Scene Graph Generation by Belief RNNs

Roei Herzig, Moshiko Raboh
School of Computer Science, Tel Aviv University
roeiherzig@mail.tau.ac.il, shikorab@gmail.com

Abstract

Understanding and describing the scene beyond an image has great value. It is a step forward from recognizing individual objects and their relationships in isolation. In addition, Scene graph is a crucial step towards a deeper understanding of a visual scene. We present a method in an end-to-end model that given an image generates a scene graph including nodes that corresponds to the objects in the image and an edges corresponds to relationship between objects. Our main contribution is introducing a novel deep structure prediction module - **Belief RNNs** - that performs learning on a large graphs in a very efficient and generic way.

1 Introduction

Extracting semantics from images is one of the main goals of computer vision. Each image can be interpreted as a knowledge graph that contains objects. The ability to predict a graph connections in a single image is crucial for understanding the relations and the semantics. Humans use those graphs connections to perform reasoning which its essential for intelligent behavior.

Scene graph, as defined by Johnson et al [3] (Figure 1), is a structured representation of an image, where the nodes in the scene graph corresponds to object bounding boxes with their object categories and edges (O_i, O_j) corresponds to the pairwise relationship between object O_i and object O_j. We consider the problem of scene graph to generate a graph that reflects accurately the connections in the image.

Figure 1: An example of a Scene Graph from [2].

The major contribution of this work is once one extracting the unary and pairwise potentials (e.g. the features from each objects and the features from predicates), our **Belief RNNs** module shows a significant boost for learning a knowledge graph in an iterative manner. The main advantages of this paper are (a) Creating a NN module to perform an end-to-end learning system that capable of using the whole context information for a large graphs in an efficient computational manner. (b) Introducing a new deep learning meta algorithm for mean-field approximation. (c) Our method that generates knowledge graph from an image outperforms the baselines.

In summary, we are proposing a new iterative deep learning structure prediction approach to process the mutual information between objects and relationships and perform message passing for generating more accurate graphs in each iteration.

2 Related work

Scene Graph generation using language priors: Visual relationship detection has been a very hot topic which drawn much attention recently. It involves detection of objects that occur in an image as well as understanding the interaction between them. [10] shows it can be exploit to use language priors to boost a relation detection and [1] proposed a model which improves prior work by leveraging language priors from semantic word embedding to fine-tune the likelihood of predicated relationship. [1] shows that even though the semantic space of possible relationships is much larger compared to classifying objects, the facts that we probably encounter each part of the relationship separately during the training and the relationships might be semantically similar to each other will assist. [1] **is also used as our baseline model, to get some intuition** about the combination of language and visual models.

Scene Graph generation using message passing: [2] takes the task of describing a scene through object and relationship prediction one step forward. [2] suggesting that understanding a visual scene goes beyond recognizing an individual objects and therefore suggest end to end model that instead of inferring each component in isolation, the model passes messages containing contextual information between a pair of GRU's, and it iteratively refines the feature map that later will be used to predict the objects and relationships. Our work is inspired by this concept of using end-to-end model instead of inferring each component in isolation.

Visual reasoning: Our suggested model is also inspired by Relation Networks [7] and Graph Search Neural Network [9]. Both of them show how to use relational reasoning for VQA or image classifications tasks as they are using structure knowledge for reasoning. They investigate the use of structured prior knowledge in the form of knowledge graphs and show that using this knowledge improves performance on image classification. Another work that uses interactions capturing between pairs of objects in an end to end manner has been done by [8].

One of the main difference between our work and [2, 8] is our ability to perform learning on the whole graph, including negative predicates. Our **Belief RNNs module** overcomes the fact that 95% of the predicates are negatives, as in real life. Moreover, our model are inferring directly on the beliefs (not taking into account the visual features), which is more generic.

3 Scene Graph Model

Our model contains two main parts: a **feature extraction module** and a **Belief RNNs module**. The feature extraction module is responsible for extracting visual features and probabilities for each object and for each predicate. The **Belief RNNs module** is a deep structure prediction module which performs learning on a large graphs or networks in an efficient and general way and will be in details later on.
The mathematical formulation of our suggested model, aimed to generate scene graph, can be represented as an inference task. Denote x as the representation of the scene graph and I as the input image:

$$x = \{x_i^{object-class}, x_{i \to j}^{predicate-class}, I | i = 1..N, j = 1..N\} \tag{1}$$

Where $x_i^{object-class}$ represents the predicated object label (i.e man, car, etc.) and $x_{i \to j}^{predicate-class}$ represents the relationship predicate label predicted between objects i and j. N is the number of objects in image I.

The inference task can be described as:

$$x^* = \arg\max_x Pr(x|I, B_I) \tag{2}$$

$$Pr(x|I, B_I) = \prod_{i=1}^{N} \prod_{j=1}^{N} Pr(x_i^{object-class}, x_{i \to j}^{predicate-class} | I, B_I) \quad (3)$$

where B_I is the set of bounding-boxes of objects per image I.

We will use mean field approximation to obtain a simpler inference task which will be used in the Belief RNNs module.

3.1 Feature extraction module

The main purpose of this module is extracting features and probabilities for each object and each predicate and sending them to the next module. As we mentioned earlier, a scene graph is a structured representation of an image where each object can be seen as a node and each predicate is an edge between two objects. Therefore, this module is used for extracting the unary and pairwise potentials which will be used later on by **the Belief RNNs** module to perform a graph learning.

The module contains two ResNet50 networks [5] for objects and predicates separately. The objects ResNet50 network is trained to predict objects labels (e.g car, person, etc.) and the predicate ResNet50 network is trained to predict predicates labels (e.g has, on ,of, etc.). The unary and pairwise beliefs will be taken from the last layer before the soft-max function.

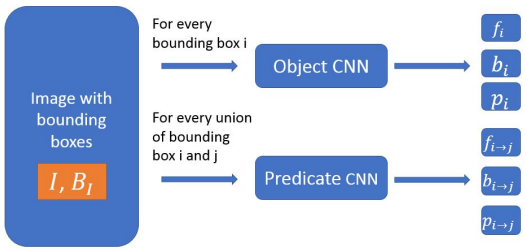

Figure 2: **Feature extraction module** process: from an image I and set of bounding-boxes B_I the two networks extract objects features f_i, objects beliefs (likelihood) b_i, objects probabilities p_i, predicates features $f_{i \to j}$, predicates beliefs (likelihood) $b_{i \to j}$ and predicates probabilities $p_{i \to j}$.

3.2 Belief RNNs module

The main idea behind **Belief RNNs module** is to perform a graph structure prediction with an improved belief in each step. The module contains a sequence of RNNs that inputs predicates and objects beliefs (likelihoods), and output a new improved predicates and objects beliefs as in Figure 3. The pipeline's outcome is the most improved belief.

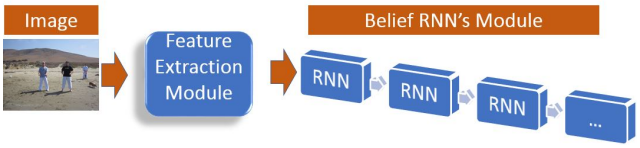

Figure 3: Full pipeline process: (a) the **feature extractor module** is receiving an image as an input. (b) the **Belief RNNs module** receives the beliefs from **feature extractor module**. (c) each RNN cell get predicates and objects beliefs, and outputs improved beliefs.

The input of each RNN cell are the beliefs from the previous RNN cell, and the output of each cell is a new improved beliefs which have been updated by the NN's in each RNN. The initial objects and predicates beliefs are taken from the likelihood of the last layer from **feature extraction module**. Once the input beliefs are inserted to the RNN, we are collecting the features per object and predicate (Feature Collector Unit), and then process them with a shared objects and predicates NN's. Those

NN's are FC networks, which given input, are calculating an updated potentials, which are the sum of the NN's output and the input belief (residual). At last, the RNN cell forward both the objects and predicates beliefs to the next RNN cell to get a new prediction of beliefs. The full updating process depicts in Figure 4.

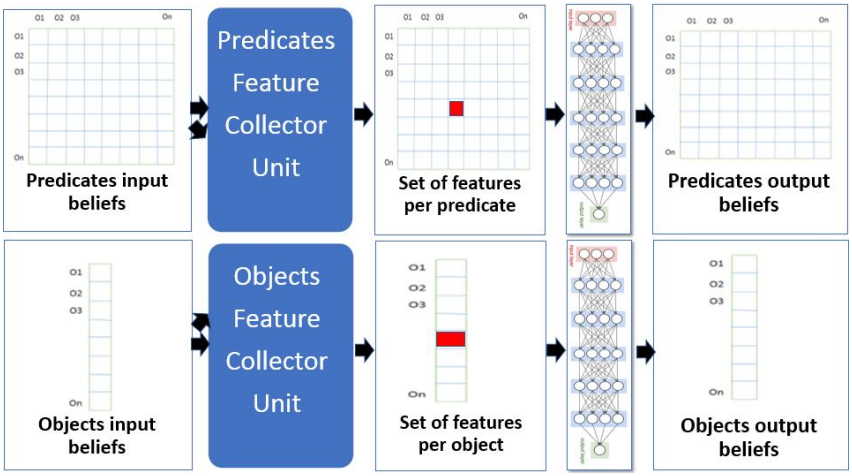

Figure 4: Process of updating the beliefs in a RNN cell. The first row is introducing the belief of predicates update while the second is the belief of objects update. The pipeline is as follows: (a) First we are getting an initial beliefs with N^2 predicates and N objects per image. (b) Collecting features for objects and predicates in Feature Collector Unit. After this unit, we have a new set of features per object and per predicate (e.g. red box). (c) Forward those features to objects NN and predicates NN to get a new beliefs prediction.

The Feature Collector Unit (FCU) is responsible for extracting objects and predicates beliefs (likelihood). The unit gets both objects and predicates beliefs input, and use this information to update objects and predicates NN's. There are differences between the predicates FCU and objects FCU. While the objects are getting the most valuable information for object classification, the predicates getting the most valuable information for predicate classification. For example, in Figure 5 car is near roadway and the roadway is near an electric pole. Therefore, we could use this global information to know that if there is a car, maybe an electric pole will be near to it. Meaning, all the other objects that have a relation to car and roadway, are also very important to have reasoning.

Figure 5: As we defined, a relationship r is a triplet of a predicate and two objects: r = $\langle O_i, R_{i,j}, O_j \rangle$while O_i and O_j are objects, and $R_{i,j}$ is the predicate. For example, image with different objects and relations between them such as: <car, near, roadway> or <window, on, building>.

The collection features process for a relationship r = $\langle O_i, R_{i,j}, O_j \rangle$, as can be seen in Figures 6, 7 is going as follows:
The predicates Feature Collector Unit will collect the following features for predicate $R_{i,j}$ between object O_i and object O_j :

1. Belief of the predicate $R_{i,j}$.
2. Belief of the objects O_j.
3. Belief of the subject O_i.
4. Belief of all other predicates $R_{i,k}$ that O_i is their subject.
5. Belief of all other predicates $R_{k,j}$ that O_j is their object.
6. Belief of all other objects (max over all objects beliefs).

The objects Feature Collector Unit will collect the following features for object O_i:

1. Belief of all other predicates $R_{i,k}$ that O_i is their subject.
2. Belief of all other predicates $R_{k,i}$ that O_i is their object.
3. Belief of all other objects (max over all objects beliefs).
4. Belief of object O_i.

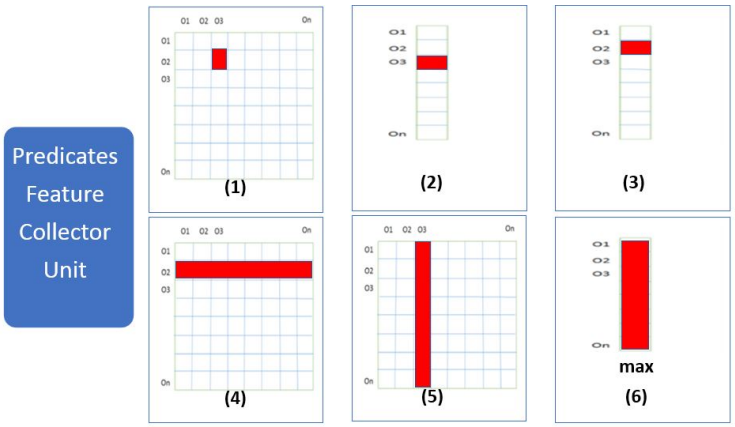

Figure 6: An example of collection features of the **predicates** Feature Collector Unit for predicate $R_{2,3}$ between subject O_2 and object O_3 as defined above (the red box are the likelihoods that have been collected).

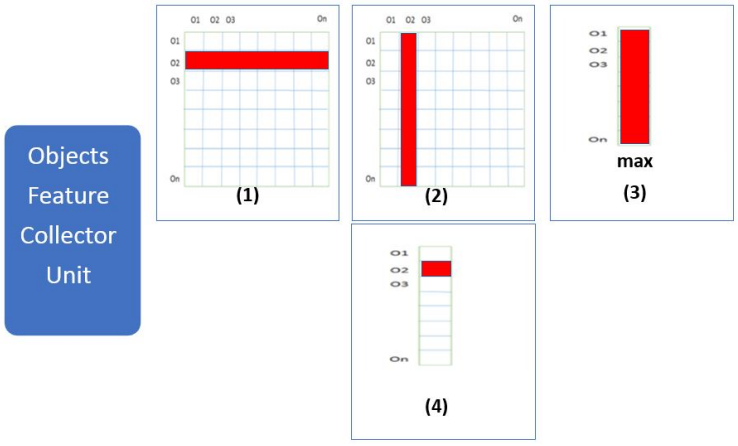

Figure 7: An example of collection features of the **objects** Feature Collector Unit for O_2 as defined above.

4 Experiments

4.1 Training Details

During the experiments we trained few derivatives of our model: single step RNN, dual step RNN, single step RNN with language prior (still in progress) both for SGCls and PredCls 4.3 (in PredCls the objects labels used for input). We trained the **Belief RNNs module** with initial learning rate of 0.01, decreasing by a factor 0.5 every 10 epochs and a batch size of 100 images. The number of epoch is 50.

We trained on a positive-negative ratio of 1:3 in the **feature extractor module**, therefore, we are also aiming to this ratio of 1:3 per image in the loss by factoring the negative predicates. The loss combined both cross-entropy of predicates and objects while the objects loss is multiplied by a hyper-parameter of 0.1. The total loss is a weighted sum of each RNN cell outputs. We also use an L2 penalty of $1e^{-6}$ as a regularization.

4.2 Dataset

The dataset we used for training and evaluation is the Visual Genome dataset [4]. Visual Genome consists of 108,077 images annotated with object detections and object-object relationships. Its annotated with average 13.5 objects and 15 predicates per image and total 75,729 unique object categories and 40,480 unique predicates. To make a direct comparison to prior work by [1, 2], we use the same preprocessed version data as they do, which made by Xu et al. [2]. As a results, its annotated with average 12 objects and 7 predicates per image. We used for evaluation, the same 150 object categories and 50 predicates categories as [1, 2] do. As well as, the same training and testing data split as defined by them.

4.3 Setup

To avoid penalizing extra detections that may be correct but missing an annotation, we used Recall@k, a standard evaluation metric for scene graph which measures the fraction of ground-truth tuples to appear in a set of k proposals. There are three different problem settings as [2, 8] defined and used:
SGGen: Detect and classify all objects and predict the predicates between them.
SGCls: Ground-truth object boxes are provided, classify them and predict their predicates.
PredCls: Boxes and labels are provided for all objects, predict their predicates.

4.4 Results

We compare our results to [1, 2, 8]. For that matter, we use the exactly the same dataset and labels categories as [1, 2] while [8] use different sets of labels categories. Currently, as described, our model does not include RPN in the **feature extractor module**.

Table 1: Results on Visual Genome

	SGGen(w/ RPN)		SGCls		PredCls	
	R@50	R@100	R@50	R@100	R@50	R@100
Lu et al. [1]	0.3	0.5	11.8	14.1	27.9	35.0
Xu et al. [2]	3.4	4.2	21.7	24.4	44.8	53.0
Deng et al. [8]	9.7	11.3	**26.5**	30.0	**68.0**	**75.2**
Our model [0 RNNs]	-	-	14.7	20.2	24.8	36.2
Our model [1 RNNs]	-	-	21.4	29.3	42.2	53.4
Our model [2 RNNs]	-	-	**22.1**	**30.1**	42.75	**54.6**

Table 2: Predicate classification R@5

predicate	[1]	[2]	ours	predicate	[1]	[2]	ours
on	**99.83**	99.17	99.2	under	25.32	56.93	**65.5**
has	97.72	96.47	**99.7**	sitting on	49.48	57.01	**76.6**
in	73.56	**88.77**	83.6	standing on	51.43	**67.01**	45.8
of	88.59	96.18	**99.6**	in front of	31.52	**64.63**	52.0
wearing	98.32	98.01	**98.7**	attached to	11.81	27.43	**33.3**
near	87.46	**95.14**	94.3	at	57.73	**70.00**	29.4
with	29.42	88.00	**90.0**	hanging from	0.00	0.00	0.00
above	47.48	**70.94**	67.7	over	4.17	0.69	**20.0**
holding	55.67	**82.80**	77.5	for	5.61	11.21	**12.5**
behind	76.43	**84.12**	81.3	riding	82.03	**91.18**	69.7
no relationship	-	-	**99.5**				

4.5 Discussion

Our results are competitive compare to [1, 2, 8]. It seems that our results are lower than [8]. However, as mentioned, [8] and us use different set of labels categories which makes it hard to compare with. In addition, we didn't fully exploit the dataset. Currently the **Belief RNNs module** use just 20% from the data available for training and we stop training the **Belief RNNs module** due to over-fitting. As we can see, the first step of **Belief RNNs module** gives a boost of 9.1%, and step 2 gives a boost of 0.7%.

Analyzing the results per predicates reveals that our results, just for the less popular predicates, are lower compare to [2]. Therefore, it supports the hypothesis that adding language prior to our model might give additional boost to results.

Moreover, we analyzed few images and deduce the following:

1. The **Belief RNNs module** strengthen the score to specific predicate in case few objects have the same predicate. For example, in image that has few glasses on the table, the model gives higher score to predicate "on" (recognizing the object "table" and all the other objects are on it).
2. The model learns that a tuple of subject and object usually fits just to few predicates, therefore, the model strengthen those predicates (only the ones that got as an option from **feature extractor module**).
3. The model overcomes the symmetric issue of the **feature extractor module**. For example, it will gives high score to <man, has, shirt> and low score to <shirt, has, man>.
4. Misclassification of the object by the **feature extractor module** might be fatal for the **Belief RNNs module** because it does have access to visual feature for a refinement.

5 Conclusion

The task of scene graph classification is a challenging task. Much of the information required to classifying a specific relationship is hidden in context of the image. Therefore, there is a need to extract the hidden information not just from the bounding boxes, but also, to infer it from other relationships and objects in the image. In addition, there is also a need to deduce which relationships will be most probably tagged by a human. Our model, propose a generic and efficient way to propagate this global context information through scene graph when classifying each relationship or object which captures semantic content in the image (which iteratively refined in each RNN cell).

6 Future Work

While this work has shown the potential to perform reasoning between relations on a graph structure, many opportunities for extending the scope of this work remain.

- This work differs from other works by didn't adding any **language priors** to the model, which can also add an additional value. From our initial experiments, using embedding language added a small improvement to the model and in certainly, additional work needs to be done in this directions.
- Adding more data to training which will resolve the dataset over-fitting problem.
- Full analysis of the experiments.
- **Belief RNNs module** should be improved by changing to a GRU or a LSTM unit which will add gating operations.
- A memory networks and attention mechanism could be considered.
- The predicates FC network could be replaced by a more sophisticated neural network such as GCN [13] which learns on graph-structured data better than usual FC networks.
- Connecting each RNN cell to another RNN cells, as in DenseNet [6]. The main reason to use skip connections in our case is because of the feature reuse. From different stages, we have different set of features, therefore the module best updating message can be a mixture of those stages.
- **Feature extraction module** could be extended with a detector such as [11, 12] including RPN.
- **Belief RNNs module** could also taking into account visual features extracted from the Feature extraction module.

References

[1] C. Lu, R. Krishna, M. Bernstein and L. Fei-Fei. Visual relationship detection with language priors, *In European Conference on Computer Vision, 2016.*

[2] Danfei Xu, Yuke Zhu, Christopher B. Choy and Li Fei-Fei. Scene Graph Generation by Iterative Message Passing *, In Computer Vision and Pattern Recognition, 2017.*

[3] J. Johnson, R. Krishna, M. Stark, L. J. Li, D. A. Shamma, M. S. Bernstein, and L. Fei-Fei. Image retrieval using scene graphs. *In IEEE Conference on Computer Vision and Pattern Recognition, 2015.*

[4] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz,S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. Bernstein, and L. Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *In arXiv, 2016.*

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *, In Computer Vision and Pattern Recognition, 2016.*

[6] G. Huang, Z. Liu, and K. Q. Weinberger. Densely connected convolutional networks. *, In Computer Vision and Pattern Recognition, 2017.*

[7] A. Santoro, D. Raposo, D. G. T. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, , and T. Lillicrap. "A simple neural network module for relational reasoning," *, CoRR, vol. abs/1706.01427, 2017. [Online]. Available: http://arxiv.org/abs/ 1706.01427.*

[8] Newell Alejandro, and Jia Deng. "Pixels to Graphs by Associative Embedding." *arXiv preprint arXiv:1706.07365 (2017).*

[9] Marino Kenneth, Ruslan Salakhutdinov, and Abhinav Gupta. "The More You Know: Using Knowledge Graphs for Image Classification." *arXiv preprint arXiv:1612.04844 (2016).*

[10] Atzmon Y., Berant J., Kezami V., Globerson A. and Chechik, G., 2016. "Learning to generalize to new compositions in image understanding." *arXiv preprint arXiv:1608.07639.*

[11] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." *arXiv preprint arXiv preprint arXiv:1612.08242 (2016).*

[12] Ren Shaoqing, et al. "Faster R-CNN: Towards real-time object detection with region proposal networks." *Advances in neural information processing systems. 2015.*

[13] Kipf Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." *arXiv preprint arXiv:1609.02907 (2016).*