Improving Realism of Scene Graph to Image Generation

Johann Lingohr
johannlingohrgmail.com

Jiefei Li
jeffuvic@ece.ubc.ca

Jay Fu
ngaifu16gmail.com

1 Introduction

Generating realistic images is an important area in computer vision because it provides insights into how our algorithms understand the visual world and has practical use for artists and graphic designers. With developments in machine learning and deep learning algorithms, there has been a significant focus on new image generating algorithms. Generative adversarial networks (GANs) [9] have become one of the most popular neural network models used to generate realistic images. These models work by jointly training an image generator for image synthesis and a discriminator for determining whether the synthesized image is fake or real. Extending this model, *Reed et al.* [2] generate images using GANs conditioning on text and are able to generate high-resolution images. *Huang et al.* [1] further improve on this using a two-stage architecture resulting in 256x256 photo-realistic images. This approach shows us, importantly, that we can break up hard tasks into manageable sub-tasks to improve results, going from a coarse-to-fine level of detail in the process. *Xu et al.* [10] go further by leveraging the attention mechanism to draw images region-by-region conditioned on relevant words in a long text description, resulting in images with much more detail. Finally, Nvidia [11] incorporate style transfer to create a style-based image generator to replace the original generator in the GAN model, bringing the realism of synthesized images to an impressive level.

While these techniques generate realistic looking images when the input sentence describes simple scenes, they do not perform well when trying to generate images corresponding to more complex scenes. Branching from these GAN models is another approach to discover the role of scene graph and scene layout in generating images from text. These allows us to explicitly represent objects and relationships in complex sentences. *Johnson et al.* [3] use graph convolution neural networks to extract features and objects from scene graphs and are able to generate images preserving relationships among multiple objects, but the images are low-resolution only and does is unable to generate good images when there are multiple relationships among the objects. *Tripathi et al.* [4] attempt to improve on this by introducing scene graph context to encourage generated images to appear realistic and better respect the scene graph relationships.

The goal of this project is to build on these methods. Specifically, we aim to improve the realism of synthesized images.

2 Method

Our framework builds from [3] and [4]. In their method a scene graph is encoded into object embedding vectors using a Graph Convolution Neural Network. These vectors are transformed into a scene layout using an object generator network consisting of a mask regression network to predict segmentation masks and a bounding box regression network to predict bounding boxes. These are combined to form the object layout and summing over all object layouts producing the scene layout. The scene layout and random noise is then passed to a Cascade Refinement Network (CRN) [8] consisting of several up-sampling modules of increasing size to generate the final image.

Building on these methods, we implement a modified CRN in which skip layer connection are added between up-sampling modules to improve quality. We believe this will help better preserve original features in the forward path. We also incorporate a scene graph context [4] that pools features from

the graph convolution neural network. The features are then embedded by a fully connected network and fed to each up-sampling module in CRN to provide context to the image generator. However, instead of feeding the embedding only to the first convolution layer of the CRN, we will pass the embedding to each up-sampling layer in the CRN so that more relational context could be preserved during the up-sample process.

2.1 Loss

As a baseline we follow [3] and [4] by training an image generator G_{img} conditioned on inputs, scene layout, and scene context embedding, jointly with an image discriminator D_{img} and object discriminator D_{obj}. The network is trained to minimize the weighted sum of size losses:

- *Box Loss* L_{box} penalizing the L_1 difference between coordinates of the ground-truth bounding box and predicted bounding box
- *Mask Loss* L_{mask} penalizing differences between ground truth masks and predicted masks
- *Pixel loss* L_{pix} penalizing the L_1 pixel-wise difference between ground-truth image and generated image
- *Adversarial image loss* L_{GAN}^{img} from D_{img} that encourages generated images to be realistic and relevant to the scene context
- *Adversarial object loss* L_{GAN}^{obj} from D_{obj} that encourages objects to appear realistic
- *Auxiliary classifier loss* L_{AC}^{obj} from D_{obj} encouraging generated objects to be classified by the object discriminator

One of our goals will be to extend and improve this loss function. Following *Odena et al.* [15] we hope to explore alternative ways to create an auxiliary loss function in a way to improve training. We believe this is particularly important for this problem since [4] observe that using layout in the image discriminator lead to mode collapse. We hope defining a better loss function will help prevent mode collapse, resulting in better training and ideally better generated images.

3 Experiments

3.1 Datasets

Figure 1. An image annotated with segmentation mask and bounding box in COCO

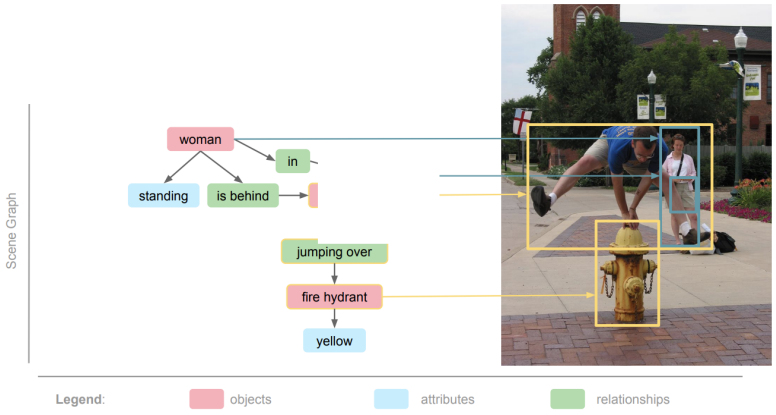

Figure 2. An example of an image annotated with scene graph in Visual Genome [14]

Similar to previous methods, we will train our model using the COCO-Stuff [5] and Visual Genome [6] datasets. The COCO-Stuff dataset contains 40k training and 5k validation annotated images with bounding boxes and segmentation masks (see Figure 1), as well as providing 80 thing categories and 90 stuff categories. Similar to [3] and [4], the segmentation masks and the bounding boxes labelled in COCO stuff can be used as ground truth to train the Graph Convolution Network to predict desired scene layout. On the other side, the Visual Genome dataset contains over 100k images annotated with their scene graphs (see Figure 2), which are the important ground truth which will be used to train the automatic scene graph generation model to predict high-quality scene graphs. Lastly, we can use the image captions and annotation in COCO to construct scene graphs based on the 2D image coordinates of the objects. These scene graph will be passed to image generator to create scene layout first and then synthesize photographic images. COCO dataset's real life image collection, in which most images contains multiple object with relationships,is considered as the best training data to train the generator and discriminator of the GAN model jointly.

3.2 Evaluation

In order to evaluate a generated image we need a measure that answers 1) how realistic does the generated image appear? 2) how well can we recognize the different objects in the image? 3) How diverse are the generated images?

To this end we use the Inception Score [7], which evaluates both the quality of generated images and diversity by applying a pre-trained classifier on generated images. Correctly predicting class labels for objects in the generated image should correspond to the generated images looking realistic.

Additionally we will explore measuring diversity by exploring the model's latent space using interpolation. *Odena et al.* [15] show that one way to check whether a model overfits is to see whether or not one observes that discrete transitions in interpolated images and regions in latent space correspond to meaningful images. We will borrow this idea to see whether our generator learns meaningful semantic features.

4 Project Management

4.1 Milestones

March 2	Finalize project structure
March 6	Get text-to-scene-graph model functioning
March 7	Finish proposal
March 10	Finish generating scene graphs for COCO dataset
March 20	Finish prototyping the GCN, CRN and GAN modules individually (Tentative)
March 27	Integrate all three modules with scene graph context network
March 31	Finish testings and start training

4.2 Teamwork Breakdown

Text-to-scene-graph module	Johann, Jeffery, Jay
Disciminator Module	Johann
GCN Module	Jeffery
CRN Module	Jay
Integration, Training, Testing	Johann, Jeffery, Jay

References

[1] Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft and Serge Belongie. Stacked Generative Adversarial Networks, 2016; arXiv:1612.04357.

[2] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele and Honglak Lee. Generative Adversarial Text to Image Synthesis, 2016; arXiv:1605.05396.

[3] Justin Johnson, Agrim Gupta and Li Fei-Fei. Image Generation from Scene Graphs, 2018; arXiv:1804.01622.

[4] Subarna Tripathi, Anahita Bhiwandiwalla, Alexei Bastidas and Hanlin Tang. Using Scene Graph Context to Improve Image Generation, 2019; arXiv:1901.03762.

[5] Holger Caesar, Jasper Uijlings and Vittorio Ferrari. COCO-Stuff: Thing and Stuff Classes in Context, 2016; arXiv:1612.03716.

[6] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein and Fei-Fei Li. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations, 2016; arXiv:1602.07332.

[7] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford and Xi Chen. Improved Techniques for Training GANs, 2016; arXiv:1606.03498.

[8] Qifeng Chen and Vladlen Koltun. Photographic Image Synthesis with Cascaded Refinement Networks, 2017; arXiv:1707.09405.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,Aaron Courville and Yoshua Bengio. Generative Adversarial Nets, 2014; arXiv:1406.2661.

[10] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang and Xiaodong He. AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks; arXiv:1711.10485.

[11] Tero Karras, Samuli Laine and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks; arXiv:1812.04948.

[12] Sebastian Schuster, Ranjay Krishna, Angel Chang, Li Fei-Fei, and Christopher D. Manning. Generating Semantically Precise Scene Graphs from Textual Descriptions for Improved Image Retrieval;

[13] Seunghoon Hongy, Dingdong Yangy, Jongwook Choiy, Honglak Lee. Inferring Semantic Layout for Hierarchical Text-to-Image Synthesis; arXiv:1801.05091.

[14] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein and Li Fei-Fei. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations; arXiv:1602.07332

[15] Augustus Odena, Christopher Olah and Jonathon Shlens. Conditional Image Synthesis With Auxiliary Classifier GANs, 2016; arXiv:1610.09585.