Indian Institute of Technology, Madras

DDP Interim report

Scene Understanding using a Semantic graph

Author:

Mohan Bhambhani
CS13B036

Supervisor:

Prof. B. Ravindran

December 22, 2017

Contents

Abstract

Deep Learning has been very successful in the recent years in various visual tasks such as object detection. But truly understanding a image also requires understanding of complex interactions between objects and their attributes. Complex artificial intelligence systems struggle in capturing minute details entailed in a image. Tasks such as visual relationship detection, visual attribute detection, scene graph generation, detailed image description, etc. have been defined to test scene understanding.

Most of the state-of-the-art methods in scene understanding tasks ignore the semantic constraints of interactions between objects of different types. We propose to incorporate knowledge of the constraints from a semantic graph (that contains such constraints) by getting rich embeddings using state-of-the-art network representation learning techniques.

1 Introduction

In the recent few years deep learning has been very successful in various supervised learning tasks in various domains like vision and natural language processing. Convolutional neural networks have produced unbelievable results in vision. Recurrent neural networks have transformed our approach towards complex natural language processing tasks such as machine translation. But, still we have a long way to go before we have human comparable AI systems. In various tasks such as question answering, where a systems' intelligence is truly tested, there is a lot of room for improvement.

State-of-the-art perception models have almost perfected detection of individual objects in an image. But, completely comprehending an image is much more than just object detection. For better understanding of an image one must be able to perceive subtle interactions between objects. These models fail to recognise rich semantic relationships. Here too, recent work has shifted focus to tasks like generating image descriptions [1], visual relationship detection [2] and scene graph generation [3] for more deep understanding of images.

Structuring visual scene in form of a scene graph [4] has led to better understanding of images. It captures objects, their relationships and their attributes. An example scene graph has been shown in the Figure 1. Such representation provides more information for various visual tasks like image retrieval [4] and also puts forward higher level visual task of generating this given a image. This has spurred interest in detection of various components of the scene graph like relationships and attributes. As defined in [3], *scene graph* is a visually-grounded graph over the object instances in an image, where the edges depict their pairwise relationships.

Scene graph representation has been used in various visual tasks like image retrieval [4], 3D scene synthesis [5], Visual question answering [6]. But the scene graph for these models is obtained from external sources like annotations. Recent work [2] [7] [3] [8] [9], has focused on the problem of scene graph generation or more specifically visual relationship detection. The final output scene graph of these models has objects as nodes, relationships as edges. Also, the objects in the output are visually grounded on the input image.

Visual relationship detection is a difficult problem. The state-of-the-art has 16.09% of Recall@50. There are two natural approaches [7] to tackle this problem. Earlier [10] this was considered as a classification task with combination of object and relationship predicates as classes. With even 1000 object categories and 200 predicate classes this will scale to 2×10^8 classes. Recent approaches [7] [3] [9], consider it as classification task of

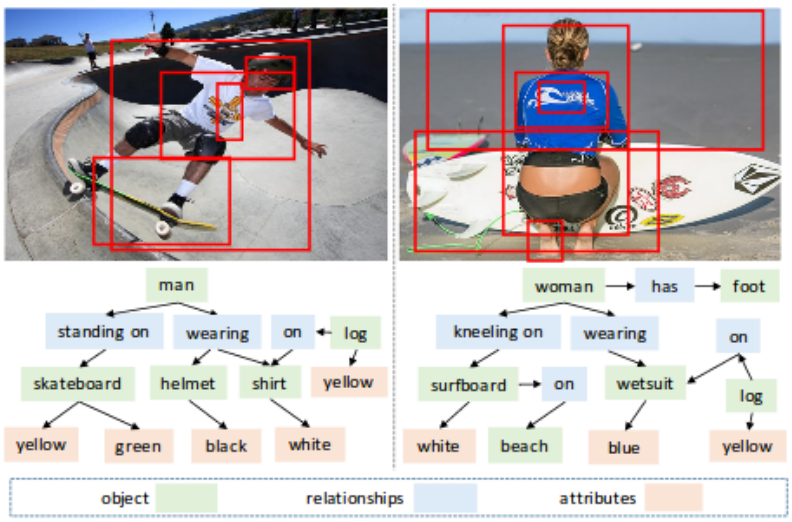


Figure 1: Example scene graphs. This figure has been taken from [8] as an illustration for clarity.

predicates given the object categories. With this relationships of very different object types but same predicates are considered in same class. But, this increases intra-class diversity. For example, 'man near horse' and 'tree near road' are very different but considered in same class. The poor results of the these approaches show that for even deep learning type complex approaches handling such high intra-class diversity remains difficult.

Above approaches ignore semantic inter-dependencies between objects, relationships and atributes. Thus they can only consider a small set of predicates. Liang *et al.* [8] propose to put semantic constraints on the classes by using a semantic action graph. But, they do not consider the structure information from the graph. There are may relationships and objects that tent to co-occur. A simple example would be man, shirt and relationship 'man-wearing-shirt'. To exploit such co-occurrences we propose to use recent development in network representation learning techniques to get the structure information. Due to such large search space for classes, it is possible even complex deep learning models may not be able to completely learn the structure information.

With success of deep learning in vision and natural language processing, there has been focus [11][12] recently on deep learning for representation learning in networks. These methods provide compact representation in form of a embedding which contains structural information of the network. A heterogeneous network can have multiple types of nodes and edges. Thus, our semantic graph can be considered as heterogeneous network. This work [13] focuses on network representation learning in heterogeneous networks.

2 Related work

2.1 Object detection

Object detection is a problem of detection and classification of many objects present in the image. In contrast to classical machine learning problems, here the output can be of variable length. As the object may be grounded anywhere in the image the search space becomes continuous. This is handled by using various heuristics. The number of possible objects can be reduced by using a sliding window protocol. This assumes that a object will not be smaller than a particular size.

The first paper to use Deep learning and get amazing results on object detection was **R-CNN** [14]. It extracts possible objects using a region proposal method called selective search. It uses similarity between pixels to segment them into a region proposal. Features of the region are extracted by passing to through Convolutional neural network and a classification technique is used on the top to perform classification. It achieved great results but it was very slow and power consuming.

Fast-RCNN[15] was a improvement to make it faster and better. Instead of passing each proposal through CNN individually it suggested to pass the whole image once and then use Region of interest pooling to get region features. This was faster and also with fully connected layers it became end-to-end differentiable. **Faster-RCNN** [16] added a region proposal network in order to get rid of the selective search algorithm. This made the network trainable end-to-end. Region proposal network uses variable size anchors. They are slided through the image giving many possible bounding boxes. Additionally it uses foreground and background prediction to make object proposals better.

In most of the relationship detection literature, Faster-RCNN is used to get objects from the image.

2.2 Scene understanding and Relationship prediction

One of the more popular ways of representing a image is text description. But, scene graphs can contain more information in a more organised manner. Also, they have object grounded in the image so there is no referential uncertainty. Thus, scene graphs are better as compared to text descriptions. Scene graphs have been used for various visual tasks such as image retrieval [4]. It can also potentially be used for visual question answering.

2.2.1 Visual Relationship Detection with Language Priors

Previously [10], there has been attempt to solve relationship detection problem by considering visual phrases or each relationship as a different class. But these methods do not scale well as we saw in Section 1. Lu *et al.* [2] first ran visual relationship detection on large scale. They considered over dataset containing over 37k relationships on 5000 images. It had 100 object categories and 70 relationship categories. It learns 2 convolutional neural networks. First one is to classify objects. Second, is trained on union of bounding boxes to classify predicates. It then combines the output of the prediction as follows:

$$V(R_{i,k,j}|O_1, O_2) = P_i(O_1)(z_k^T CNN(O1, O2) + s_k)P_j(O_2)$$

where, z_k and s_k are used to get relationship likelihoods. k lies between 1 and 70. $P_i(O_1)$ are the likelihoods obtained from the first CNN. CNN(O1,O2) is the output of the second CNN.

Additionally, it learnt priors from pretrained word vectors. They learn a relationship projection function to get relationship embedding from word vectors. Sum of similarities of objects and predicates in word vector space are used to get similarity between two relationships. The function is trained to minimise loss. The combined output is likelihood times the prior.

2.2.2 Detecting Visual Relationships with Deep Relational Networks

In [7], first, object detection is performed using Faster-RCNN. Pairs are filtered using their position in the image. For the filtered pair of objects DR-Net module takes in subject features, object features and predicate features. Features are concatenation of appearance features obtained from convolutional neural networks and spatial features of locations in the image. The paper proposes to solve conditional model it as a conditioanl distribution given the object categories. The model could also improve the object and subject type.

Conditional Random Fields have been used to model contitional distribution. x_s and x_o are the features for the subject and the object; x_r is the feature representation of the relationship

$$\begin{aligned} p(r,s,o|x_r, x_s, x_o) &= \frac{1}{2} exp(\phi(r, s, o|x_r, x_s, x_o; W) \\ \phi =& \Psi_a(s|x_s; W_a) + \Psi_a(o|x_o; W_a) + \Psi_r(r|x_r; W_r) \\ &+ \Psi_{rs}(r, s|W_{rs}) + \Psi_{ro}(r, o|W_{ro}) + \Psi_{so}(s, o|W_{so}) \end{aligned}$$

Given s and o, conditional distribution for r will be:

$$p(r|s,o,x_r,x_s,x_o) \propto exp(\Psi_r(r|x_r;W_r) + \Psi_{rs}(r,s|W_{rs}) + \Psi_{ro}(r,o|W_{ro}))$$

In typical formulations potential function is often considered a linear function.

$$q_r = \sigma(W_r x_r + W_{rs} 1_s + W_{ro} 1_o)$$

Similar equations can be obtained for object and subject type by conditioning on the other two. Then these can be considered as 3 parallel LSTMs.

2.2.3 Scene Graph Generation by Iterative Message Passing

The approach here [3] is very similar to the previous paper. The interesting part here is instead of predicting each relationship in isolation they use unique message passing architecture (similar to Graph neural network) to perform the prediction. They exploit the unique property of scene graph being bipartite. The construct a initial graph using the visual features. The also construct the dual of the graph where edges in scene graph become nodes and nodes become edges.

The message passing and aggregation can be described as follows:

$$m_i = \sum_{j:i\to j} \sigma(v_1^T[h_i, h_{i\to j}])h_{i\to j} + \sum_{j:j\to i} \sigma(v_2^T[h_i, h_{j\to i}])h_{j\to i}$$
$$m_{i\to j} = \sigma(w_1^T[h_i, h_{i\to j}])h_i + \sigma(w_2^T[h_j, h_{i\to j}])h_j$$

m_i is the aggregated input to the $i_t h$ node of the primal graph. $m_{i\to j}$ is the input to the node in dual graph. h_i is the hidden state of the nodes in primal graph at the previous time stamp. While, $h_{i\to j}$ is the hidden state of the nodes in dual graph at the previous time stamp. Using these the hidden state is again computed and passed to adjacent nodes in the primal or dual graph.

2.2.4 Deep Variation-structured Reinforcement Learning for Visual Relationship and Attribute Detection

The paper first builds a semantic graph to encode the semantic correlations between object types, predicates and attribute types. This encodes the possible relationships that could occur. It thus reduces the search space from order of tens of thousands to tens.

For this they use a unique variation structured reinforcement learning. Previous methods had to perform relationship predictions over all the object pairs, thus chances of them

getting falsely detected are high. This method puts a constraint on the relationship that can occur using the semantic action graph.

Scene graph is generated by performing BFS traversal on a subgraph of the semantic graph. The traversal outputs a scene graph. For the BFS traversal, agent at each time step must select the next edges from the possible edges of the object. The environment is set up such a way that at each time it passes the state information to the agent. The state information includes visual features of two objects, whole image features and features of previous 2 states. It also passes a set of possible attributes, possible predicates of the object, and set of possible objects to the agent. Objects are visited only once. When a object is visited, the agent predicts 3 actions. First one if for the attribute of the object. Second one is predicate between the pair of objects. Last one selects a object (or a edge) to visit next. For a current object all the relationships are found and then only it visits next objects.

This method has lot of shortcomings like it can not predict any back edges during the traversal. The feature vector passed in over 21k dimensional.

2.2.5 Pixels to Graphs by Associative Embedding

The paper [9] tries to solve object detection and relationship detection problem is a single deep network. They use stacked hourglass network to get 2 heatmaps. Hourglass network is an auto-encoder type of network built only using convolution and pooling layers. It is a deep residual network. Each hourglass network has about 24 layers.

It first tries to predict 2 heat maps. In the first heatmap a 1 denotes there is an object centered at that pixel. In the second heatmap a 1 denotes there is a relation centered at that pixel. Using the final layer features at the pixels where heatmap is 1, the object class, relationship class, object bounding box height and width are predicted.

Matching objects with relationship predicates is complex here. They have 3 additional fully connected layers. The compress the final layer features in very small dimension (8 is used in paper). They define 2 additional loss functions:

$$L_{pull} = \frac{1}{\sum_{i=1}^{n} K_i} \sum_{i=1}^{n} \sum_{k=1}^{K_i} (h_i - h'_{ik})^2$$

$$L_{push} = \sum_{i=1}^{n1} \sum_{j=i+1}^{n} \max(0, m - ||h_i - h_j||)$$

The relationship embedding is converted into 2 small dimensional embeddings. First one must be similar to the subject and second one must be similar to object. Pull loss functions is defined in such a way that compressed embedding of object and one of the embeddings of the relationship are closer. Also, if there is no relationship between two objects their embeddings are pushed apart. In the equations h_i is the compressed object embedding and h_{ij} are the compressed relationship embeddings.

2.3 Network representation learning

2.3.1 DeepWalk: Online Learning of Social Representations

The paper [11] proposes to use the work on language models for vertex representation modelling in graphs. A document is considered analogous to graph, words to nodes in graph, edges to co-occurrence and sentences to a Random walks in a graph. To justify this the paper also shows that the power-law distribution of vertices appearing in short random walks follows a power-law, much like the distribution of words in natural language.

It then shows how Skip-gram model can be used to get vertex representations. For each vertex as a starting node some number of random walks of fixed length are performed. For each random walk, for each node the representations are updated such that the probability co-occurrence of nodes occurring in the context is high. The paper uses Hierarchical softmax to approximate probability distribution.

The paper shows mutually beneficial connection between graphs and language modelling. Language modelling is actually sampling from an unobservable language graph. The authors believe that insights obtained from modelling observable graphs may in turn yield improvements to modelling unobservable ones.

2.3.2 node2vec: Scalable Feature Learning for Networks

In the paper [12], feature learning is formulated as Maximum likelihood optimisation problem, where the objective function maximises the log-probability of observing a network neighbourhood N(u) for a node u conditioned on its feature representation, given by f. It is assumed that likelihood of observing a neighbourhood node is independent of observing any other node.

The paper experiments with different search techniques to sample random walks on the network. The neighbourhoods sampled by BFS lead to embeddings that correspond

closely to structural equivalence as they obtain microscopic view of the neighbourhood. While, the nodes sampled through DFS more accurately reflect a macro-view of the neighbourhood which is essential in inferring communities based on homophily. But characterising the exact nature of node-to-node dependencies is hard given constraints on the sample size and a large neighbourhood to explore, resulting in high variance.

The paper introduces 2 parameters to control BFS/DFS traversals. Return parameter controls the likelihood of immediately revisiting a node in the walk. It thus controls the distance from the starting node. In-out parameter allows search to differentiate between inward and outward node. If it is greater than 1, BFS like behaviour is approximated as samples comprise of nodes in small locality. If it is less than 1, it reflects DFS which encourages outward exploration.

2.3.3 metapath2vec: Scalable Representation Learning for Heterogeneous Networks

As many networks in the open world are also heterogeneous, this paper [17] proposed to do network representation learning for such networks. The meta-path is used to guide the random walk, so as to construct the set of neighbour nodes of the node. Thus, we get the embeddings of the nodes. It tries to retain the structural and semantic relationships present in the heterogeneous network.

One must provide a set of meta-paths. A meta-path is a path that connects multiple node types through a set of relations and can be used to encode the semantic relationship information of the network. Heterogeneous skip-gram model is based on the original model with superposition of different node types. It tries to maximise:

$$argmax_\theta \sum_{v \in V} \sum_{t \in T_V} \sum_{c_t \in N_t(v)} \log(p(c_t|v;\theta))$$

where, V is set of nodes. T_V is set of node types. $N_t(v)$ denotes v's neighbourhood with the t^th type of nodes and $p(c_t|v;\theta)$ is commonly defined as a softmax function that is: $p(c_t|v;\theta) = \frac{e^{X_{ct}\dot{X}_v}}{\sum_{u \in V} e^{X_u \Delta X_v}}$. Here, X_v is embedding vector of node v. Over this as done in node2vec, negative sampling is used.

The meta-path-based random walk strategy ensures that the semantic relationships between different types of nodes are properly incorporated into skip-gram. The only difference here is that metapath guides the walk. Jump probabilities are restricted as per the input of metapaths provided.

2.4 Dataset description

Visual Genome [2] contains over 100k images. For each image, it contains a annotated scene graph. It also contains question answers and image descriptions for each image. On an average each image contains 25 objects and 22 relationships. It has over 1.7 million questions on the images. It contains over 42,374 unique relationships. Above papers use different set of Top-k object categories and predicate categories for evaluation.

3 Problem statement

To get scene graph from a image, we can assume that region proposals are given. We can also use a object detection system to get initial set of objects. For each object we need to infer the class label and bounding box center coordinates, height and width. Alternatively, we can also use a end-to-end network that also does object detection for us. The task now remains to detect relationships and assign classes to capture the interactions between the objects. We must also detect for each object in the proposal, a set of attributes. For this we have knowledge from a semantic graph about the semantic constraints on the object and predicate given the subject class.

$$\mathbf{x_{objs}} = \{x_i^{cls}, x_i^{bbox} | i = 1 \dots n$$
$$\mathbf{x_{rels}} = \{x_{i,j} | i = 1 \dots n, j = 1 \dots n, i \neq j\}$$

,where n is the number of proposal boxes, $x_i^{cls} \in C$ is the class label of the i-th proposal box, $x_i^{bbox} \in \mathrm{R}^4$ is the bounding box offsets relative to the i-th proposal box coordinates, and $x_{i,j} \in R$ is the relationship predicate between the i-th and the j-th proposal boxes. C is the set of possible object classes and R is the set of possible predicate classes. $\mathbf{x_{objs}}$ is a set of objects in the scene graph. $\mathbf{x_{rels}}$ is a set of relationships in the scene graph defined on the objects.

Let B_I be the set of object proposals obtained from region proposal network. We are also given a semantic graph S. So, we would like to get the following probability distribution:

$$p(\mathbf{x_{objs}}, \mathbf{x_{rels}} | I, S)$$

This can be broken into:

$$p(\mathbf{x_{objs}}, \mathbf{x_{rels}} | I, S) = p(B_I | I) p(\mathbf{x_{objs}} | I, B_I) p(\mathbf{x_{rels}} | I, \mathbf{x_{objs}}, S)$$

4 Plan of Action

From the literature survey it is clear that information from multiple sources must be combined to make the final prediction. More specifically we would like to combine the following information to make the prediction:

- Visual information of the object and relationship
- Local context from the image
- Global semantic knowledge

For the visual information as used in most of the previous research we can use Faster-RCNN. [7] and [3] use parallel LSTM to pass information for object detection and relationship detection. We would like to separate object detection and relationship detection. This makes it easier as we'll still have local context and global context to combine during the relationship detection.

Object prediction: We would like to get a visual embedding that has context of the image as well as the visual information about the object. There are multiple ways to do this. The foremost step here would be to get the features from a convolutional neural network. One way to inculde the global context is to add the embedding of the whole image (either by concatenation or plain sum). A better way which we think would be to take the initial set of features obtained from convolutional neural networks and pass them through a Bi-directional LSTM. With this we should have the combined visual and local context set of features.

Semantic graph embedding: Semantic graph embedding can be obtained using any random walk based method on heterogeneous networks. An important part would be to define notion of similarity between object types and relationship types. A very naive approach would be to define this is the following manner. Similarity of 2 objects can be defined as the number of common relationship predicates they can have, both as a subject and object. Similarly similarity between predicate classes can be defined. A naive approach from here on would be to use Singular Value Decomposition (SVD) to get compact representation of the similarities. Or we could use random walk or other Graph Convolution based methods.

Relationship detection and prediction: We should now have a embedding for each object containing visual information and local context of the image. To predict the image both visual information and global context must be combined. From the semantic graph and spatial features we can prune out pairs for which we are most certain that they won't

have a relationship between them. A naive approach for this would be to use concatenated visual embedding and semantic graph node embedding of the object pairs to make the relationship prediction. But, here we lose out on the information of the current state of the scene graph already generated.

A more better method would be where we can keep the context information of the already predicted relationships. As menthioned earlier, many relationships tend to co-occur. Our human annotated ground truth may only have some of them. For example, 'bottle-ontop-table' and 'table-below-bottle' can coexist. For this we can use LSTM to store the context information of already detected relationships.

Further improvement would be one that has the chance to also predict possible occurrence of other relationships that could have occurred in the image based on the context of already found relationships. This approach would be similar to [8] where Reinforcement learning bot predicts the next relationship. But, this was restricted in many ways, one being it can only predict n relationships and the other that it cann't predict backedges during the BFS-traversal. We would like to design a search based algorithm that is not restricted by such traversal based constraints. One possible solution is to allow the agent to predict relationship from those occurring at less than2-hop distance.

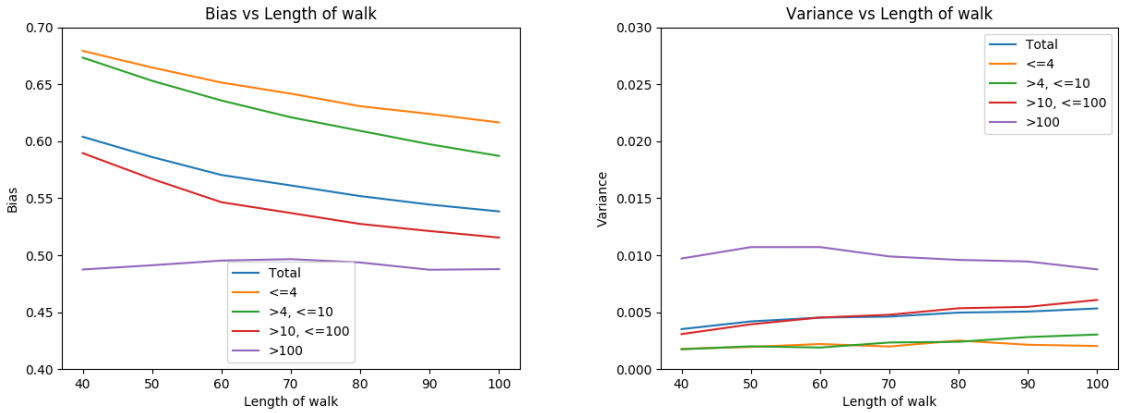
5 Experiments performed

5.1 Bias-variance decomposition analysis of node2vec

The aim here was to analyse recent work on random walk based methods for deep network representation learning.
To look at the bias-variance decomposition of the error and how they vary with the change in the parameters, we deviate from measure as accuracy to mean squared error. Experiments were performed on BlogCatelog3 network, with node Label classification task. The task was multi-label, multi-class classification, i.e. each node may belong to multiple classes.
Bias calculation Bias was calculated by taking mean squared error of the mean of the output probability distribution from Logistic Regression on the embeddings of these 10 models to that of the expected output probability distribution. This task was multi-class multi-label so for mean squared error the probability mass was equally distributed among the positive classes. Bias thus calculated is an approximate of the actual bias as we can not deduct the underling function.
Variance calculation Variance was calculated by taking Mean squared error of the mean

output probability distribution with each of the 10 output probability distributions. As the variance of due to negative classes was high, both bias and variance were calculated on the positive classes only.

Task For node label classification, embeddings of 90% of the nodes was taken as training data and rest 10% was taken as test data. Logistic regression was used for classification. It was found that the model had high bias and low variance. It was also observed that nodes with smaller degree have more bias and less variance than nodes with higher degree. One possible reason for this could be due to their less number of occurrences in random walks. Many methods were tried to decrease this bias in the system.



(a) Bias vs length of random walk

(b) Variance vs length of random walk

Figure 2: BlogCatelog - Bias and variance trends for different category of nodes

- Longer random walks
- Number of walks form each node proportional to its degree
- Sample 2 Random walks for a Random walk and concatenate them.
- Smaller values of k.

No major improvements were seen using these methods.

5.2 Reproduction of Pixel to Graphs by Associative Embedding results

Initially, the plan was to incorporate the semantic constraints in this network itself. So many experiments were run to reproduce the results of the paper. There were many problems initially to run the code due to the system requirements. Some parallel operations were serialised to fit the model in the memory. But, out of the 6 loss functions

2 did not converge. Obtained results are as follows. The decrease in heatmap loss and the pull loss was not significant after 5 epochs.

Table 1: Losses

	Heatmap	Obj class.	Obj reg.	Rel class.	Pull	Push
Initial	2482.52	11394	4.3e+07	6823.8	12596	21382
5 epochs	1907.47	791.51	2247.9	130.5	7958.1	1.8

6 References

[1] Andrej Karpathy and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3128–3137.

[2] Cewu Lu et al. "Visual relationship detection with language priors". In: *European Conference on Computer Vision*. Springer. 2016, pp. 852–869.

[3] Danfei Xu et al. "Scene Graph Generation by Iterative Message Passing". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[4] Justin Johnson et al. "Image retrieval using scene graphs". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3668–3678.

[5] Angel X Chang, Manolis Savva, and Christopher D Manning. "Learning Spatial Knowledge for Text to 3D Scene Generation." In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 2028–2038.

[6] Damien Teney, Lingqiao Liu, and Anton van den Hengel. "Graph-Structured Representations for Visual Question Answering". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[7] Bo Dai, Yuqi Zhang, and Dahua Lin. "Detecting Visual Relationships With Deep Relational Networks". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[8] Xiaodan Liang, Lisa Lee, and Eric P. Xing. "Deep Variation-Structured Reinforcement Learning for Visual Relationship and Attribute Detection". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[9] Alejandro Newell and Jia Deng. "Pixels to Graphs by Associative Embedding". In: *Advances in Neural Information Processing Systems*. 2017, pp. 2168–2177.

[10] Mohammad Amin Sadeghi and Ali Farhadi. "Recognition using visual phrases". In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE. 2011, pp. 1745–1752.

[11] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. “Deepwalk: Online learning of social representations”. In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 701–710.

[12] Aditya Grover and Jure Leskovec. “node2vec: Scalable feature learning for networks”. In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2016, pp. 855–864.

[13] Shiyu Chang et al. “Heterogeneous network embedding via deep architectures”. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2015, pp. 119–128.

[14] Ross Girshick et al. “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 580–587.

[15] Ross Girshick. “Fast r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[16] Shaoqing Ren et al. “Faster R-CNN: Towards real-time object detection with region proposal networks”. In: *Advances in neural information processing systems*. 2015, pp. 91–99.

[17] Yuxiao Dong, Nitesh V. Chawla, and Ananthram Swami. “Metapath2Vec: Scalable Representation Learning for Heterogeneous Networks”. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2017, pp. 135–144.