Chris Rockwell
4/23/2019

Replication and Evaluation of "Image Generation from Scene Graphs [1]"

1. Introduction

Inspired by the papers "Building Machines that Learn and Think Like People [2]" and "Towards Evaluating the Robustness of Neural Networks [3]" from class, I am convinced neural networks are learning pattern recognition as opposed to model building in many applications. With this in mind, "Image Generation from Scene Graphs" seemed a worthy paper for replication, since it attempts the goal of flexible generation that should favor model building over pattern recognition. Furthermore, because the paper did not have a satisfying quantitative evaluation, there seemed to be a nice option for extension.

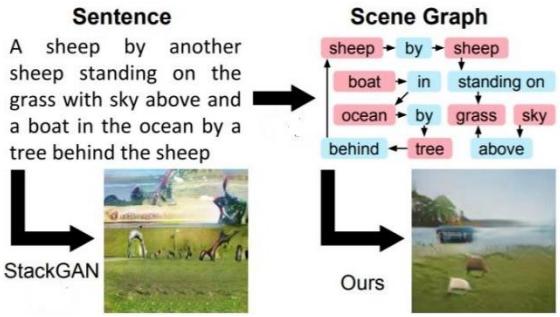
The paper was submitted by authors including soon-to-be Michigan Professor Justin Johnson and the respected Prof. Fei-Fei Li. It was published at CVPR 2018 and already has 33 citations on Google Scholar. While of course author, university, citation count, and conference title do not guarantee an important or quality work, they are evidence supporting my assertion that this is a worthwhile paper to replicate. The authors additionally provided code on Github, which was appealing given the complexity of the model and training. Nevertheless, replication required understanding of the code and parameters along with careful training, allowing some room for a project. Designing and conducting proper evaluation would also require plenty of work. This required editing selective portions of the paper code along with that of a visual question answering algorithm to be used with fresh evaluation code. This part ended up being more work than anticipated, and the method isn't as good of a method for evaluation as hoped, but I believe the result still has some utility and provides additional insight about the replicated paper.

In the rest of the report, I detail the replicated paper and touch upon the method used for question answering in evaluation. I then describe how I was able to replicate the paper and setup for the additional evaluation step. Finally, I present qualitative and quantitative results of replication and conclude.

2. Image Generation from Scene Graphs

The goal of this paper is to generate images best representing scene graphs taken as input, where a scene graph represents a sentence in terms of objects and their relationships (**Fig. 1**). A simpler but similar task of generating images from limited descriptions of objects has seen great success in the past by combining RNNs and GANs [4,5]. However, these models struggle to properly generate objects in complex sentences (**Fig. 1**). This paper leverages the additional information from scene graphs to allow explicit reasoning about relationships between objects.

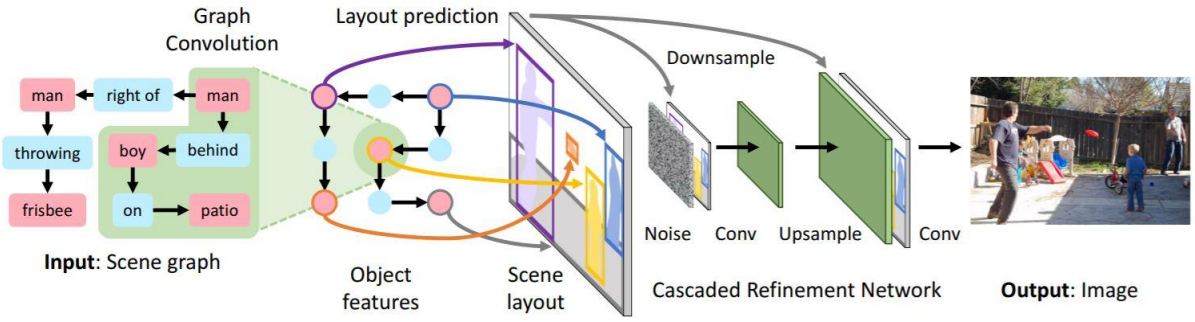
Fig. 1. Scene graph, corresponding to a sentence, is used as input to model. Typical text-to-image models such as StackGan struggle to generate proper images on more complex sentences



Their process can be broken down distinctly into five sequential pieces.

1. Embed objects and relationships from scene graph: modeling objects in scene graphs as nodes and relationships as edges, the network learns to embed nodes and edges based on function.
2. Embed aggregate scene graph: the network next uses a graph convolutional network to represent these embedded objects and relationships as aggregated object embeddings
3. Generate scene in image shape: taking aggregated object embeddings as input, the network utilizes transpose convolutions followed by a sigmoid operator to predict masks for each object. The network additionally predicts bounding boxes using a multilayer perceptron. Embedding vectors for each object are then shifted and scaled to their corresponding mask and bounding box.
4. Synthesize image: the network uses a cascaded refinement network to create an image at increasing resolutions, with each step taking as input scene layouts, prior images and gaussian noise
5. Discriminative step: finally, the network is trained adversarially against a discriminator, which tries to determine whether both the entire image and objects in the image are real or fake.

Fig. 2. Model overview



Training minimizes losses from discriminators, bounding box and mask, per-pixel difference from ground truth, and separate classifier loss of objects in image. Evaluation takes place on COCO [6] and Visual Genome [7] datasets and the primary result is of humans comparing generated images to those generated from the corresponding sentences using StackGAN. Additionally, the authors report qualitative results and analysis, generated object bounding box IOU, and inception scores [8].

None of the authors' quantitative approaches are especially satisfying. Their primary result requires human evaluation of each image. Not only does this mean testing is expensive, but it means validation is likely to not use a good proxy for testing accuracy. Bounding box IOU is clearly an incomplete score, as objects could be at various locations and still accurately represent a given scene graph. It therefore is not a good measure of how well relationships are represented. Furthermore, this approach does not capture how realistic the image looks as a whole. Inception scores, which essentially measure how well generated objects can be classified, captures object clarity but again says little about realism of the image or how well relationships are represented.

I therefore propose an alternative method of evaluation: answering questions about generated images. To evaluate, I use this metric along with a qualitative report of selected images used in the paper. By breaking down specific images with the visual question answering framework, we can gain additional insight into differences between the authors' model and ground truth images and images from the model I trained.

3. Visual Question Answering

Along with scene graphs, captions, and images, the Visual Genome dataset additionally has many questions and answers corresponding to each image. We therefore can use visual question answering algorithms to measure accuracy of generated images with more breadth. Answering questions correctly can require relationships between objects to be properly represented, yet allows some flexibility in where objects are. Our algorithm is trained to predict based on ground truth images, so in theory extracting features for prediction from less realistic images should be more difficult, encouraging realistic images.

The visual question answering algorithm used is "MUTAN: Multimodal Tucker Fusion for Visual Question Answering" [9]. The algorithm was selected because it was implemented in Pytorch, so could more easily be combined with the image generation algorithm. Additionally, it was one of the few models available online pretrained on the Visual Genome dataset. Finally, its performance was near state-of-the-art on tested datasets. Although detailed explanation of the algorithm is perhaps beyond the scope of this presentation, general understanding was required for the project and I will thus broadly outline the method used for question-answering.

1. Extract features from images using ResNet [10] and embed questions using a GRU recurrent network [11]
2. Fuse embeddings using bilinear models [12]
3. Extract features on combined embeddings using a variant of the Tucker decomposition [13]
4. Produce class scores for answers from features (answering as classification problem)

4. Replication

Replication consisted first of the fairly straightforward task of setting up, training and validating the image generation model. Second was the much more involved setting up evaluation and evaluating using the image generation model combined with visual question answering model.

The image generation setup consisted of first downloading the COCO and Visual Genome datasets and configuring their APIs, and changing flags and dependencies within the code to be

consistent. Next, I setup a proper GPU workspace and used hyperparameters as close to the original paper as possible. The paper did not describe how to split training between COCO and Visual Genome, but given evaluation will completely take place on Visual Genome, I trained solely on this dataset which gave good results. Unfortunately, training was more expensive than originally imagined and took over a week. The GPU used in the paper was significantly better than that used for replication, and the large network combined with long training time of one million iterations were to blame. Despite setup being somewhat straightforward, the long train time and problems of getting consistency through a rather massive code, this first step took a couple weeks to get a good result.

Nevertheless, the second step required the vast majority of the time. The general pipeline for evaluation is:

1. Load data
    a. Select 500 images from Visual Genome test set with answers in set used to train visual question answering model
    b. Get question vector and answer integer from visual question answering model
2. From scene graph on these images, generate images using my and the authors' "image generation from scene graphs" model
3. Extract features from generated image, to be used as input to pretrained model
4. Answer a maximum of 30 questions about each image using pretrained model
5. Compute accuracy as percentage of exactly correct answers (classification)

Step 1a was required as the visual question answering model was not trained or evaluated on the entire Visual Genome dataset. Rather, it was primarily trained and evaluated on the VQA dataset, and therefore it was not able to generate every possible answer in Visual Genome. Additionally, it mapped each of these questions to vectors of integers and answers to integers, making combining data loaders necessary.

Setting up the framework to select images compatible was not trivial. First, I took functions from the data loader in the image generation code, then would load only if the corresponding ID appeared in the library used by the visual question answerer. This required using the data loader from the visual questioning algorithm to output all IDs and store them in a hash table along with corresponding question vectors and answer integers. Now, I could load feasible images along with questions, answers, question vectors, answer vectors and scene graphs.

Next, I loaded my and the authors' image generation model, properly processed the scene graphs and called a forward pass of their code. This was modified to evaluate using scene graph variables as opposed to json files used in authors' evaluation. Additionally, post-processing was required to get images in proper format to be used by visual question answering model.

From generated images, I formatted to be consistent with the visual question answering extract code, extracted features using a pretrained ResNet, and saved in another hash table mapping ID to features. From here, I could load scene graphs along with original images, and use the ID to lookup features, and question and answer vectors. I used these to call a forward pass of the visual question answerer and calculate results. Analysis required also mapping scene graphs to strings from the scene graph code and looking through images to get those with most interesting characteristics.

Many of the steps in the evaluation framework seem fairly straightforward. However, figuring out proper configuration, datatypes, and matching would often take many tries and was

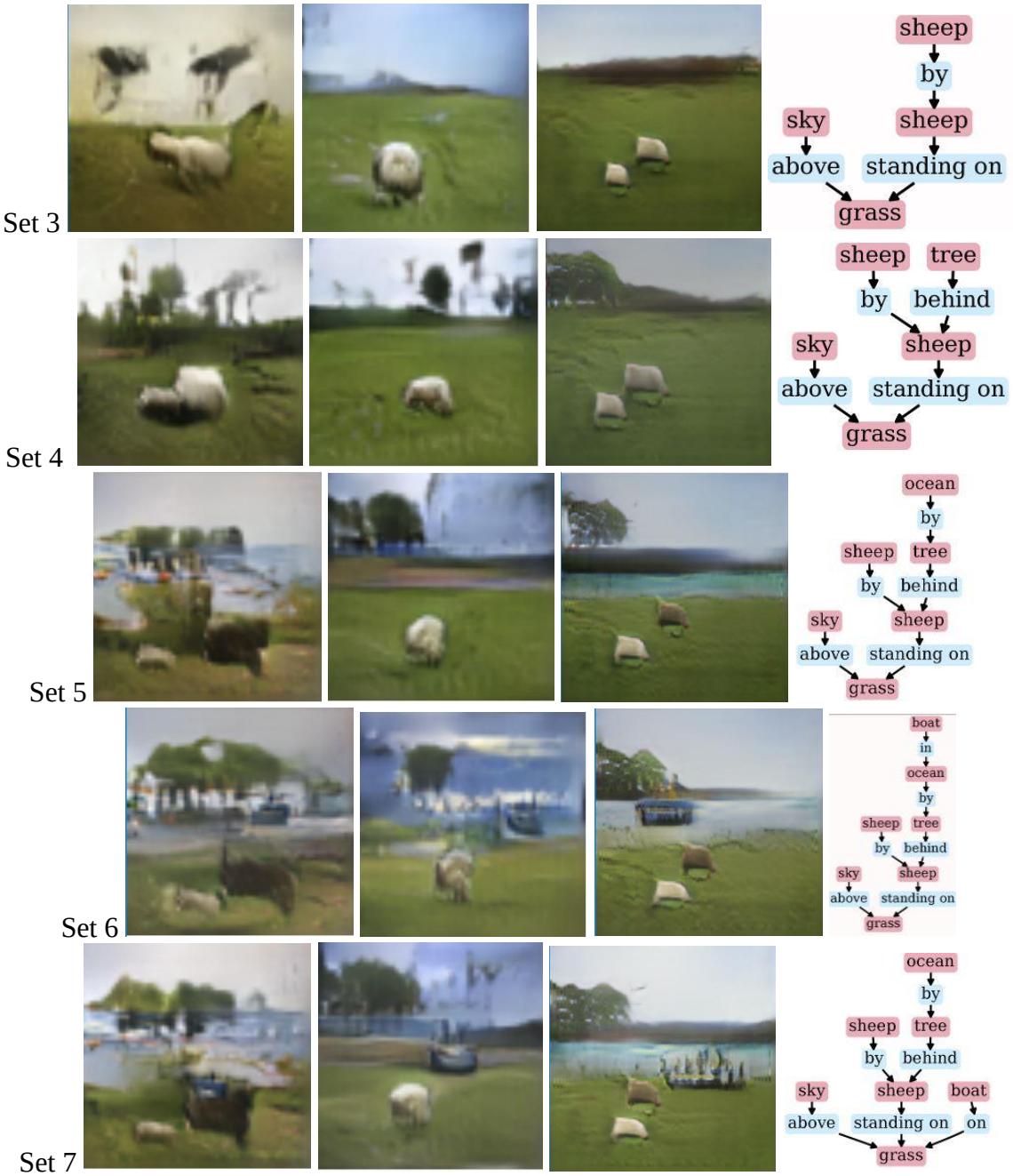
very slow. I also found even more difficulty setting up the visual question answering portion compared to the image generation portion because of memory issues. Setup required saving large amounts of extracted features on COCO, Visual Genome and the additional VQA dataset [14], which needed to be downloaded. Compatibility issues of the data representations in both models required good understanding of the code and substantial effort. This latter part of the project was therefore deceivingly difficult and was completed over the course of about a month. Code is publicly available at https://github.com/crockwell/sg2im. Evaluation primarily takes place using generate_imgs.py, extract_chris.py, eval_vqa.py, and vqa_analysis.py.

5. Replication: Qualitative Results

I display my images compared to those selected in the paper using both a 64x64 and 128x128 model. Note the below images in the paper were reported from 128x128 model, though I primarily compare to the authors' 64x64 model for this paper, which I refer to as "authors' model." When looking at images generated from other scene graphs it seems the images selected were, not surprisingly, some of the better results from the paper. This comparison is thus likely to be slightly biased toward the authors' images. This combined with the lack of some training details including split of datasets used meant the authors' model should be expected to outperform mine. Still, my replicated model seemed to perform pretty competitively. Keep in mind in both cases scene graphs can be pretty interpretable.

In most cases our model arguably captures the general idea of the scene graph, as does that of the authors. In Set 2, all images appear to have something like a sheep standing on grass below the sky. More complex examples get harder to see clearly, such as Set 5, but still capture the main idea. In this one, it does appear there is a sheep standing on grass, below the sky, in front of trees by the ocean, standing near another sheep (maybe). However, in both 64x64 models, we see clear errors. One error is in Set 3, in which images appear to only have one sheep. While the 128x128 model does not make any obvious mistakes in this setting, some of this could possibly be selection bias as their 64x64 is trained the same way and makes errors.

Fig 3. Left to right: my model, authors' model, authors' 128x128 model, scene graph



In general, the authors' model seems slightly more polished. Set 1 seems to have our zebra floating in the air, and images in Sets 3, 6 and 7 appear clearer. However, ours seems to do a better job at representing multiple sheep in Set 4 and possibly Set 5. As expected, it is difficult to properly evaluate images to compare these models. Qualitatively, they perform within a margin of error for such a small sample size, motivating the question answering evaluation.

6. Extension: Question Answering Results

Visual question answering is evaluated using as many question-answer pairs available up to a maximum of 30 questions per image from 500 images selected randomly from the test set.

Answering is treated as classification into one of 2000 classes, and accuracy is computed as proportion of correct classification. I subdivide accuracy by the number of words in an answer.

There are a couple clear takeaways from the quantitative results. Clearly, longer answers are less common and significantly more difficult than short ones. Accuracies between using images from our model, the authors' model, and ground truth are also very close, at a relatively low number. I will proceed to analyze these numbers with examples from the dataset.

Accuracy	Ground truth image	My model	Authors' model	# Questions
One Word	0.373	0.358	0.362	2031
Two Words	0.182	0.281	0.289	121
Three Words	0.062	0.00	0.01	97
Total	0.349	0.338	0.343	2249

By looking more deeply at question, answer, and image pairs we can get an intuition for why even on ground truth the model only gets about 35% of answers correct. This should not have a major impact on our results considering we are only interested in difference in performance between models, but is important to note nonetheless. However, it is possible a more idealized dataset without some of these issues would see a larger increase in accuracy upon answers generated from ground truth images compared to synthetic ones.

Issues with questions and answers aside, it is noteworthy that visual question answering is difficult, and there are examples of the algorithm missing questions that seem very obvious to a human. For instance, in **Fig. 4**, question 3: "What color are the poles?" is straightforward on the ground truth image, yet is missed.

Fig. 4. Left to right: ground truth, my model, authors' model, scene graph

Q1: What covers the ground? A: Snow. Correct, Wrong, Correct
Q2: What is the weather? A: Sunny. Correct, Correct, Correct
Q3: What color are the poles? A: Yellow. Wrong, Wrong, Wrong
Q4: Who are there? A: People. Wrong, Wrong, Wrong

However, there are also other reasons the model struggles. One is that some questions make little sense for a given picture, and are therefore difficult to answer even on the ground truth. For example, on the second set of images in **Fig. 4**, question 4: "Who are there" makes very little sense and is very difficult to answer given the image. The answer "People" does not clearly follow from the picture and not surprisingly the question answering model gets this question wrong on each image.

We also see limits of having too many classes and requiring exact classification in our problem in the first two questions in **Fig. 5**. In both questions, the model outputs "Palm" to all three images, yet in one question the slightly different ground truth is "Palm trees," meaning they

are counted as incorrect. Additionally, we see from these two questions that essentially duplicate questions exist – beyond those that are exactly the same and are filtered out.

Fig. 5. Left to right: ground truth, my model, authors' model, scene graph

Q1: What type of tree is in the picture? A: Palm. Correct, Correct, Correct
Q2: What type of trees are in the scene? A: Palm trees. Wrong, Wrong, Wrong
Q3: What are the words on the red sign? A: Do not enter. Wrong, Wrong, Wrong
Q4: What colors are the train cars? A: Red and green. Correct, Wrong, Wrong
Q5: What color shirt is the man driving the ban [sic] wearing? A: Red.

Some questions also are especially difficult to answer with the low-resolution input size of 64x64, which is much lower than that used in typical VQA settings. A pixelated image makes answering difficult even by a human on the ground truth. One example would be **Fig. 6** question 1: "What is in the hand of boy on the bench with the hat on?" As expected, the question answering algorithm incorrectly answers for ground truth and generated images. A human should also struggle with this answer even from ground truth image because of low resolution. Because of computational and model constraints, it is difficult to create larger resolutions images – however, this should be more feasible as computing power increases.

We can also gain insight from considering questions the model gets correctly. There are examples when generated images get answers correct that makes it appear better at generating images than it really is. Sometimes, this is because questions allow for likely guesses, even when the ground truth image provides little help to answering the question.

An example of this can be found in question 2 in **Fig. 6**. Despite it being very difficult for a human (and likely computer) to identify a woman wearing sunglasses, it correctly guesses "Black," as a human may. The point that the computer is essentially using prior knowledge rather than visual information is strengthened by the fact the model incorrectly answers question 3, "What is the woman sitting on the bench wearing?", which has a correct answer "Sunglasses." More evidence for this is that both generated images yield a correct answer on this question despite also getting this question wrong. Additionally, it seems difficult for a human to do the same by a human without reasoning or prior knowledge about likelihood of sunglasses being black.

Fig. 6. Left to right: ground truth, my model, authors' model, scene graph

Q1: What is in the hand of boy on the bench with the hat on? A: Sandwich. Wrong, Wrong, Wrong

Q2: What color are the sunglasses of the woman on the bench? A: Black. Correct, Correct, Correct

Q3: What is the woman sitting on the bench wearing? A: Sunglasses. Wrong, Wrong, Wrong

Q4: What color are the leaves? A: Green. Correct, Correct, Correct

Q5: What color are the trees leaves? A: Green. Correct, Correct, Correct

Q6: What part of a tent is the the [sic] background? A: Top. Correct, Correct, Correct

Q7: What color is the building? A: Brown. Correct, Correct, Correct

Guessing in general is made easier by the high proportion of one-word answers in the dataset. Of the 2249 question-answer pairs tested, 2031 of them (90.3%) were one word. Performance declined significantly beyond one word, suggesting the difference in difficulty. Furthermore, one can see many answers are colors or numbers, and these should yield a non-negligible blind accuracy rate. Of the 30 answers tested in **Fig. 6**, for example, these included "White" twice, "Green" three times, "Black" four times times, "2" three times, "Gray", "Brown," and "Blue". It therefore seems a one could get some accuracy from simply considering RGB channels of the image. It is also not surprising every time a count is requested all three models provide correctly "2".

Many questions corresponding to these answers are also pretty easy. In **Fig. 6**, "What color are the leaves?" is especially simple given nearly half of the ground truth image is green. Permutations of the same question, which are difficult to filter out, seem common on basic questions, such as questions 4 and 5 in **Fig. 6.**

In summary, there are many questions that the model is likely to get incorrect on all images, and many others it is likely to get correct on all images. However, performance between generated images and ground truth images is still closer than these two facts would suggest. A big part of the reason generated images score so well compared to ground truth images is because they likely exhibit characteristics most common of certain classes, regardless of whether those characteristics look realistic to humans. This combined with memorization of common results in training is a large drawback of using the visual question answering benchmark to assess generated images. **Fig. 6** is a damning example of this: all three pictures score the same 16/30 questions answered correctly, despite the generated images looking significantly worse than the ground truth. Question 5 in **Fig. 5** and questions 6 and 7 in **Fig. 6** are instances of surprising correct answers on synthetic images. The algorithm appears to have simply memorized **Fig. 6** question 7, that the building is brown, given neither a building, nor the color brown, appear in the scene graph.

Although it is often the case the more-realistic image scores better (e.g. **Fig. 7** and **Fig. 8**), it does not occur often enough to be convincing.

Fig. 7. Left to right: ground truth, my model, authors' model, scene graph

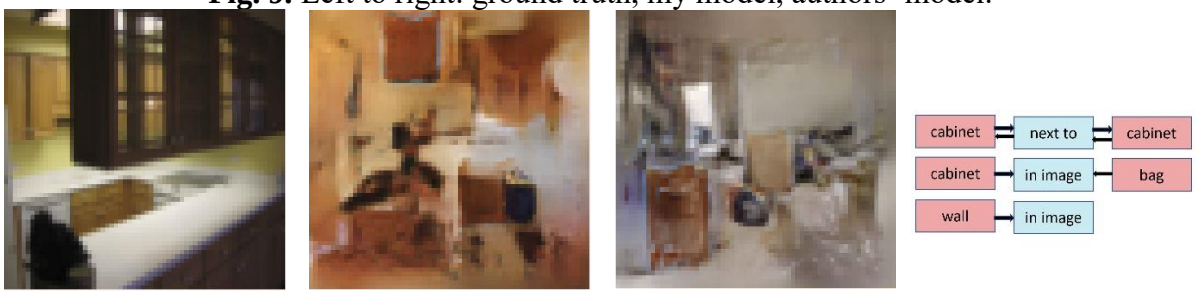
Questions answered correctly (left to right): 19/30, 8/30, 6/30

Fig. 8. Left to right: ground truth, my model, authors' model, scene graph

Questions answered correctly (left to right): 5/6, 2/6, 2/6

There are also examples where the generated images are actually setup to answer questions better, even by humans. In **Fig. 9**, for example, the question "What room was the picture taken?" was easier for me to answer at first glance from both my model and the authors' model than the ground truth (Answer: "Kitchen"). Here, the odd angle and coloring at first confused me as to what room the ground truth was, and likely did for the model too: the visual question answering model got this question correct on both generated images but not the ground truth. In this example, the ground truth gets 2/7 questions correct, while our model gets 4/7 and the authors' model gets 5/7 questions correct. In this case, the scene graph does not explicitly denote kitchen, but does indicate objects likely to appear in a kitchen (e.g. cabinet).

Fig. 9. Left to right: ground truth, my model, authors' model.



Scene graph:
Q1: What room was the picture taken? A: Kitchen. Wrong, Correct, Correct

In either case, our method appears to be potentially biased in favor of synthetic images because they can provide contextual clues to question answering algorithms despite not doing so to humans. A comparison to adversarial attack [3] seems appropriate: a feature extractor can recognize patterns in an image even when humans cannot. This is a limitation in VQA as

evaluation, but at least it should not bias comparison between our model and that of the authors. Because GANs have been more successful at realistic image generation [4], the existence of bias when using a model mostly supervised in earlier steps of generation may imply future work placing more emphasis on the discriminator is important.

Compared to the authors, our model yields 11 less questions answered correctly (.5%) out of 2,249 questions, which seems hardly significant at first glance. Because of the reasons stated earlier, however, especially that the difference between our model and the ground truth is only 25 questions (1.1%), the 11-question difference is more meaningful. Therefore, the VQA metric would suggest the author's model does perform measurably better on the task. Question 1 in **Fig 4.** would be one instance where the model more clearly resembles snow. This would also be an example of questions helping us identify differences between our model and that of the authors.

7. Conclusion and Takeaways

Because the paper's task is generating images from complex scene graphs, it is difficult to assess how well the authors' model does, much less compare our replication to that of the authors. Qualitatively, in some images our model looks better than that of the authors and in others the authors' looks better, though on average the slight edge goes to that of the authors. However, both models are clearly significantly worse than ground truth images. This is not especially surprising considering the difficulty of the task.

I attempted to get a good quantitative benchmark to compare both models with the ground truth using visual question answering. Due to constraints in the dataset, resolution, and potential bias, this was not as effective as desired. Nevertheless, it provided a deeper understanding of the limits of the authors' method and dataset. Additionally, it helped identify differences in synthetic images as well as images to consider for more analysis. The results from this study suggest again that the authors' model was slightly better.

The authors did not describe some details of training including how they split training between COCO and Visual Genome datasets, so with more powerful computing and more time, it is likely I could have better tuned the model. Still, comparing models is subjective enough to mean replication results were more than acceptable already.

Despite the problem being highlighted as especially difficult, I find generation methods to be great candidates for research because of the aspiration of model-building. The Visual Genome dataset has limitations, as highlighted in this study, but conceptually is very pleasing because of the variety of data that can be used to train and evaluation model understanding. I am inspired to perhaps pursue improving or harnessing similar datasets in order to move towards neural networks that can model-build.

8. References

[1] Johnson et al., "Image generation from scene graphs". In CVPR 2018.
[2] Lake et al., "Building machines that learn and think like people". In Behavioral and brain sciences 2017.
[3] Carlini and Wagner, "Towards evaluating the robustness of neural networks". In SP 2017.
[4] Reed at al., "Generative adversarial text-to-image synthesis". In ICML 2016.
[5] Zhang et al., "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks". In ICCV 2017.

[6] Lin et al., “Microsoft coco: Common objects in context”. In ECCV 2014.
[7] Krishna et al., “Visual genome: Connecting language and vision using crowdsourced dense image annotations”. In IJCV 2017.
[8] T. Salimans et al., “Improved techniques for training gans”. In NeurIPS 2016.
[9] Ben-Younes et al., “Mutan: Multimodal tucker fusion for visual question answering”. In CVPR 2017.
[10] He et al., “Deep residual learning for image recognition”. In CVPR 2016.
[11] Kiros et al., “Skip-thought vectors”. In NeurIPS 2015.
[12] Fukui et al., “Multimodal compact bilinear pooling for visual question answering and visual grounding”. Arxiv:1606.01847 2016.
[13] Tucker, “Some mathematical notes on three-mode factor analysis”. In Psychometrika, 1966.
[14] Antol et al., “Vqa: Visual question answering.” In ICCV 2015.