Honeywell Inc. He has published extensively and has U.S. and international patents. He has received university and industry awards for research excellence, outstanding contributions and team efforts as well as journal/conference best paper awards. His research interests include computer vision, pattern recognition and data mining, machine learning, artificial intelligence, image processing, image and video database, graphics and visualization, robotics, human-computer interactions, and biological, medical, military and intelligence applications. Dr. Bhanu is a Fellow of IEEE, AAAS, IAPR, SPIE, NAI and AIMBE.

Hengyue Liu received the B.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, and M.S. in electrical engineering from the University of Southern California. He is currently working towards his Ph.D in computer vision at the Visualization and Intelligent Systems Laboratory (VISLab), University of California, Riverside, CA, USA. His research interests include object detection, scene graph generation and mobile vision.