Relationship Prediction for Scene Graph Generation

Uzair Navid Iftikhar
CS 231 N
unavid@stanford.edu

Aamnah Khalid
CS 229
aamnah@stanford.edu

Abstract

In 2015, Johnson et al. [1] proposed a new framework for thinking about image retrieval problems: scene graphs. In a nutshell, these would encode information regarding the contents of an image, taking into account spatial and semantic relationships. While researchers have had some success generating images based on scene graphs [2], going in the opposite direction and generating scene graphs from images appears to be a more complex problem, and is, in part, the subject of Stanford University's challenge called GQA: Visual Reasoning in the Real World [5]. By comparing state-of-the-art approaches such as Neural Motifs [3], DenseNets [28] and the use of textual descriptions [4], we explore this problem further.

1. Introduction

Proposed by Johnson et al. in 2015 [1], scene graphs offer powerful representations of images, aimed at preserving information relating to semantic and spatial relationships, as well as salient attributes of objects, such as their color, texture, or material. For example, as opposed to simply identifying that a picture contains a man and a mirror, a scene graph might help identify that the man is standing in front of the mirror, that the mirror is hanging on a wall, and that the wall is painted blue. Being able to generate scene graphs for images would be immediately beneficial to large-scale image retrieval systems, by allowing a deeper and perhaps more natural understanding of images. The semantic knowledge captured by a scene graph is both more abstract and detailed than state of the art object retrieval and image labelling- it simultaneously provides global context of the scene and outlined local relationships between entities in the image. While exciting, the complex nature of the scene graph generation task also makes it hard to accurately and reliably fulfill on real data.

Our investigation is primarily motivated by Stanford University's GQA dataset [5], which has been developed precisely for the scene understanding challenge. This synthetic dataset consists of 113,000 images with associated scene graphs, as well as question-answer data regarding the images, where answers are mapped to specific regions of relevance in the image. Our work concentrates on the images and their respective scene graphs.[1]

[1] Equal contribution in all parts of project.

Given that powerful models such as R-CNN have made bounding-box generation a relatively easy task, our investigation has been focused on predicted relationships between objects [18]. We specifically focused our efforts on detecting relationships between objects. Experiments included maximum likelihood estimation, natural language processing and convolutional networks. We found that an architecture incorporating DenseNet[28] performed best. The model, based on Dornadula et al.'s work [27], performed somewhat well, achieving accuracies of around 30%.

2. Related Work

While the importance of context in scene semantics has been appreciated in literature for a while [20], scene graph parsing has seen increased interest recently. However, the task of reasoning about the complex dependencies of components in an image continues to present a unique challenge. Therefore, a large part of our investigation was focused on studying existing methods and recent advances in scene understanding of visual data.

The body of literature on scene graphs belongs to either of two main categories: scene graph generation and relationship proposals. The majority of scene graph generation has models have been trained on the *Visual Genome* dataset [9] which gives relationships, objects and question-answers associated with 108,077 images. However, state-of-the-art models focus entirely on the visual data and its graphical representation [3][10][7][8][15][21][22][25]. Visual relationship predication is studied beyond the scope of scene graph generation, however recently, scene graphs have become a useful structure to encode these relations. [11][24][25]

Visual relationship prediction poses a task that is at least quadratic in the number of objects present in the graph (depending on the model, the task can be quadratic over the object classes [21]). During the review we found that most existing models for scene parsing use Faster RCNN with a VGG backbone as the underlying object detector. [3] [8] [6] [7] The Faster R-CNN model [19] consists of two modules. The first is a deep convolutional network that processes the image to propose regions. The second module applies a Region of Interest (*RoI)* pooling layer to each proposed region to extract a fixed length feature vector that is fed into a fully-connected layer to extract object classes.
[5]

We studied a variety of ways to establish some semantic context over the graph and propagate it in the model. In Xu et

al. [7] the model decomposed the problem into two sub-graphs– one for objects and one for relationships – to perform message passing.

Zellers et al. in their Neural Motifs paper [3] emphasize the importance of common, often high-level, structural patterns in relationships between components of an image. They find that the distribution of the relations is highly skewed once the relevant object categories have been identified. Their proposed neural network model, Stacked Motif Network (MOTIFNET) [8], first predicts bounding regions on an image, then establishes labels for regions and then proposes relationships. Bidirectional LSTMs compute global context between each of these stages and propagate them to the next step. Particularly interesting is the proposed strategy to construct a contextualized representation of bounding regions, B, and objects, O, using the bidirectional LSTM layer:

$$D = biLSTM([\, c_i; \boldsymbol{W}_2\, \hat{\mathbf{o}}_i]_{\{i=1,\ldots,n\}})$$

where the "edge context" D = [d_1 , … d_n] contains the states for each bounding region in B at the final layer of the model and W2 is a parameter matrix mapping o^i into R^100. The model then predicts which relation the edge can be classified by computing probabilities using context D and overlap of bounding boxes. We choose to highlight this particular feature of MOTIFNET among its many novel techniques because one aim of our project is to understand how to effectively represent semantic context. On Visual Genome [9], this technique fares far better than the iterative message passing approach proposed by Xu et al. which we were previously considering as a possible approach to take [10].

Scene graph generation is not a task limited to visual data. The work on semantically precise scene graphs from textual descriptions from Schuster et al. [4] is quite informative on how to deal with the question-answer data. The experiments use rule-based and classifier-based parsers to create scene graphs. The parsers perform drastically better than k-nearest neighbor models however, they do not seem to improve more than 2.5% over an object classifier that does not predict relationships between components of an image. The authors deduce that the single sentence restriction over their parser inhibits the model from achieving sizeable performance gains. Nonetheless, the model appears to be a relevant comparison point for many scene graph parsers we encountered during our literature review and therefore a fair candidate for investigating the utility of QA data in logical reasoning about visual data.

3. Methods

3.1. Loss for Neural Networks

Models for scene graph generation are not a real valued regression problem, but predict predicates from within a finite multi-class set of possible labels. As a result, the problem, especially its subsection on relationship prediction, is well suited to cross entropy loss:

$$-\frac{1}{N}\left(\sum_{i=1}^{N} \mathbf{y_i} \cdot \log(\hat{\mathbf{y}}_\mathbf{i})\right)$$

Where N is the number of data points and both the ground truth vector **y** as well as predicted probabilities for each class, $\hat{\mathbf{y}}$ are *k*-dimensional vectors given that there are *k* possible classes.

In our approach we compute our loss over predictions from visual modules for the relationships. There is an alternate approach proposed by *Lu and Krishna*[11] where the object predicates are mapped to a word-embedding vector space and the distance of the object-pair's word vectors as well as the variance over the vector embedding all relation-object triples in the dataset contribute to the loss calculation of the model. For our dataset, the approach presented two major problems. Firstly, the relationship predicates in GQA are not limited to single word tokens. Therefore, state-of-the-art embeddings such as GloVe [26] could not be used. Furthermore, word-vector based loss functions such as the ones presented in the paper rely on calculations over the entire dataset which is a computationally expensive operation. In fact, the original paper itself limits its training to only 4000 images in the Visual Genome dataset and its results lack reproducibility in the larger body of work on scene graphs. We therefore focused our efforts on deriving visual relations based on the visual module and training language models separately.

3.2. Evaluation Metric

Due to Bayes' rule, all papers discussed above aim to accurately model the conditional distribution of a relationship predicate given an object pair (where the order of the objects is relevant):

P(Object1, Relation, Object2) =
P(Object1) **P**(Relation|Object1, Object2) **P**(Object2)

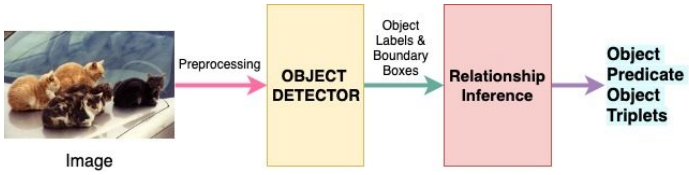
Present models are evaluated on the *Recall@*k metric [11]. Recent work by Zhang, Ji, et al. [36] has raised concerns about current evaluation metrics as it penalizes models for incorrect scene graphs even when most of the relations are classified correctly. Most visual datasets also suffer from incomplete labelling which in a detailed annotation task such as scene graphs can misdirect training [5]. We therefore propose the following metric to evaluate our models for a single image:

$$\sum_{\mathcal{R} \in \text{Image}} \left(\frac{1_{[predrictedRelationship == \mathcal{R}]}}{\text{total relations in image}} \right)$$

For the metric to be valid, our preprocessed subset of the dataset (which is discussed further in the following section) did not contain any triplets where the two objects at hand were unrelated.

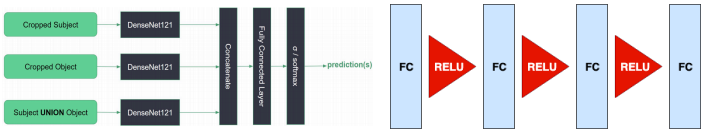
3.3. Visual Modules

Based on current state-of-the-art models, we decided on the following pipeline for scene graph generation:



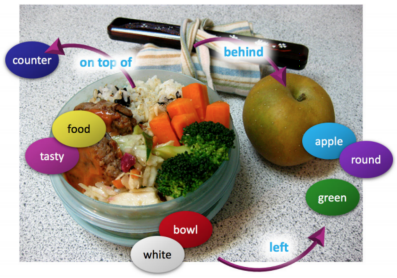
Using a Deep Learning model for object detection introduced some unwanted stochasticity into our experiments. Since the bounding boxes in the scene graphs associated with the images had been generated by an R-CNN network, any fine-tuning on the dataset would likely regenerate almost the exact same bounding boxes and object labels at great computational cost.

For visual relation models, the input to our algorithms would be a color image, which would then be processed through a deep neural network (CNN or fully-connected), and we would output all relationships between object pairs found in the image. For our experiments, we sampled and preprocessed a subset of 12,000 images and worked to predict relationship predicates for object-relation-object triplets encoded in the scene graphs. We tried two main neural networks for prediction: DenseNet and a Fully-Connected model.



Left: DenseNet architecture **Right:** Fully connected network architecture

4. Data



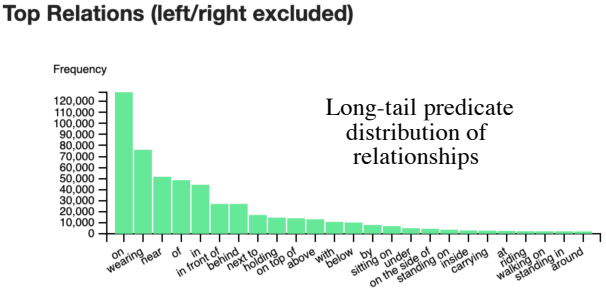
An example of a scene graph, from the paper introducing the GQA dataset. [5]

As mentioned before, we are using Stanford's GQA dataset [5], developed precisely for the scene understanding challenge. This contains 113,000 images with associated scene graphs, as well as 22 million questions of varying compositionality degrees, measuring performance on an array of reasoning skills such as object and attribute recognition, transitive relation tracking, spatial reasoning, logical inference and comparisons. Each image is accompanied by a scene graph describing the objects, attributes, and relations it contains. Questions are associated with a functional program which lists the series of reasoning steps that have to be performed to arrive at the answer. Answers data is annotated with bounding boxes of relevant regions in the image. As the full dataset is well over 50GB, we used a subset consisting of 12,000 images.

Unlike the *Visual Genome* dataset, GQA does not provide data-loading API and neither are any publicly available. As a result, the data required significant pre-processing. The images were of varying sizes and dimensions, therefore we added square padding to the right and left in order to preserve bounding box regions, and were all resized to 256 x 256. This also meant that the bounding boxes for the images had to be rescaled and unlike *Visual Genome* there was no single scaling factor applicable across the dataset because the original regions were bounded on the scale of the individual picture. We also rearranged the `sceneGraph.json` file to give a more intuitive `object1-relation-object2` structure of the relationship data which previously mapped from non-unique object IDs (often many in the same picture for the same object class) to the object ID of another region, without specifying the object class itself. In the original scene graph structure, this which resulted in $O(k)$ look-up time for k possible classes- costly given that each image has multiple objects, in order to expose relationships.

As a synthetically labelled dataset, bounding regions and their labels in GQA have been generated by the Fast-RCNN model we discussed earlier. Therefore it seemed plausible to use an existing pre-trained object detection model to get the object predicates, bounding boxes and the likelihood of the predicates. Many existing detectors such as YOLOv3, Mask R-CNN and Fast-RCNN [13][17][18][19] have been trained on the same images as *Visual Genome*, the dataset that GQA is itself based on. Therefore, following the approach of many of the scene graph generation models mentioned above, we used a pre-trained object-detector-YOLOv3.

The relationship data encoded in the scene graphs follows the characteristic long-tail distribution; this also presents a potential problem for learning as models may learn to simply output the most common class.



5. Experiments

To reiterate, the precise problem we wish to solve is that of inferring relationships between objects in a given picture.[2] Each of our learning algorithms required a training set, validation set (for hyperparameter tuning), and test set. To accommodate this, we used random sampling to execute a 75 - 12.5 - 12.5 split. While deep learning techniques have been known to use splits closer to 90-10 or even 99-1, these would likely not have worked well with our small dataset sample, as the dev / test sets would not have been able to capture the range of variance present in the data as well. The process that we followed for hyperparameter tuning was somewhere between a random search, and an organized search of the hyperparameter space. Specifically, we first guessed 4 – 5 values for each hyperparameter and considered its effect on the performance of the system when all other hyperparameters were kept stable. Then, once we had potential independent values for each

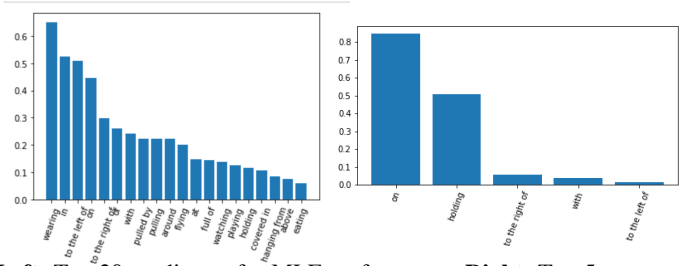
[2] As our project evolved, we abandoned object detection baseline from Milestone.

hyperparameter, we conducted a search of 5 values surrounding the value we found. All such tuning was conducted on the validation set. We used PyTorch [30], Tensorflow [31], Numpy [32], Matplotlib [33], Jupyter Notebook [34] and Scikit Learn [38] throughout our experiments.

5.1. Baselines

a) *Maximum Likelihood Estimation*:

As a baseline for relationship predictions, we implemented a maximum likelihood estimator that estimated the conditional distribution of relations given an ordered pair of objects. To account for relationships that do not occur, we applied Laplace smoothing with lambda = 1. Furthermore, to increase computation efficiency, we pruned out objects and relationship predicates that only occurred once in our training dataset. Even with this pruning, we had 262 relationship predicates and 1317 distinct object labels. Since the underlying distribution of predicates is not multivariate gaussian, we cannot apply expectation maximization to efficiently get a model of the real distribution. The MLE algorithm itself is computationally expensive and often results in python terminating the process. To train over 7000 images in vectorized code, we needed 40 GB RAM on a SAIL cluster. For a dataset of over 100,000 images, MLE is not feasible. The evaluation metric was percentage of correctly guessed relations for each image and this metric was averaged over the entire dataset. We got an average accuracy **0.372** over the validation set. This method assumes that relation-object triples appear independently of



Left: Top 20 predicates for MLE performance. **Right:** Top 5 predicates for linear SVM performance distribution of relationships

each other which is a false assumption. A counter example is that of "cat"-"on top of"-"car". Knowing this relation exists tells us that with 100% probability we have "car"-"underneath"-"cat" however if such an example was not labelled in the dataset, our MLE model will likely not predict with. Therefore, our estimator does not accurately represent the Bayesian network of the real scene graph. We also evaluated accuracy of prediction for each relationship label. Specifically, we counted the number of times the label was correctly predicted and divided that by the number of times it showed up in the ground truth data for our validation set. We have shared results for the top 20 relationship predicates in the chart below:

b) Support Vector Machines:

Viewing the relationship prediction problem as a high-dimensional classification task, we also experimented with implementing a support vector classifier where inputs are the ordered pairs of *(source object, target object)* and the outputs are relationship labels (all data given to model as index based IDs). We used MinMax scaling on the data to prevent any one feature from dominating. Our algorithm used a linear kernel with parameter *1/#relationship predicates*. The classifier performed quietly poorly with an average accuracy of **0.0591.** Since the data may not have been linearly separable, we also tried to train a sigmoid kernel. Both our own implementation and the one from Scikit [38] were not able to fit despite each of them running for 4 hours on 80 GB RAM machines. Since multi-class classifiers can take varying time depending on which decision function one uses for the prediction labels, we tried both *"one-vs-rest*", which trains *#relationship predicates* classifiers, as well as "*one-v-one",* which trains $O($ (# *relationship predicates*$)^2$) classifiers. Literature review showed that such behavior is to be expected since SVMs are not scalable to large datasets with his many features.

5.2. Deep Learning on Visual Data

a) *Learning from Bounding Box Shapes*

Our first hypothesis was that the bounding boxes themselves – regardless of what they contained – encoded enough information to be able to predict some high-level relationships. We built a 4-layered neural network that simply accepted the eight coordinates demarcating the two bounding boxes, and had a softmax activation at the end to output a probability distribution over the potential relationships that the bounding boxes may be encoding. The network we built followed the structure:

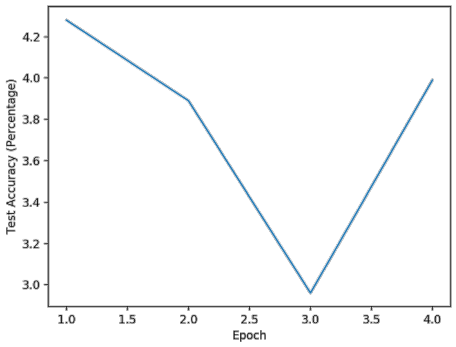


Figure: Validation set accuracy over epochs during training of **FC-ReLU 4-Layer network**

FullyConnectedLayer (8, 512) -> ReLU Activation ->
FullyConnectedLayer (512, 512) -> ReLU Activation ->
FullyConnectedLayer (512, 512) -> ReLU Activation ->
FullyConnectedLayer (512, numRelationships)

The network did not make it past 4.2% accuracy, even after the hyperparameter tuning discussed earlier, and training for 4 epochs. This is significantly below baseline performance. We also tried an alternative version of this with handpicked features, encoding whether a box was to the left of another, to the right of another, above it, below it, encompassing it, larger vs. smaller than it. This had very similar results to those shown above (Note: we lost the actual findings because our Google Cloud disk cannot be accessed after the recent crash.) These experiments helped us realize that bounding box shapes are not enough to parametrize object relationships, and that a more powerful representation would be needed, as well as a deeper, more powerful network to be able to adequately learn from such information.

b) *DenseNets*

Next, we decided to use a method pointed out by Dornadula et al [27]. DenseNets reduce the learning of redundant feature maps by giving each layer direct access to the input image and gradients from the loss function. The highlevel approach is to have feed-forward connections between each layer of the network so that in a network with L layers, there are $\frac{L(L+1)}{2}$ direct connections. Layer *n* receives a concatenation of the feature maps from layers 1,…, *n*-1. Transition layers in between these dense blocks use batch normalization, pooling and convolution which account for any changes in sizes of feature maps in between dense blocks [28].

Our relationship predictor accepts cropped images containing the source object, the target object, and the union of the bounding boxes of the two as input, processes it through a DenseNet to output a class [28]. The original experiments conducted by Dornadula et al had been on the Visual Genome dataset, which is different from what we were using it for. We incorporated the network into our experiments, writing new DataSet classes for Pytorch so that we could work with the code we had found [29]. This network seemed to perform very well,

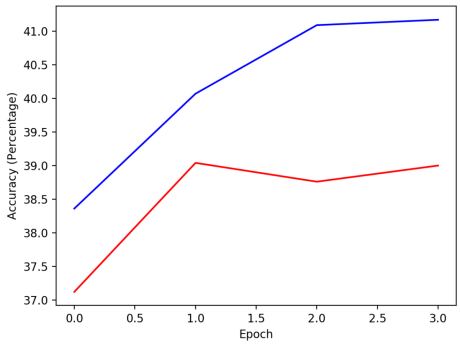
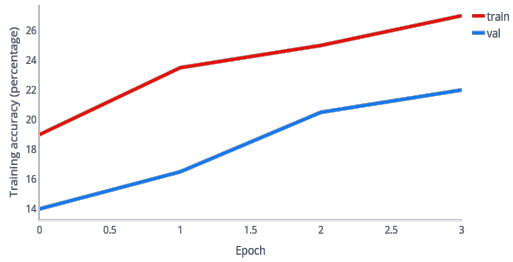


Figure: Accuracy over epochs during training of **DenseNet** model (blue: training, red: val)

converging on a training accuracy of 41%, and a validation accuracy of 38.5%. This beat MLE baseline by a small margin

5.3. Neural Motifs

To evaluate the efficacy of context propagation in MOTIFNET[8], we added a dataloader for GQA and edited the evaluation function of the model. For fair comparison, the model was trained for 4 epochs with stochastic gradient descent and achieved a test accuracy of 22.1%.



We believe that the complex network architecture of MOTIFNET required more epochs and training examples for better accuracy.

5.4. Natural Language Processing on QA Data

As preprocessing, we stripped punctuation from Q&A data and combined them into singular strings for each image. As the full dataset is well over 50GB, we used a subset of it associated with the 12000 images we used, which required manipulation of the json and HDF% files.

a) Qualitative Results

To begin predicting relations from question answer data, we used 10 samples to evaluate the RuleBased and ObjectDetector parser proposed by *Schruster et al.* [4]. An example of QA input is:

Q: Are there blankets under the brown cat?

A: No, there is a towel under the cat.

Parser from *Schruster et al.* generated the scene graph data:

Source: towel-12 **Relation:** under **Target:** cat-15
Nodes: cat-7 (attribute brown); towel-12; cat-15.

Qualitative results showed that this parser did not add to relationships predicted by visual modules, exposing the need for more complex models.

6. Conclusion and further work

Quite recently, more research has started into loss functions that are optimized for the scene graph learning task. A recent paper from Zhang et al. [1] proposes three new loss functions that account for the flaws in existing cross-entropy loss. Of particular interest are the class agnostic losses proposed that account for positive and negative direction predicted for the relationship label and allow for the model to be penalized less if the label has been predicted accurately. Another proposed function is the class aware loss that allows for less penalty when multiple instances of the same class are confused so that we account for the semantic understanding of the model. Work can be done to implement an optimized pytorch version of various models altered to include these losses since the paper does not attempt to combine its proposed losses with state-of-the-art functions.

The relevance of question answer data still remains an open question. Work needs to be done to meaningfully incorporate QA data into the scene graph generation pipeline. Our attempts to incorporate this data into the LSTM structure of neural motifs to enhance context propagation did not yield meaningful results. However, we believe that an investigation into incorporating real data feedback from textual interactions on visual data can greatly enhance semantic understanding and even correct for wrong predictions made by visual modules.

References

[1] Johnson, Justin, et al. "Image retrieval using scene graphs." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[2] Johnson, Justin, Agrim Gupta, and Li Fei-Fei. "Image generation from scene graphs." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[3] Zellers, Rowan, et al. "Neural motifs: Scene graph parsing with global context." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[4] Schuster, S., Krishna, R., Chang, A., Fei-Fei, L., Manning, C. D. (2015). Generating semantically precise scene graphs from textual descriptions for improved image retrieval. In Proceedings of the fourth workshop on vision and language (pp. 7080). Citeseer.

[5] Hudson, D. A., Manning, C. D. (2019). GQA: a new dataset for compositional question answering over real-world images. arXiv preprint arXiv:1902.09506.

[6] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei. Scene Graph Generation by Iterative Message Passing. arXiv:1701.02426 [cs], Jan. 2017. arXiv: 1701.02426.

[7] Y. Li, W. Ouyang, B. Zhou, K. Wang, and X. Wang. Scene graph generation from objects, phrases and region captions. In Proceedings of the IEEE International Conference on Computer Vision, 2017

[8] www.github.com/rowanz/neural-motifs

[9] Krishna, Ranjay, et al. "Visual genome: Connecting language and vision using crowdsourced dense image annotations." International Journal of Computer Vision 123.1 (2017): 32-73

[10] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei. Scene Graph Generation by Iterative Message Passing. arXiv:1701.02426 [cs], Jan. 2017. arXiv: 1701.02426.

[11] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei. Visual relationship detection with language priors. In ECCV, 2016.

[12] Y. Li, W. Ouyang, B. Zhou, K. Wang, and X. Wang. Scene graph generation from objects, phrases and region captions. In Proceedings of the IEEE International Conference on Computer Vision, 2017

[13] "Object Detection And Tracking In Pytorch." Towards Data Science. N. p., 2018. Web. 16 May 2019.

[14]] Zhou, Bolei, et al. "Places: A 10 million image database for scene recognition." *IEEE transactions on pattern analysis and machine intelligence* 40.6 (2018): 1452-1464.

[15]] https://github.com/yikang-li/MSDN

[16] Zhang, H., Kyaw, Z., Chang, S.F., Chua, T.S./ Visual translation embedding network for visual relation detection. In/ CVPR (2017)

[17] https://github.com/cfotache/pytorch_objectdetecttrack

[18] He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.

[19] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.

[20] Oliva, A., Torralba, A.: The role of context in object recognition. Trends in cognitive sciences 11(12), 520–527 (2007)

[21] Yang, Jianwei, et al. "Graph r-cnn for scene graph generation." Proceedings of the European Conference on Computer Vision (ECCV). 2018. Li, Yikang, et al.

[22] "Factorizable net: An efficient subgraph-based framework for scene graph generation." Proceedings of the European Conference on Computer Vision (ECCV). 2018.

[23] Li, Y., Ouyang, W., Wang, X.: Vip-cnn: A visual phrase reasoning convolutional neural network for visual relationship detection. In: CVPR (2017) Dai, B., Zhang,

[24] Y., Lin, D.: Detecting visual relationships with deep relational networks. In: CVPR (2017)

[25] Cai, Liwei, and William Yang Wang. "Kbgan: Adversarial learning for knowledge graph embeddings." arXiv preprint arXiv:1711.04071 (2017).

[26] Pennington, Jeffrey, Richard Socher, and Christopher Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing

[27] Dornadula et al. “Graphr: Scene Graph Generation using Deep Variation-structured Reinforcement Learning.” CS234 Project.

[28] Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[29] DeepVariationStructuredRL https://github.com/nexusapoorvacus/DeepVariationStructuredRL

[30] Paszke, Adam, et al. "Automatic differentiation in pytorch." (2017).

[31] Abadi, Martín, et al. "Tensorflow: A system for largescale machine learning."12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16). 2016.

[32] Numpy, GitHub. https://github.com/numpy/numpy

[33] Matplotlib, GitHub. https://github.com/matplotlib/matplotlib

[34] Jupyter Notebook, GitHub. https://github.com/jupyter/notebook

[35] PyTorch BERT: https://github.com/huggingface/pytorch-pretrained-BERT

[36] Zhang, Ji, et al. "Graphical Contrastive Losses for Scene Graph Parsing." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[37] Radial Bias Function Kernel https://en.wikipedia.org/wiki/Radial_basis_function_kernel

[38] Scikit Learn https://scikit-learn.org/stable/supervised_learning.html