Graph R-CNN for Scene Graph Generation

Jianwei Yang[1⋆], Jiasen Lu[1⋆], Stefan Lee[1], Dhruv Batra[1,2], and Devi Parikh[1,2]

[1]Georgia Institute of Technology [2]Facebook AI Research
{jw2yang, jiasenlu, steflee, dbatra, parikh}@gatech.edu

Abstract. We propose a novel scene graph generation model called Graph R-CNN, that is both effective and efficient at detecting objects and their relations in images. Our model contains a Relation Proposal Network (RePN) that efficiently deals with the quadratic number of potential relations between objects in an image. We also propose an attentional Graph Convolutional Network (aGCN) that effectively captures contextual information between objects and relations. Finally, we introduce a new evaluation metric that is more holistic and realistic than existing metrics. We report state-of-the-art performance on scene graph generation as evaluated using both existing and our proposed metrics.

Keywords: Graph R-CNN, Scene Graph Generation, Relation Proposal Network, Attentional Graph Convolutional Network

1 Introduction

Visual scene understanding has traditionally focused on identifying *objects in images* – learning to predict their presence (*i.e.*image classification [9, 15, 33]) and spatial extent (*i.e.*object detection [7, 22, 30] or segmentation [21]). These object-centric techniques have matured significantly in recent years, however, representing scenes as collections of objects fails to capture relationships which may be essential for scene understanding.

A recent work [12] has instead proposed representing visual scenes as graphs containing objects, their attributes, and the relationships between them. These *scene graphs* form an interpretable structured representation of the image that can support higher-level visual intelligence tasks such as captioning [38], visual question answering [1, 11, 34, 36–38], and image-grounded dialog [3]. While scene graph representations hold tremendous promise, extracting scene graphs from images – efficiently and accurately – is challenging. The natural approach of considering every pair of nodes (objects) as a potential edge (relationship) – essentially reasoning over fully-connected graphs – is often effective in modeling contextual relationships but scales poorly (quadratically) with the number of objects, quickly becoming impractical. The naive fix of randomly sub-sampling edges to be considered is more efficient but not as effective since the distribution of interactions between objects is far from random – take Fig. 1(a) as an example,

⋆ Equal contribution

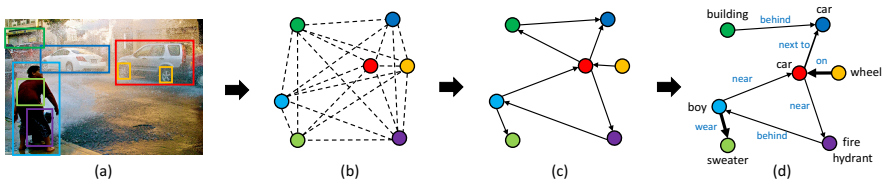

Fig. 1. Given an image (a), our proposed approach first extracts a set of objects visible in the scene and considers possible relationships between all nodes (b). Then it prunes unlikely relationships using a learned measure of 'relatedness', producing a sparser candidate graph structure (c). Finally, an attentional graph convolution network is applied to integrate global context and update object node and relationship edge labels.

it is much more likely for a 'car' and 'wheel' to have a relationship than a 'wheel' and 'building'. Furthermore, the types of relationships that typically occur between objects are also highly dependent on those objects.

Graph R-CNN. In this work, we propose a new framework, Graph R-CNN, for scene graph generation which effectively leverages object-relationship regularities through two mechanisms to intelligently sparsify and reason over candidate scene graphs. Our model can be factorized into three logical stages: 1) object node extraction, 2) relationship pruning, and 3) graph context integration, which are depicted in Fig. 1. In the object node extraction stage, we utilize a standard object detection pipeline [31]. This results in a set of localized object regions as shown in Fig. 1b. We introduce two key novelties in the rest of the pipeline to incorporate the real-world regularities in object relationships discussed above. First, we introduce a relation proposal network (RePN) that learns to efficiently compute *relatedness scores* between object pairs which are used to intelligently prune unlikely scene graph connections (as opposed to random pruning in prior work). A sparse post-pruning graph is shown in Figure 1c. Second, given the resulting sparsely connected scene graph candidate, we apply an attentional graph convolution network (aGCN) to propagate higher-order context throughout the graph – updating each object and relationship representation based on its neighbors. In contrast to existing work, we predict per-node edge attentions, enabling our approach to learn to modulate information flow across unreliable or unlikely edges. We show refined graph labels and edge attentions (proportional to edge width) in Figure 1d.

To validate our approach, we compare our performance with existing methods on the Visual Genome [14] dataset and find that our approach achieves an absolute gain of 5.3 on Recall@50 for scene graph generation [39]. We also perform extensive model ablations and quantify the impact of our modeling choices.

Evaluating Scene Graph Generation. Existing metrics for scene graph generation are based on recall of ⟨subject, predicate, object⟩ triplets (e.g. `SGGen` from [14]) or of objects and predicates given ground truth object localizations (e.g. `PredCls` and `PhrCls` from [14]). In order to expose a problem with these

metrics, consider a method that mistakes the boy in Figure 1a as a man but otherwise identifies that he is 1) standing behind a fire hydrant, 2) near a car, and 3) wearing a sweater. Under the triplet-based metrics, this minor error (boy vs man) would be heavily penalized despite most of the boy's relationships being correctly identified. Metrics that provide ground-truth regions side-step this problem by focusing strictly on relationship prediction but cannot accurately reflect the test-time performance of the entire scene graph generation system.

To address this mismatch, we introduce a novel evaluation metric (`SGGen+`) that more holistically evaluates the performance of scene graph generation with respect to objects, attributes (if any), and relationships. Our proposed metric `SGGen+` computes the total recall for singleton entities (objects and predicates), pair entries ⟨object, attribute⟩ (if any), and triplet entities ⟨subject, predicate, object⟩. We report results on existing methods under this new metric and find our approach also outperforms the state-of-the-art significantly. More importantly, this new metric provides a more robust and holistic measure of similarity between generated and ground-truth scene graphs.

Summary of Contributions. Concretely, this work addresses the scene graph generation problem by introducing a novel model (Graph R-CNN), which can leverage object-relationship regularities, and proposes a more holistic evaluation metric (`SGGen+`) for scene graph generation. We benchmark our model against existing approaches on standard metrics and this new measure – outperforming existing approaches.

2 Related Work

Contextual Reasoning and Scene Graphs. The idea of using context to improve scene understanding has a long history in computer vision [16, 26, 27, 29]. More recently, inspired by representations studied by the graphics community, Johnson *et al.* [12] introduced the problem of extracting scene graphs from images, which generalizes the task of object detection [6, 7, 22, 30, 31] to also detecting relationships and attributes of objects.

Scene Graph Generation. A number of approaches have been proposed for the detection of both objects and their relationships [2,17–19,23,25,28,39,41–43,45]. Though most of these works point out that reasoning over a quadratic number of relationships in the scene graph is intractable, each resorted to heuristic methods like random sampling to address this problem. Our work is the first to introduce a trainable relationship proposal network (RePN) that learns to prune unlikely relationship edges from the graph without sacrificing efficacy. RePN provides high-quality relationship candidates, which we find improves overall scene graph generation performance.

Most scene graph generation methods also include some mechanisms for context propagation and reasoning over a candidate scene graph in order to refine the final labeling. In [39], Xu *et al.*decomposed the problem into two sub-graphs – one for objects and one for relationships – and performed message passing.

Similarly in [17], the authors propose two message-passing strategies (parallel and sequential) for propagating information between objects and relationships. Dai *et al.* [2] address model the scene graph generation process as inference on a conditional random field (CRF). Newell *et al.* [25] proposed to directly generate scene graphs from image pixels without the use of object detector based on associative graph embeddings. In our work, we develop a novel attentional graph convolutional network (aGCN) to update node and relationship representations by propagating context between nodes in candidate scene graphs – operating both on visual and semantic features. While similar in function to the message-passing based approach above, aGCN is highly efficient and can learn to place attention on reliable edges and dampen the influence of unlikely ones.

A number of previous approaches have noted the strong regularities in scene graph generation which motivate our approach. In [23], Lu *et al.*integrate semantic priors from language to improve the detection of meaningful relationships between objects. Likewise, Li *et al.* [18] demonstrated that region captions can also provide useful context for scene graph generation. Most related to our motivation, Zeller *et al.* [41] formalize the notion of motifs (*i.e.*, regularly occurring graph structures) and examine their prevalence in the Visual Genome dataset [14]. The authors also propose a surprisingly strong baseline which directly uses frequency priors to predict relationships – explicitly integrating regularities in graph structure.

Relationship Proposals. Our Relationship Proposal Network (RePN) is inspired and relates strongly to the region proposal network (RPN) of faster R-CNN [31] used in object detection. Our RePN is also similar in spirit to the recently-proposed relationship proposal network (Rel-PN) [44]. There are a number of subtle differences between these approaches. The Rel-PN model independently predicts proposals for subject, objects and predicates, and then re-scores all valid triples, while our RePN generates relations conditioned on objects, allowing it to learn object-pair relationship biases. Moreover, their approach is class agnostic and has not been used for scene graph generation.

Graph Convolutional Networks (GCNs). GCNs were first proposed in [13] in the context of semi-supervised learning. GCNs decompose complicated computation over graph data into a series of localized operations (typically only involving neighboring nodes) for each node at each time step. The structure and edge strengths are typically fixed prior to the computation. For completeness, we note that an upcoming publication [35] has concurrently and independently developed a similar GCN attention mechanism (as aGCN) and shown its effectiveness in other (non computer vision) contexts.

3 Approach

In this work, we model scene graphs as graphs consisting of image regions, relationships, and their labellings. More formally, let I denote an image, V be a set of nodes corresponding to localized object regions in I, $E \in \binom{V}{2}$ denote the relationships (or edges) between objects, and O and R denote object and relationship

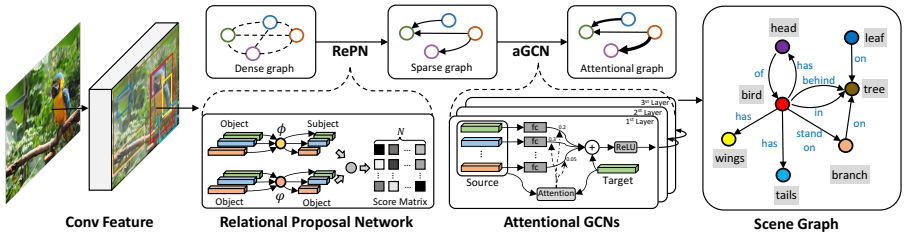

Fig. 2. The pipeline of our proposed Graph R-CNN framework. Given an image, our model first uses RPN to propose object regions, and then prunes the connections between object regions through our relation proposal network (RePN). Attentional GCN is then applied to integrate contextual information from neighboring nodes in the graph. Finally, the scene graph is obtained on the right side.

labels respectively. Thus, the goal is to build a model for $P(S = (V, E, O, R)|I)$. In this work, we factorize the scene graph generation process into three parts:

$$P(\mathcal{S}|\boldsymbol{I}) = \overbrace{P(\boldsymbol{V}|\boldsymbol{I})}^{\text{Object Region Proposal}} \underbrace{P(\boldsymbol{E}|\boldsymbol{V},\boldsymbol{I})}_{\text{Relationship Proposal}} \overbrace{P(\boldsymbol{R},\boldsymbol{O}|\boldsymbol{V},\boldsymbol{E},\boldsymbol{I})}^{\text{Graph Labeling}} \tag{1}$$

which separates graph construction (nodes and edges) from graph labeling. This is a normal factorization process, but let us recap the intuition behind each term. First, the object region proposal $P(\boldsymbol{V}|\boldsymbol{I})$ is typically modeled using an off-the-shelf object detection system such as [31] to produce candidate regions. Notably, existing methods typically model the second relationship proposal term $P(\boldsymbol{E}|\boldsymbol{V},\boldsymbol{I})$ as a uniform random sampling of potential edges between vertices $\boldsymbol{V}$. In contrast, we propose a relationship proposal network (RePN) to directly model $P(\boldsymbol{E}|\boldsymbol{V},\boldsymbol{I})$ – making our approach the first that allows for learning the entire generation process end-to-end. Finally, the graph labeling process $P(\boldsymbol{R},\boldsymbol{O}|\boldsymbol{V},\boldsymbol{E},\boldsymbol{I})$ is typically treated as an iterative label refinement process [2, 17, 39]. A brief pipeline is shown in Fig. 2.

In the following, we discuss the components of our proposed Graph R-CNN model corresponding to each of the terms in Eq. 1. First, we discuss our use of Faster R-CNN [31] for node generation in Section 3.1. Then in Section 3.2 we introduce our novel relation proposal network architecture to intelligently generate edges. Finally, in Section 3.3 we present our graph convolutional network [13] with learned attention to adaptively integrate global context for graph labeling.

3.1 Object Proposals

In our approach, we use the Faster RCNN [31] framework to extract a set of n object proposals from an input image. Each object proposal i is associated with

a spatial region $r_i^o = [x_i, y_i, w_i, h_i]$, a pooled feature vector x_i^o, and an initial estimated label distribution p_i^o over classes $C=\{1, \ldots, k\}$. We denote the collection of these vectors for all n proposals as the matrices $R^o \in \mathbb{R}^{n\times 4}$, $X^o \in \mathbb{R}^{n\times d}$, and $P^o \in \mathbb{R}^{n\times|C|}$ respectively.

3.2 Relation Proposal Network

Given the n proposed object nodes from the previous step, there are $O(n^2)$ possible connections between them; however, as previously discussed, most object pairs are unlikely to have relationships due to regularities in real-world object interactions. To model these regularities, we introduce a relation proposal network (RePN) which learns to efficiently estimate the *relatedness* of an object pair. By pruning edges corresponding to unlikely relations, the RePN can efficiently sparsify the candidate scene graph – retaining likely edges and suppressing noise introduced from unlikely ones.

In this paper, we exploit the estimated class distributions (P^o) to infer relatedness – essentially learning soft class-relationships priors. This choice aligns well with our intuition that certain classes are relatively unlikely to interact compared with some other classes. Concretely, given initial object classification distributions P^o, we score all $n * (n-1)$ directional pairs $\{\boldsymbol{p}_i^o, \boldsymbol{p}_j^o | i \neq j\}$, computing the relatedness as $s_{ij} = f(\boldsymbol{p}_i^o, \boldsymbol{p}_j^o)$ where $f(\cdot,\cdot)$ is a learned relatedness function. One straightforward implementation of $f(\cdot,\cdot)$ could be passing the concatenation $[\boldsymbol{p}_i^o, \boldsymbol{p}_j^o]$ as input to a multi-layer perceptron which outputs the score. However, this approach would consume a great deal of memory and computation given the quadratic number of object pairs. To avoid this, we instead consider an asymmetric kernel function:

$$f(\boldsymbol{p}_i^o, \boldsymbol{p}_j^o) = \langle \Phi(\boldsymbol{p}_i^o), \Psi(\boldsymbol{p}_j^o) \rangle, i \neq j \tag{2}$$

where $\Phi(\cdot)$ and $\Psi(\cdot)$ are projection functions for subjects and objects in the relationships respectively[1]. This decomposition allows the score matrix $S = \{s_{ij}\}^{n\times n}$ to be computed *with only two projection processes for X^o followed by a matrix multiplication*. We use two multi-layer perceptrons (MLPs) with identical architecture (but different parameters) for $\Phi(\cdot)$ and $\Psi(\cdot)$. We also apply a sigmoid function element-wise to S such that all relatedness scores range from 0 to 1.

After obtaining the score matrix for all object pairs, we sort the the scores in descending order and choose top K pairs. We then apply non-maximal suppression (NMS) to filter out object pairs that have significant overlap with others. Each relationship has a pair of bounding boxes, and the combination order matters. We compute the overlap between two object pairs $\{u, v\}$ and $\{p, q\}$ as:

$$IoU(\{u,v\},\{p,q\}) = \frac{I(r_u^o, r_p^o) + I(r_v^o, r_q^o)}{U(r_u^o, r_p^o) + U(r_v^o, r_q^o)} \tag{3}$$

[1] We distinguish between the first and last object in a relationship as subject and object respectively, that is, $\langle$`subject`, `relationship`, `object`$\rangle$.

where operator I computes the intersection area between two boxes and U the union area. The remaining m object pairs are considered as candidates having meaningful relationships $\boldsymbol{E}$. With $\boldsymbol{E}$, we obtain a graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, which is much sparser than the original fully connected graph. Along with the edges proposed for the graph, we get the visual representations $X^r = \{\boldsymbol{x}_1^r, ..., \boldsymbol{x}_m^r\}$ for all m relationships by extracting features from the union box of each object pair.

3.3 Attentional GCN

To integrate contextual information informed by the graph structure, we propose an attentional graph convolutional network (aGCN). Before we describe our proposed aGCN, let us briefly recap a 'vanilla' GCN in which each node i has a representation $z_i \in \mathbb{R}^d$, as proposed in [13]. Briefly, for a target node i in the graph, the representations of its neighboring nodes $\{z_j \mid j \in \mathcal{N}(i)\}$ are first transformed via a learned linear transformation W. Then, these transformed representations are gathered with predetermined weights α, followed by a non-linear function σ (ReLU [24]). This layer-wise propagation can be written as:

$$\boldsymbol{z}_i^{(l+1)} = \sigma\left(\boldsymbol{z}_i^{(l)} + \sum_{j \in \mathcal{N}(i)} \alpha_{ij} W \boldsymbol{z}_j^{(l)}\right) \tag{4}$$

or equivalently we can collect node representations into a matrix $Z \in \mathbb{R}^{d \times Tn}$

$$\boldsymbol{z}_i^{(l+1)} = \sigma\left(W Z^{(l)} \boldsymbol{\alpha}_i\right) \tag{5}$$

for $\boldsymbol{\alpha}_i \in [0, 1]^n$ with 0 entries for nodes not neighboring i and $\boldsymbol{\alpha}_{ii} = 1$. In conventional GCN, the connections in the graph are known and coefficient vector $\boldsymbol{\alpha}_i$ are preset based on the symmetrically normalized adjacency matrix of features.

In this paper, we extend the conventional GCN to an attentional version, which we refer to as aGCN, by learning to adjust $\boldsymbol{\alpha}$. To predict attention from node features, we learn a 2-layer MLP over concatenated node features and compute a softmax over the resulting scores. The the attention for node i is

$$u_{ij} = w_h^T \sigma(W_a[\boldsymbol{z}_i^{(l)}, \boldsymbol{z}_j^{(l)}]) \tag{6}$$

$$\boldsymbol{\alpha}_i = \text{softmax}(\boldsymbol{u}_i), \tag{7}$$

where w_h and W_a are learned parameters and $[\cdot, \cdot]$ is the concatenation operation. By definition, we set $\boldsymbol{\alpha}_{ii} = 1$ and $\boldsymbol{\alpha}_{ij} = 0 \, \forall j \notin \mathcal{N}(i)$. As attention is a functions of node features, each iteration results in altered attentions which affects successive iterations.

aGCN for Scene Graph Generation. Recall that from the previous sections we have a set of N object regions and m relationships. From these, we construct a graph G with nodes corresponding to object and relationship proposals. We insert edges between relation nodes and their associated objects. We also add skip-connect edges directly between all object nodes. These connections allow

information to flow directly between object nodes. Recent work has shown that reasoning about object correlation can improve detection performance [10]. We apply aGCN to this graph to update object and relationship representations based on global context.

Note that our graph captures a number of different types of connections (i.e. `object` $\leftrightarrow$ `relationship`, `relationship` $\leftrightarrow$ `subject` and `object` $\leftrightarrow$ `object`). In addition, the information flow across each connection may be asymmetric (the informativeness of `subject` on `relationship` might be quite different from `relationship` to `subject`). We learn different transformations for each type and ordering – denoting the linear transform from node type a to node type b as W^{ab} with s=subjects, o=objects, and r=relationships. Using the same notation as in Eq. 5 and writing object and relationship features as Z^o and Z^r, we write the representation update for object nodes as

$$\boldsymbol{z}_i^o = \sigma(\overbrace{W^{\texttt{skip}} Z^o \boldsymbol{\alpha}^{\texttt{skip}}}^{\substack{\text{Message from} \\ \text{Other Objects}}} + \overbrace{W^{sr} Z^r \boldsymbol{\alpha}^{sr} + W^{or} Z^r \boldsymbol{\alpha}^{or}}^{\substack{\text{Messages from} \\ \text{Neighboring Relationships}}}) \tag{8}$$

with $\boldsymbol{\alpha}_{ii}^{\texttt{skip}}$=1 and similarly for relationship nodes as

$$\boldsymbol{z}_i^r = \sigma(\boldsymbol{z}_i^r + \underbrace{W^{rs} Z^o \boldsymbol{\alpha}^{rs} + W^{ro} Z^o \boldsymbol{\alpha}^{ro}}_{\text{Messages from Neighboring Objects}}). \tag{9}$$

where $\boldsymbol{\alpha}$ are computed at each iteration as in Eq. 7.

One open choice is how to initialize the object and relationship node representations z which could potentially be set to any intermediate feature representation or even the pre-softmax output corresponding to class labels. In practice, we run both a visual and semantic aGCN computation – one with visual features and the other using pre-softmax outputs. In this way, we can reason about both lower-level visual details (*i.e.*two people are likely talking if they are facing one another) as well as higher-level semantic co-occurrences (*i.e.*cars have wheels). Further, we set the attention in the semantic aGCN to be that of the visual aGCN – effectively modulating the flow of semantic information based on visual cues. This also enforces that real-world objects and relationships represented in both graphs interact with others in the same manner.

3.4 Loss Function

In Graph R-CNN, we factorize the scene graph generation process into three sub-processes: $P(\boldsymbol{R}, \boldsymbol{O}|\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{I})$, $P(\boldsymbol{E}|\boldsymbol{V}, \boldsymbol{I})$, $P(\boldsymbol{V}|\boldsymbol{I})$, which were described above. During training, each of these sub-processes are trained with supervision. For $P(\boldsymbol{V}|\boldsymbol{I})$, we use the same loss as used in RPN, which consists of a binary cross entropy loss on proposals and a regression loss for anchors. For $P(\boldsymbol{E}|\boldsymbol{V}, \boldsymbol{I})$, we use another binary cross entropy loss on the relation proposals. For the final scene graph generation $P(\boldsymbol{R}, \boldsymbol{O}|\boldsymbol{V}, \boldsymbol{E}, \boldsymbol{I})$, two multi-class cross entropy losses are used for object classification and predicate classification.

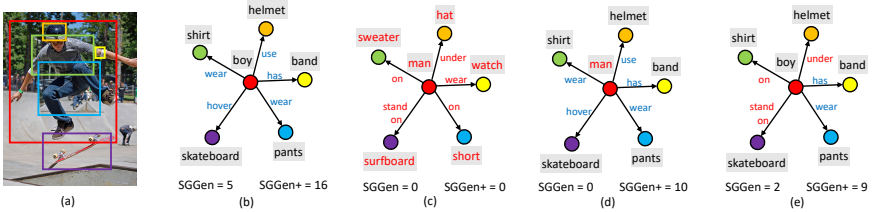

Fig. 3. A example to demonstrate the difference between `SGGen` and `SGGen+`. Given the input image (a), its ground truth scene graph is depicted in (b). (c)-(e) are three generated scene graphs. For clarity, we merely show the connections with *boy*. At the bottom of each graph, we compare the number of correct predictions for two metrics.

4 Evaluating Scene Graph Generation

Scene graph generation is naturally a structured prediction problem over attributed graphs, and how to correctly and efficiently evaluate predictions is an under-examined problem in prior work on scene graph generation. We note that graph similarity based on minimum graph edit distance has been well-studied in graph theory [5]; however, computing exact solution is NP-complete and approximation APX-hard [20].

Prior work has circumvented these issues by evaluating scene graph generation under a simple triplet-recall based metric introduced in [39]. Under this metric which we will refer to as `SGGen`, the ground truth scene graph is represented as a set of $\langle$`object`, `relationship`, `subject`$\rangle$ triplets and recall is computed via exact match. That is to say, a triplet is considered 'matched' in a generated scene graph if all three elements have been correctly labeled, and both `object` and `subject` nodes have been properly localized (i.e. bounding box IoU > 0.5). While simple to compute, this metric results in some unintuitive notions of similarity that we demonstrate in Figure 3.

Figure 3a shows an input image overlaid with bounding box localizations of correspondingly colored nodes in the ground truth scene graph shown in (b). (c), (d), and (e) present erroneously labeled scene graphs corresponding to these same localizations. Even a casual examination of (c) and (d) yields the stark difference in their accuracy – while (d) has merely mislabeled the boy as a man, (c) has failed to accurately predict even a single node or relationship! Despite these differences, neither recalls a single complete triplet and are both scored identically under `SGGen` (i.e. 0).

To address this issue, we propose a new metric called `SGGen+` as the augmentation of `SGGen`. `SGGen+` not only considers the triplets in the graph, but also the singletons (object and predicate). The computation of `SGGen+` can be formulated as:

$$Recall@N = \frac{C(O) + C(P) + C(T)}{N} \tag{10}$$

where $C(\cdot)$ is a counting operation, and hence $C(O)$ is the number of object nodes correctly localized and recognized; $C(P)$ is for predicate. Since the location of predicate depends on the location of subject and object, only if both subject and object are correctly localized and the predicate is correctly recognized, we will count it as one. $C(T)$ is for triplet, which is the same as `SGGen`. Here, N is the number of entries (the sum of number of objects, predicates and relationships) in the ground truth graph. In Figure 3, using our `SGGen+`, the recall for graph (c) is still 0, since all predictions are wrong. However, the recall for graph (d) is not 0 any more since most of the object and all predicate predictions are correct, except for one wrong prediction for the red node. Based on our new metric, we can obtain much finer measurement of scene graph similarity.

5 Experiments

Recently, there are some inconsistencies in existing work on scene graph generation in terms of data preprocessing, data splits, and evaluation. This makes it difficult to systemtically benchmark progress and cleanly compare numbers across papers. So we we first clarify the details of our experimental settings.

Datasets. There are a number of splits of the Visual Genome dataset that have been used in the scene graph generation literature [18, 39, 44]. The most commonly used is the one proposed in [39]. Hence, in our experiments, we follow their preprocessing strategy and dataset split. After preprocessing, the dataset is split into training and test sets, which contains 75,651 images and 32,422 images, respectively. In this dataset, the top-frequent 150 object classes and 50 relation classes are selected. Each image has around 11.5 objects and 6.2 relationships in the scene graph.

Training. For training, multiple strategies have been used in literature. In [18, 25, 39], the authors used two-stage training, where the object detector is pre-trained, followed by the joint training of the whole scene graph generation model. To be consistent with previous work [18,39], we also adopt the two-stage training – we first train the object detector and then train the whole model jointly until convergence.

Metrics. We use four metrics for evaluating scene graph generation, including three previously used metrics and our proposed `SGGen+` metric:

- **Predicate Classification (PredCls)**: The performance for recognizing the relation between two objects given the ground truth locations.
- **Phrase Classification (PhrCls)**: The performance for recognizing two object categories and their relation given the ground truth locations.
- **Scene Graph Generation (SGGen)**: The performance for detecting objects (IoU > 0.5) *and* recognizing the relations between object pairs.
- **Finer Scene Graph Generation (SGGen+)**: Besides the triplets counted by `SGGen`, it considers the singletons and pairs (if any), as described earlier.

Evaluation. In our experiments, we multiply the classification scores for subjects, objects and their relationships, then sort them in descending order.

Based on this order, we compute the recall at top 50 and top 100, respectively. Another difference in existing literature in the evaluation protocol is w.r.t. the SGCls and PredCl metrics. Some previous works [18, 25] used different models to evaluate along different metrics. However, such a comparison is unfair since the models could be trained to overfit the respective metrics. For meaningful evaluation, we evaluate a single model – the one obtained after joint training – across all metrics.

5.1 Implementation Details

We use Faster R-CNN [31] associated with VGG16 [32] as the backbone based on the PyTorch re-implementation [40]. During training, the number of proposals from RPN is 256. For each proposal, we perform ROI Align [8] pooling, to get a 7×7 response map, which is then feed to a two-layer MLP to obtain each proposal's representation. In RePN, the projection functions $\Phi(\cdot)$ and $\Psi(\cdot)$ are simply two-layer MLPs. During training, we sample 128 object pairs from the quadratic number of candidates. We then obtain the union of boxes of the two objects and extract a representation for the union. The threshold for box-pair NMS is 0.7. In aGCN, to obtain the attention for one node pair, we first project the object/predicate features into 256-d, and then concatenate them into 512-d, which is then feed to a two-layer MLP with a 1-d output. For aGCN, we use two aGCN layers at the feature level and semantic level, respectively. The attentions on the graph is updated in each aGCN layer at the feature level, which is then fixed and sent to the aGCN at semantic level.

Training. As mentioned, we perform stage-wise training – we first pretrain Faster R-CNN for object detection, and then fix the parameters in backbone to train the scene graph generation model. SGD is used as the optimizer, with initial learning rate 1e-2 for both training stages.

5.2 Analysis on New Metric

We first quantitatively demonstrate the difference between our proposed metric `SGGen+` and `SGGen`. We compare them by perturbing ground truth scene graphs. We consider assigning random incorrect labels to objects; perturbing objects 1) without relationships, 2) with relationships, and 3) both. We vary the fraction of nodes which are perturbed among {20%, 50%, 100%}. Recall is reported for both metrics. As shown in Table 1, `SGGen` is completely insensitive to the perturbation of objects without relationships (staying at 100 consistently) since it only considers relationship triplets. Note that there are on average 50.1% objects without relationships in the dataset, which `SGGen` omits. On the other hand, `SGGen` is overly sensitive to label errors on objects with relationships (reporting 54.1 at only 20% perturbation where the overall scene graph is still quite accurate). Note that even at 100% perturbation the object localizations and relationships are still correct such that `SGGen+` provides a non-zero score, unlike `SGGen` which considers the graph entirely wrong. Overall, we hope this analysis demonstrates that `SCGen+` is better behaved compared to `SCGen`.

Perturb Type	**none**	**w/o relationship**			**w/ relationship**			**both**		
Perturb Ratio	0%	20%	50%	100%	20%	50%	100%	20%	50%	100%
SGGen	100.0	100.0	100.0	100.0	54.1	22.1	0.0	62.2	24.2	0.0
SGGen+	100.0	94.5	89.1	76.8	84.3	69.6	47.9	80.1	56.6	22.8

Table 1. Comparisons between `SGGen` and `SGGen+` under different perturbations.

	SGGen+		SGGen		PhrCls		PredCls	
Method	R@50	R@100	R@50	R@100	R@50	R@100	R@50	R@100
IMP [39]	-	-	3.4	4.2	21.7	24.4	44.8	53.0
MSDN [18]	-	-	7.7	10.5	19.3	21.8	63.1	66.4
Px2Graph [25]	-	-	9.7	11.3	26.5	30.0	68.0	75.2
IMP† [39]	23.6	28.1	3.8	4.7	20.4	25.0	44.9	52.4
MSDN† [18]	26.6	31.2	6.1	7.3	21.8	25.8	47.7	56.3
NM-Freq† [41]	26.6	31.2	6.1	7.3	21.8	25.8	47.7	56.3
Graph R-CNN (Us)	**33.9**	**35.3**	**11.4**	**13.5**	**31.3**	**33.1**	**54.7**	**59.4**

Table 2. Comparison on Visual Genome test set [14]. We reimplemented IMP [39] and MSDN [18] using the same object detection backbone for fair comparison.

5.3 Quantitative Comparison

We compare our Graph R-CNN with recent proposed methods, including Iterative Message Passing (IMP) [39], Multi-level scene Description Network (MSDN) [18]. Furthermore, we evaluate the neural motif frequency baseline proposed in [41]. Note that previous methods often use slightly different pre-training procedures or data split or extra supervisions. For a fair comparison and to control for such orthogonal variations, we reimplemented IMP, MSDN and frequency baseline in our codebase. Then, we re-train IMP and MSDN based on our backbone – specifically, we used the same pre-trained object detector, and then jointly train the scene graph generator until convergence. We denote these as IMP† and MSDN†. Using the same pre-trained object detector, we report the neural motif frequency baseline in [41] as NM-Freq†.

We report the scene graph generation performance in Table 2. The top three rows are numbers reported in original paper, and the bottom four rows are the numbers from our re-implementations. First, we note that our re-implementations of IMP and MSDN (IMP† and MSDN†) result in performance that is close to or better than the originally reported numbers under some metrics (but not all), which establishes that the take-away messages next are indeed due to our proposed architectural choices – relation proposal network and attentional GCNs. Next, we notice that Graph R-CNN outperforms IMP† and MSDN†. This indicates that our proposed Graph R-CNN model is more effective to extract the scene graph from images. Our approach also outperforms the frequency baseline on all metrics, demonstrating that our model has not just learned simple co-occurrence statistics from training data, but rather also captures context in

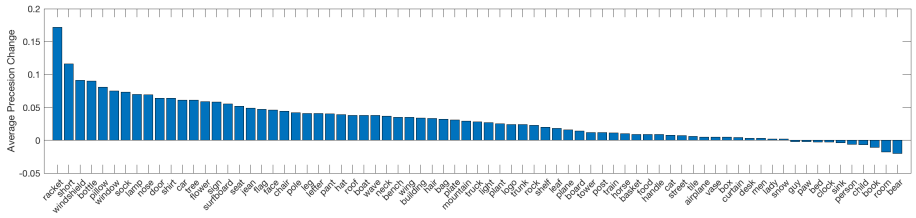

Fig. 4. Per category object detection performance (average precision) change after adding RePN.

RePN	GCN	aGCN	Detection	SGGen+		SGGen		PhrCls		PredCls	
			mAP@0.5	R@50	R@100	R@50	R@100	R@50	R@100	R@50	R@100
-	-	-	10.6	21.2	23.7	3.2	4.2	18.8	23.5	34.2	43.6
✓	-	-	**13.1**	27.4	29.1	6.7	9.1	20.5	24.6	36.4	44.7
✓	✓	-	13.0	31.5	33.9	10.7	12.8	29.9	31.9	52.6	57.9
✓	-	✓	13.0	**33.9**	**35.3**	**11.4**	**13.5**	**31.3**	**33.1**	**54.7**	**59.4**

Table 3. Ablation studies on Graph R-CNN. We report the performance based on four scene graph generation metrics and the object detection performance in mAP.

individual images. More comprehensively, we compare with IMP and MSDN on the efficiency over training and inference. IMP uses 2.15× while MSDN uses 1.86× our method. During inference, IMP is 2.80× while MSDN is 3.18× slower than our Graph R-CNN. This is mainly due to the simplified architecture design in our model.

5.4 Ablation Study

In Graph R-CNN, we proposed two novel modules – relation proposal network (RePN) and attentional GCNs (aGCN). In this sub-section, we perform ablation studies to get a clear sense of how these different components affect the final performance. The left-most columns in Table 3 indicate whether or not we used RePN, GCN, and attentional GCN (aGCN) in our approach. The results are reported in remaining columns in Table 3. We also report object detection performance mAP@0.5 following Pascal VOC's metric [4].

In Table 3, we find RePN boosts `SGGen` and `SGGen+` significantly. This indicates that our RePN can effectively prune the negative connections between objects to achieve high recall for the correct relationships. We also notice it improves object detection significantly. For investigation, we show the per category object detection performance change when RePN is added in Fig. 4. For visual clarity, we dropped every other column when producing the plot. We can see that almost all object categories improve after adding RePN. Interestingly, we find the detection performance on categories like *racket*, *short*, *windshield*, *bottle*

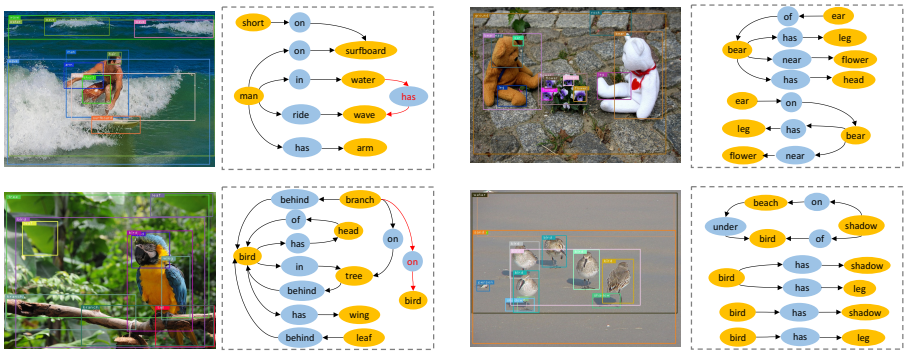

Fig. 5. Qualitative scene graph generation results from Graph R-CNN. Orange circles are the predicted subjects/objects and blue circles are the predicted relationships. Words in red font denote incorrect predictions.

are most significantly improved. This suggests that RePN might push features of related objects closer together, reducing competitiveness between detections of related classes. Since evaluating `PhrCls` and `PredCls` use the ground truth object locations, the number of relation pairs is already very small (typically less than 100). As a result, RePN has less effect on these two metrics.

By adding the GCNs into our model, the performance is further improved. These improvements demonstrate that the aGCN in our Graph R-CNN can captures meaningful contexts across the graph. We also compare the performance of our model with and without attention. We see that by adding attention on top of GCNs, the performance is further improved. This indicates that controlling the extent to which contextual information flows through the edges is crucial. These results align with our intuitions mentioned in the introduction. Fig. 5 shows generated scene graphs for test images. With RePN and aGCN, our model is able to generate more accurate scene graphs. The red arrow and word shows the failure predictions of the relationship in the generated scene graph.

6 Conclusion

In this work, we introduce a new model for scene graph generation – Graph R-CNN. Our model includes a relation proposal network (RePN) that efficiently and intelligently prunes out pairs of objects that are unlikely to be related, and an attentional graph convolutational network (aGCN) that effectively propagates contextual information across the graph. We also introdce a novel scene graph generation evaluation metric (`SGGen+`) that is more fine-grained and realistic than existing metrics. Our proposed approach outperforms existing approaches at scene graph generation, as evaluated using existing metrics as well as our proposed metric.

References

1. Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Lawrence Zitnick, C., Parikh, D.: Vqa: Visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2425–2433 (2015)
2. Dai, B., Zhang, Y., Lin, D.: Detecting visual relationships with deep relational networks (2017)
3. Das, A., Kottur, S., Gupta, K., Singh, A., Yadav, D., Moura, J.M., Parikh, D., Batra, D.: Visual dialog (2017)
4. Everingham, M., Van Gool, L., Williams, C., Winn, J., Zisserman, A.: The pascal visual object classes challenge 2012 results. In: See http://www. pascal-network. org/challenges/VOC/voc2012/workshop/index. html. vol. 5 (2012)
5. Gao, X., Xiao, B., Tao, D., Li, X.: A survey of graph edit distance. Pattern Analysis and Applications **13**(1), 113–129 (2010)
6. Girshick, R.: Fast r-cnn (2015)
7. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation (2014)
8. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn (2017)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2016)
10. Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y.: Relation networks for object detection (2018)
11. Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C.L., Girshick, R.: CLEVR: a diagnostic dataset for compositional language and elementary visual reasoning (2017)
12. Johnson, J., Krishna, R., Stark, M., Li, L.J., Shamma, D.A., Bernstein, M., Fei-Fei, L.: Image retrieval using scene graphs (2015)
13. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks (2017)
14. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., et al.: Visual genome: Connecting language and vision using crowdsourced dense image annotations **123**(1), 32–73 (2017)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks (2012)
16. Ladicky, L., Russell, C., Kohli, P., Torr, P.H.: Graph cut based inference with co-occurrence statistics (2010)
17. Li, Y., Ouyang, W., Wang, X.: Vip-cnn: A visual phrase reasoning convolutional neural network for visual relationship detection (2017)
18. Li, Y., Ouyang, W., Zhou, B., Wang, K., Wang, X.: Scene graph generation from objects, phrases and region captions (2017)
19. Liang, X., Lee, L., Xing, E.P.: Deep variation-structured reinforcement learning for visual relationship and attribute detection (2017)
20. Lin, C.L.: Hardness of approximating graph transformation problem. In: International Symposium on Algorithms and Computation. pp. 74–82. Springer (1994)
21. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection (2017)
22. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: SSD: Single shot multibox detector (2016)
23. Lu, C., Krishna, R., Bernstein, M., Fei-Fei, L.: Visual relationship detection with language priors (2016)

24. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines (2010)
25. Newell, A., Deng, J.: Pixels to graphs by associative embedding (2017)
26. Oliva, A., Torralba, A.: The role of context in object recognition. Trends in cognitive sciences **11**(12), 520–527 (2007)
27. Parikh, D., Zitnick, C.L., Chen, T.: From appearance to context-based recognition: Dense labeling in small images (2008)
28. Peyre, J., Laptev, I., Schmid, C., Sivic, J.: Weakly-supervised learning of visual relations (2017)
29. Rabinovich, A., Vedaldi, A., Galleguillos, C., Wiewiora, E., Belongie, S.: Objects in context (2007)
30. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection (2016)
31. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks (2015)
32. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
33. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions (2015)
34. Teney, D., Liu, L., Hengel, A.v.d.: Graph-structured representations for visual question answering (2017)
35. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks (2018)
36. Wang, P., Wu, Q., Shen, C., Dick, A., van den Hengel, A.: Fvqa: Fact-based visual question answering (2017)
37. Wang, P., Wu, Q., Shen, C., van den Hengel, A.: The vqa-machine: Learning how to use existing vision algorithms to answer new questions (2017)
38. Wu, Q., Shen, C., Wang, P., Dick, A., van den Hengel, A.: Image captioning and visual question answering based on attributes and external knowledge (2017)
39. Xu, D., Zhu, Y., Choy, C.B., Fei-Fei, L.: Scene graph generation by iterative message passing (2017)
40. Yang, J., Lu, J., Batra, D., Parikh, D.: A faster pytorch implementation of faster r-cnn. https://github.com/jwyang/faster-rcnn.pytorch (2017)
41. Zellers, R., Yatskar, M., Thomson, S., Choi, Y.: Neural motifs: Scene graph parsing with global context (2018)
42. Zhang, H., Kyaw, Z., Chang, S.F., Chua, T.S.: Visual translation embedding network for visual relation detection (2017)
43. Zhang, H., Kyaw, Z., Yu, J., Chang, S.F.: Ppr-fcn: weakly supervised visual relation detection via parallel pairwise r-fcn (2017)
44. Zhang, J., Elhoseiny, M., Cohen, S., Chang, W., Elgammal, A.: Relationship proposal networks. In: CVPR (2017)
45. Zhuang, B., Liu, L., Shen, C., Reid, I.: Towards context-aware interaction recognition for visual relationship detection. In: ICCV (2017)