Scene Graphs for Image Representation and Generation

Björn Bebensee [1]

Abstract

Representations of visual scenes often come in the form of unstructured data such as textual descriptions. Recent work has shown that many tasks can benefit from a more structured representation of semantic relationships within the image. In this paper we give a short introduction to scene graphs and their effectiveness. We show how they can be generated from images and present some of their applications in image retrieval and image generation.

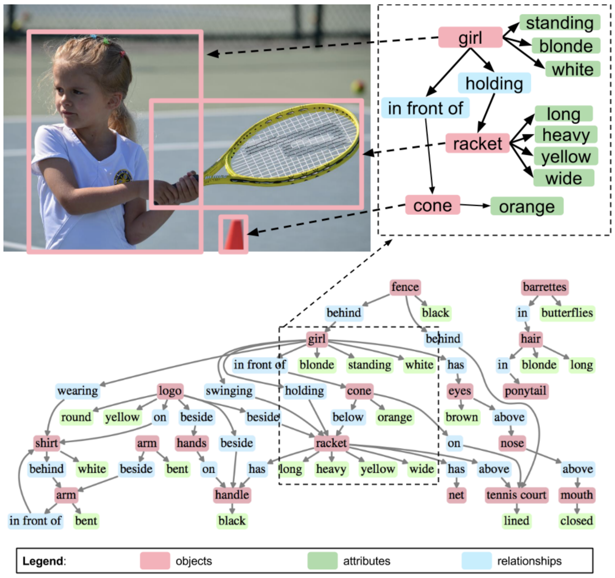


Figure 1. An example of a scene graph along with its grounding in the image (Johnson et al., 2015).

1. Introduction

In order to understand scenes depicted in images it is important to understand the semantic relationships between different objects in the image. Traditionally, there have been many text-based approaches to representing visual scenes (Gupta & Davis, 2008; Sadeghi & Farhadi, 2011; Zitnick et al., 2013). As text descriptions of images often might not to represent semantic relationships between objects in complex scenes well and as they do not offer any sort of visual grounding, a new line of research on scene graphs has become popular and achieved significant progress in recent years across various applications. Scene graphs allow for semantic reasoning on relationships in highly complex scenes and can be useful for many tasks such as image retrieval (Johnson et al., 2015), visual question answering (Teney et al., 2017) and image generation (Johnson et al., 2018). They offer structured representations of scenes and each node in the graph is visually grounded in an object. Edges in the graph represent relationships between objects such as positional attributes (“in front of”, “behind”) and actions (“holding”, “wearing”). They prove most effective for tasks that can benefit from such a strongly structured representation. In this paper we first define scene graphs in section 2, present a method to generate scene graphs from images in section 3, show how scene graphs can be used for image retrieval in section 4 and for image generation in section 5.

2. Scene graph representation

A scene graph is a type of graph that is grounded in an image and describes the relationships of different objects. More specifically, each vertex in the graph represents an object in an image such as a person, animal, house or car. The edges of the graph assign relationships to pairs of vertices. These relationships typically include actions (running, drinking, wearing), spatial relationships (in front of, behind) and comparisons (taller than, longer, same color as).

Formally, given a set of objects $\mathcal{O}$, a set of relationships $\mathcal{R}$ and a set of edges $\mathcal{E} \subseteq \mathcal{O} \times \mathcal{R} \times \mathcal{O}$, we define a scene graph $G = (\mathcal{O}, \mathcal{E})$. Any given object $o \in \mathcal{O}$ is grounded in the image described by the graph. That is, o is assigned a location (x, y) in the image and inherits its bounding box and class. Edges are labelled with relationships $r \in \mathcal{R}$ that describe how different vertices representing objects

[1] Department of Computer Science and Engineering, Seoul National University, Seoul, South Korea. Correspondence to: Björn Bebensee (2019-21343) <bebensee@snu.ac.kr>.

Topics in Artificial Intelligence, Department of Computer Science and Engineering, Seoul National University. Copyright 2019 by the author(s).

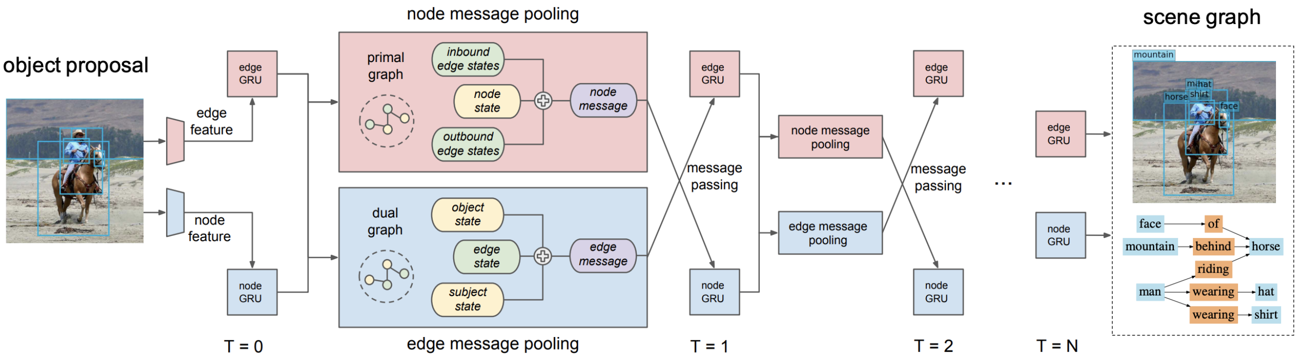


Figure 2. An illustration of the iterative message passing model for scene graph generation proposed by (Xu et al., 2017). The graph is divided in primal and dual graphs consisting of node GRUs and edge GRUs respectively. Messages are passed between the graphs iteratively until a prediction is made after N steps.

in the graph relate to one another. Each edge $e \in E$ is describes the relationship of two objects in the image, i.e. $e = (o_1, r, o_2)$ for two vertices $o_1, o_2 \in O$ and a relationship $r \in \mathcal{R}$. Each object o_i can be considered a pair of its class $c_i \in \mathcal{C}$ and a set of attributes $A_i \subseteq \mathcal{A}$ and is written as $o_i = (c_i, A_i)$ (Johnson et al., 2015; Newell & Deng, 2017).

We formally define the grounding of objects in the image as a map $\gamma : \mathcal{O} \rightarrow B$ where B is a set of candidate bounding boxes so γ associates each object $o \in \mathcal{O}$ with a region in the image.

3. Generating scene graphs from images

In many cases we need to extract a scene graph from a given image first so that we can then use it for applications in image retrieval, visual question answering and others. Apart from scene graphs, there has been some previous work on finding semantic relationships between objects in an image. However, much of this line of work has focused on finding these relationships between pairs of objects locally rather than relationships in the image globally. In many cases the interpretation of semantic relationships between objects may depend on different parts of the image. One such way of generating scene graphs from images that takes into account global information is proposed by Xu et al. (Xu et al., 2017). Instead of separately detecting and then recognizing objects to find their respective classes and attributes, they propose an end-to-end model that uses iterative message passing to pass contextual information between a pair of bipartite subgraphs.

For the inference algorithm they use a set of bounding box proposals generated by a Region Proposal Network (RPN). For each of these proposals they refine the bounding box coordinates further and infer an object class as well as object-object relationships. Given some set

$$x = \{x_i^{cls}, x_i^{bbox}, x_{i\rightarrow j} \mid i = 1 \dots n,\ j = 1 \dots n,\ i \neq j\}$$

where there are n distinct bounding box proposals, x_i^{cls} denotes the class, x_i^{bbox} denotes the change relative to the proposal of the i-th bounding box and $x_{i\rightarrow j}$ denotes the relationship between the i-th and j-th bounding box, they formulate the problem of finding such a scene graph as finding the optimal x^* that maximizes the probability function

$$\arg\max_{x^*} \prod_{i \in V} \prod_{j \neq i} \Pr(x_i^{cls}, x_i^{bbox}, x_{i\rightarrow j} \mid I, B_I). \quad (1)$$

Here, B_I are the bounding box proposals given by the RPN for an image I. In order to do the inference they implement this model using GRUs and each node and edge has their own hidden state respectively but all nodes (all edges) share the weights of the GRU, i.e. they have the same update rule. The hidden states are initialized using features extracted from the bounding box regions of the image. Then, at each time step message are passed between neighboring GRUs. More specifically, they propose to split the scene graph into a primal graph containing all the node GRU hidden states and a dual graph containing all the edge GRU hidden states for message pooling. Instead of passing messages through the graph, they then pass messages between these two subgraphs. Each hidden state is updated using a message pooling function that learns adaptive weights for incoming messages. This is repeated iteratively for N steps after which bounding box changes, object classes and object relationships are predicted based on the hidden states. An illustration of this iterative message passing method can be seen in figure 2.

Xu et al. evaluate their proposed iterative message passing method on the Visual Genome dataset. They find that their model outperforms the baseline by Lu et al. which performs relationship detection on objects in images using priors from natural language (Lu et al., 2016). Additionally, they test their model on support relation prediction using the NYU Depth v2 dataset. In this task the model predicts what an object in an image is supported by, i.e. the floor or a piece of furniture. They achieve state-of-the-art performance on this task using their model as well. Overall, they find that their model performs very well and is effective in creating scene graphs in an end-to-end fashion.

4. Image Retrieval using scene graphs

Johnson et al. first introduced the concept of scene graphs to better capture semantic contents in the image domain and to provide a natural language abstraction in the form of graphs to describe images (Johnson et al., 2015). In the image retrieval task using scene graphs they use a scene graph as a query to retrieve the image that best matches the scene described by the graph. They score images according to their agreement with the scene graph given the best possible grounding. To model the distribution of all possible groundings they construct a conditional random field (CRF) and perform maximum a priori (MAP) inference to find the most likely grounding.

Given a scene graph $G = (\mathcal{O}, \mathcal{E})$ and a grounding γ with a set of bounding boxes B, the objective function

$$\gamma^* = \arg\max_{\gamma} \prod_{o \in \mathcal{O}} P(o \mid \gamma_o) \prod_{(o,r,o') \in \mathcal{E}} P(\gamma_o, \gamma_{o'} \mid o, r, o') \quad (2)$$

where γ_o is shorthand notation for $\gamma(o)$ for an object $o \in \mathcal{O}$. Here the term $P(o \mid \gamma_o)$ describes the probability of a bounding box γ_o belonging to the object class c with attributes A for $o = (c, A)$. The second term $P(\gamma_o, \gamma_{o'} \mid o, r, o')$ models how well two bounding boxes $\gamma_o, \gamma_{o'}$ in the image model the relationship between objects o, o' given by (o, r, o'). Johnson et al. use a R-CNN to obtain the probabilities for object class and attributes (*unary potentials*) and a Gaussian mixture model to obtain the probability distribution of relationships over different bounding boxes in the image (*binary relationship potentials*).

Johnson et al. train the model using a newly collected dataset of real-world scene graphs. Specifically, the dataset contains 5,000 images from the intersection of the YFCC100m and COCO datasets. To obtain human-generated scene graph annotations for all these images they employ Amazon Mechanical Turk. In total, the dataset contains 93,000 object instances of 6,745 object classes, 110,000 attribute instances of 3,743 types of attributes and 112,000 instances of 1,310 types of object relationships.

To evaluate the model they first evaluate it on full scene graphs in an ablation study where they find that the model with both the unary potentials and binary relationship potentials as described above performs best. Then, they compare the scene graph models to a CNN-based model which uses the L_2 distance of the output layer features, as well as GIST descriptor and SIFT descriptor-based models finding the closest images to the query image (and the query scene graph for the proposed scene graph image retrieval model). They find that the model is able to retrieve images fitting the full, complex scene graph query. They also test the model on smaller scene graphs such representing queries such as “smiling man wearing hat” which are easily interpretable and which you might encounter on an image search engine. They find that their model performs better and retrieves images which are closer to the query image in terms of recall than the competing models due to the additional semantics that are provided by the scene graph and not easily captured from the image by CNN, GIST and SIFT models. Lastly, Johnson et al. evaluate their scene graph-based models (using both unary potentials and binary relationship potentials or using just object or attribute unary potentials) for object localization in terms of median IoU. Again they find that the model using both the unary potentials and the binary relationship potentials performs best, as the additional context provided by the model appears to help localize ambiguous objects that appear in many different shapes as well as rare objects that do not appear often.

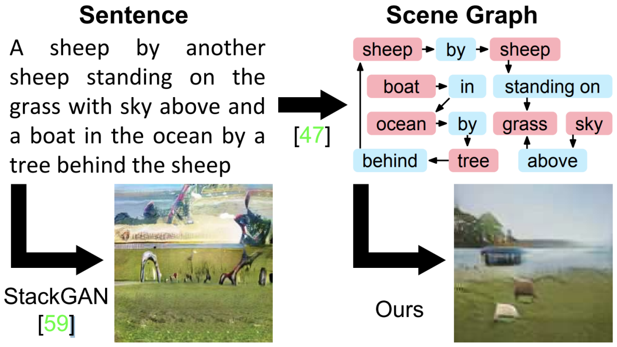

Sentence
A sheep by another sheep standing on the grass with sky above and a boat in the ocean by a tree behind the sheep
[47]
Scene Graph
sheep
by
sheep
boat
in
standing on
ocean
by
grass
sky
behind
tree
above
StackGAN
[59]
Ours

Figure 3. An example image generated by the scene graph image generation model proposed by (Johnson et al., 2018).

5. Generating images from scene graphs

Image synthesis is a task in which we would like to generate a realistic image of a scene depicted in some sort of abstract description like a natural language sentence. Another way to describe the scene that we want to generate an image of are scene graphs. Johnson et al. propose such a method that can generate images from complex scene graphs (Johnson et al., 2018). Previous work has made great progress in text to image synthesis, however these methods often fail for longer and more complex descriptions of scenes. As scene graphs provide additional semantic information that is difficult to extract or even entirely missing from sentences descriptions, they may be the ideal foundation to generate images depicting complex scenes.

Johnson et al. structure their image generation method into

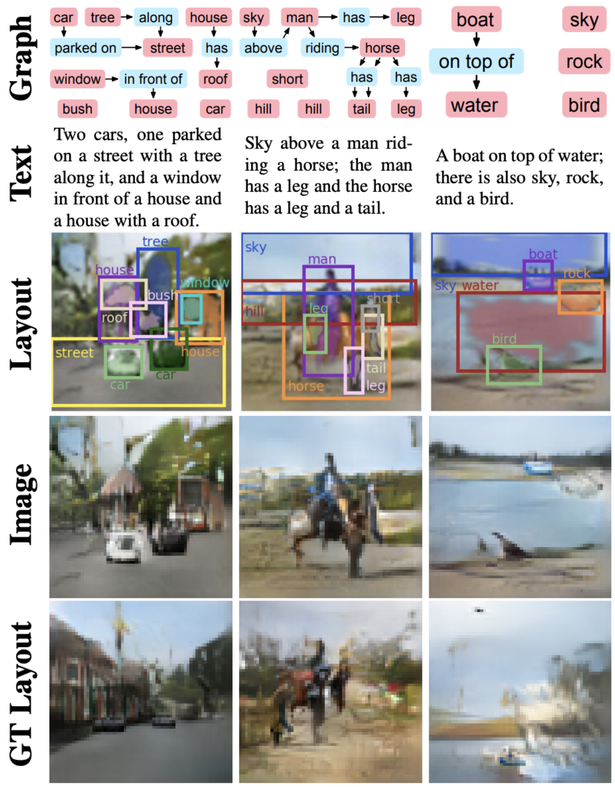
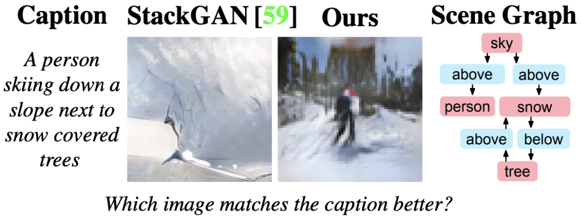


Figure 4. Examples for the image generation process including the scene layout generated by the model (Johnson et al., 2018).


Caption
StackGAN [59]
Ours
Scene Graph
A person skiing down a slope next to snow covered trees
sky
above
above
person
snow
above
below
tree
Which image matches the caption better?

User choice	332 / 1024 (32.4%)	**692 / 1024** **(67.6%)**

Figure 5. Users prefer images generated by the model proposed by Johnson et. al than ones generated using StackGAN across 1,024 validation image pairs (Johnson et al., 2018).

three steps. First the input scene graph needs to be processed such that it is in an adequate form for image generation. For this task they use *Graph Convolutional Networks* which convolute the graph by propagating information along edges of the graph. Their graph convolutional layer updates all edges with the same update rule: for each object $o_i \in \mathcal{O}$ and all relationships $o_i, r, o_j \in \mathcal{O}$ (i.e. for all nodes and edges in the scene graph) there are vectors $v_i, v_r \in \mathbb{R}^{D_{in}}$ and they are updated with functions g_s (subject vectors), g_p (predicate vectors), and g_o (object vectors) to $v_r' = g_p(v_i, v_r, v_j)$ for all relationships and to $v_i' = h(V_i^s \cup V_i^o)$ for all objects. Here h is pooling function to obtain a single output vector and V_i^s, V_i^o are candidate vectors computed over vectors of all objects and edges a given node is connected to. These candidate vectors are computed as follows:

$$V_i^s = \{g_s(v_i, v_r, v_j) : (o_i, r, o_j) \in E\} \quad (3)$$

$$V_i^o = \{g_s(v_i, v_r, v_j) : (o_i, r, o_j) \in E\} \quad (4)$$

The functions g_s, g_p, g_o are implemented using a MLP. In the second step, after a series of graph convolution layers, the convoluted graph contains embedding vectors at every node and edge representing information that has been aggregated across relationships in the scene graph. Johnson et al. use these embedding vectors to obtain a scene layout that roughly describes the positions of objects in the image and how they are related to each other. They use an *object layout network* which, for each embedding vector, computes a mask and a bounding box and then combines all of these object layouts into one scene layout. Third, they can utilize this scene layout to condition a *Cascaded Refinement Network* (CRN) which is trained to generate a realistic image that respects the object positions given in the scene layout. In order to generate the image, a series of convolutional refinement modules iteratively, using the scene layout and the previous modules output, generates a higher resolution image output. For the first module Gaussian noise is used as input. Two final convolutional layers generate the final image output from the last module's output. To train this model to output realistic looking images, Johnson et al. employ two discriminator networks, one which makes certain that images look realistic overall and another one that makes certain each object in the image looks realistic. The entire model is trained end-to-end.

They evaluate the model on Visual Genome and COCO and conduct two user studies using Amazon Mechanical Turk. They find that for a given caption users prefer images generated using their proposed method on the scene graph over an image generated by StackGAN (Zhang et al., 2017) using the caption (see figure 5). Similarly, users were able to identify recognizable objects in images generated by their method much better than in images generated by StackGAN. Qualitatively the results are good and images generated keep the structure described by the scene graph well. An example can be seen in figure 3. Figure 4 includes some examples of scene layouts and the corresponding images generated by the model.

6. Concluding Remarks

We have introduced scene graphs and shown that they provide a powerful and effective tool for representations of visual scenes. They can be applied to a variety of domains such as image retrieval and image generation. Many tasks can benefit from such a structured representation that not only provides semantic relationships between objects but also grounds these objects in the image. Beyond the short introduction to applications of scene graphs offered in this paper, they may also provide useful in visual question answering and other fields in the future.

References

Gupta, A. and Davis, L. S. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *European conference on computer vision*, pp. 16–29. Springer, 2008.

Johnson, J., Krishna, R., Stark, M., Li, L.-J., Shamma, D., Bernstein, M., and Fei-Fei, L. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3668–3678, 2015.

Johnson, J., Gupta, A., and Fei-Fei, L. Image generation from scene graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1219–1228, 2018.

Lu, C., Krishna, R., Bernstein, M., and Fei-Fei, L. Visual relationship detection with language priors. In *European Conference on Computer Vision*, pp. 852–869. Springer, 2016.

Newell, A. and Deng, J. Pixels to graphs by associative embedding. In *Advances in neural information processing systems*, pp. 2171–2180, 2017.

Sadeghi, M. A. and Farhadi, A. Recognition using visual phrases. In *CVPR 2011*, pp. 1745–1752. IEEE, 2011.

Teney, D., Liu, L., and van den Hengel, A. Graph-structured representations for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2017.

Xu, D., Zhu, Y., Choy, C. B., and Fei-Fei, L. Scene graph generation by iterative message passing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5410–5419, 2017.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. N. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5907–5915, 2017.

Zitnick, C. L., Parikh, D., and Vanderwende, L. Learning the visual interpretation of sentences. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1681–1688, 2013.