# Artificial Intelligence Enabled Image Generation from Text

A Project Report
Presented to
The Faculty of the College of
Engineering

San Jose State University
In Partial Fulfillment
Of the Requirements for the Degree

**Master of Science in Software Engineering**

By

Sivaranjani Kumar, Akshaya Nagarajan, Pooja Patil, Vigneshkumar Thangarajan

May 2021

**APPROVED**

______________________________

Vishnu Pendyala, Project Advisor

# ABSTRACT

## Artificial Intelligence Enabled Image Generation from Text

By

Sivaranjani Kumar, Akshaya Nagarajan, Pooja Patil, Vigneshkumar Thangarajan

Synthesizing high-quality photo-realistic images is one of the challenging problems in computer vision and has a plethora of practical applications. Of many techniques that are available to solve this, Generative Adversarial Networks (GANs) have achieved greater success in recent times. Similarly, generating images conditioned on textual description has many applications and to list a few – a quick visual summary for a text paragraph to enhance the learning experience for students and real-time image generation in sports for a commentary text. Although GANs are successful in these tasks, they are difficult to train, unstable, and are sensitive to hyperparameters. Previous models like Stack-GAN face challenges in generating images that are related to text description as they were using the same text embedding throughout the training process. In addition, these models suffer in the image refinement process, if the initial generated image is of poor quality.

The project module is divided into following modules, first module is generating images with multiple objects from the given text. We used OP-GAN model which will generate photo realistic images considering the multiple objects mentioned in the text description. For instance, if the textual description is “two skiers are posing on a snowy slope”, the multiple objects in the images are “two skiers” and the “snowy slope”. In addition to the GAN model, we also plan to deploy the model as a web application. Here the user can enter text description, in turn resulting in generation of images. Based on the GAN model described above, appropriate photo realistic images will be generated for the given input text. The second module is evaluating OP-GAN model using Semantic Object Accuracy (SOA). This new metric uses a pre-trained object detector (YOLOv3 Network) to check whether the generating images contained the object mentioned in the image caption (text description).

## Acknowledgments

The authors are deeply indebted to Professor Vishnu Pendyala for his invaluable comments and assistance in the preparation of this study.

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1. Table 3. Oxford dataset by Inception Score (IS) Project Overview

## Introduction

In recent years, Generative Adversarial Network (GANs) have shown major improvements and capabilities to generate photo realistic images given an input of textual description. Over the years, GANs have achieved State of the Art performances in synthesizing images containing single object given the textual descriptions as input to the model. Our approach focuses on synthesizing high-resolution images with multiple objects present in an image, given the textual description of the images. Generating multiple objects in a single image is a challenging problem, as there are high chances of missing out the objects and overlapping of created objects in the image. There were attempts in the past to generate images from directed scene graphs, but it didn't achieve significant results. In addition to that, the existing evaluation metrics like Inception Distance doesn't really align with human eye evaluation. In this paper, we will be seeing a new state of art method to generate synthetic images and a new better way for evaluating the same.

## Proposed Areas of Study and Academic Contribution

In earlier research, the process of creating new images from text-based natural language descriptions also known as image synthesis were heavily based or relied on analysis of a word to image correlation. However, recent advancements in deep learning methods and deep generative models can generate realistic visual images using neural network models to generate captivating images based on natural language descriptions of text. A successful approach to generate realistic images based off on text descriptions proposed by [1] called StackGAN (Stacked Generative Adversarial Network), defines a model with two-stages. In StackGAN++, which is second version of StackGAN has a noise vector which is introduced along with the conditioning variables [1] which is input to the first generator. A novel approach which is similar to StackGAN++ is [4] AttnGAN (Attentional Generative Adversarial Network).

In addition to generating text, embedding with conditional variables like previous [8], [1],[2] AttnGAN's text encoder component generated an additional separate text embedding based on individual words. In addition to attentional generative network, Deep Attentional Multimodal Similarity Model (DAMSM) [9] is also the major contributions of AttnGAN [4] It tends to add "attentional" details region by region on specific regions of the image independently. After the final stage outputs its high-resolution image, DAMSM is used to calculate the similarity between the generated image and the text embedding at two levels, sentence level and word level.

Most of the existing text to image synthesis models/methods tend to depend heavily on the quality of initial image like [8], [1], [9] and then will refine the initial generated images to a high-resolution one. If the initial image is not well generated then, the quality of images from the refinement process won't be of acceptable quality. Despite the recent advancement in Generative Adversarial Networks to synthesize synthetic images, there are some challenges presented. Visual realism is hard to achieve. The standard of the image quality has scopes of improvements. The image generated should precisely illustrate the given textual summary. The model is unlikely to produce the expected result when it is evaluated with a previously unseen scenario.

We set out to resolve these challenges by combining the technique of text embedding and capabilities of Generative Adversarial Networks to synthesize images. The essential terms in the caption are captured to develop individual objects in the image, which are then combined once the background is generated, forming images with fine-grained details. We intend to use multiple discriminators to produce images with high resolution. We aim to develop an efficient deep learning model that demonstrates the capability to generate multiple objects in an image with in-depth details that are seemingly plausible.

## Current State of the Art

The most commonly available /benchmark datasets used with GAN models for image synthesis are CUB dataset which contains 200 birds' images along with text description, COCO dataset which contains 328k images with 91 different object types and Oxford dataset contains 102 categories of flowers with 40-258 images each and text descriptions. Additionally, COCO dataset’s images also contain images with multiple objects in comparison to CUB and Oxford datasets which has images with only single object along with their text descriptions.

Text to Image generation GAN models is judged based on the evaluation metrics. Out of many evaluation metrics used, Fréchet Inception Distance (FID) [10] is commonly used as an evaluation metric, which compares the images generated with the real images from the data distribution. A low FID [6] is better as this means there is more relationship between the synthetic images being generated by the GAN models and real images. Inception Scores (IS) calculated the randomness of conditional distribution of images along with the marginal distribution of generated images, which are supposed to be low and high respectively. Both low and high randomness of conditional distribution and marginal distribution respectively are desired features as low randomness means the images are of same data distribution and high randomness means that the images generated are diverse.

The performance of different GAN methods used for Text to Image synthesis for datasets discussed above along with the evaluation metrics is reported in the table below,

| Rank | Model | FID | SOA-C | IS |
|---|---|---|---|---|
| 1 | OP-GAN | 24.70 | 35.58 | 27.88 |
| 2 | DM - GAN |  | 33.44 | 30.49 |
| 3 | AttnGAN |  | 25.88 | 25.89 |
| 4 | StackGAN + OP | 55.30 |  | 12.12 |

*Table 1. COCO dataset by SOA-C*

| Rank | Model | FID | IS |
|---|---|---|---|
| 1 | DM-GAN | - | 4.75 |
| 2 | Attention-Driver Generator | - | 4.58 |
| 3 | MirrorGAN | - | 4.56 |
| 4 | AttnGAN | - | 4.36 |
| 5 | StackGAN-V2 | 15.30 | 3.82 |
| 6 | StackGAN-V1 | 51.89 | 3.70 |
| 7 | StackGAN | - | 3.7 |

*Table 2. CUB dataset by Inception Score (IS)*

| Rank | Model | FID | IS |
|---|---|---|---|
| 1 | StackGAN-V2 | 48.68 | 3.26 |
| 2 | StackGAN-V1 | 55.28 | 3.20 |

| | | | |
|---|---|---|---|
| 3 | StackGAN | | 3.2 |

*Table 3. Oxford dataset by Inception Score (IS)*

# Chapter 2. Project Architecture

## Introduction

The primary objective of this project is to develop a GAN model which has the capability to generate images containing multiple objects based on the text descriptions. The model will have semantic intelligence of what part of text is used to generate the corresponding parts of images. The model strives to develop realistic images that are closely related to description.

In general, GANs have two different networks: a generator which generate images identical to the one given in input samples and a discriminator to differentiate between the real and generated images. AttnGAN architecture is used as a base architecture for our project. It works based on conditional GAN where attention and additional information is conditioned on the generator and three discriminators. Attention is basically, giving importance to different words from the caption to the specific regions of the image.

## Architecture Subsystems

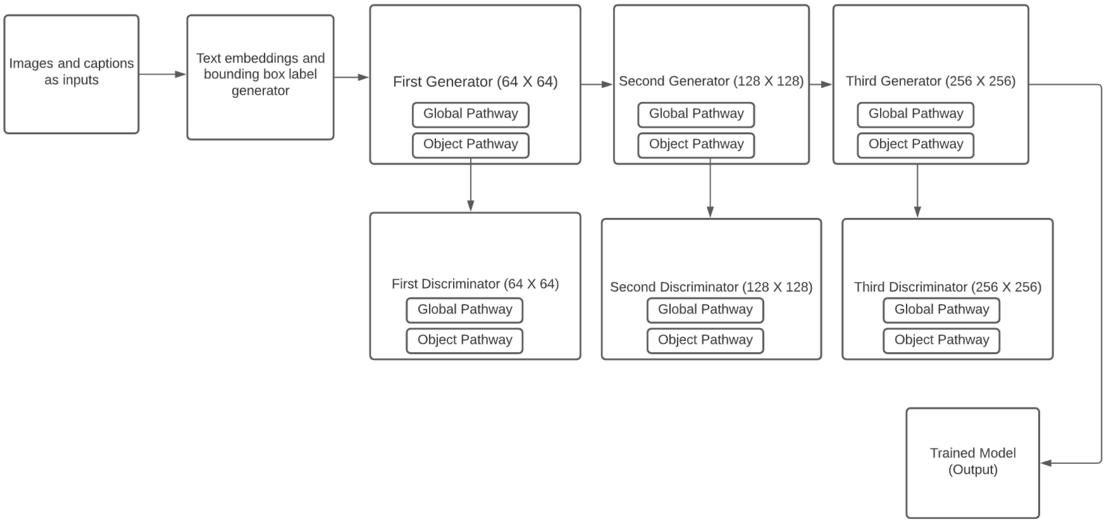
Images and captions as inputs
Text embeddings and bounding box label generator
First Generator (64 X 64)
Global Pathway
Object Pathway
Second Generator (128 X 128)
Global Pathway
Object Pathway
Third Generator (256 X 256)
Global Pathway
Object Pathway
First Discriminator (64 X 64)
Global Pathway
Object Pathway
Second Discriminator (128 X 128)
Global Pathway
Object Pathway
Third Discriminator (256 X 256)
Global Pathway
Object Pathway
Trained Model (Output)

*Figure 1. High-level Architecture Diagram*

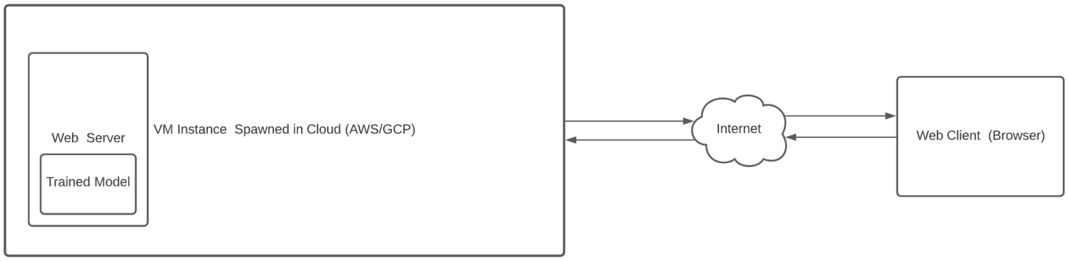

*Figure 2. Deployment Diagram*

A deep attentional multimodal similarity model (DAMSM) helps the generator by sending the added information and response for the generated images and its similarity with the captions. Multiple object pathways have been employed in the architecture where it is applied several times, at different locations and for each object. Each of the three generator and three discriminators have object pathways to perfect the features at each level of resolution (64 X 64, 128 X 128, 256 X 256). Whereas the previous approach used only one object pathways in both the generator and discriminator at a low resolution.

Object's that are associated with the caption are represented in the one-hot vector and the bounding box provides the object's location and size. Object pathway is initialized as an empty zero tensor, after processing all the objects and extracting features, it will have specific feature at specific location and others are marked as zero. A conditioning label is calculated using the one hot vectors, text embeddings and noise vector. This label is consumed by first object pathway in the generator to extract the features and transform them with respect to the bounding box using spatial transformer network (STN). Global pathways generate the features by receiving the individual objects location and size and apply convolutional layer in order to get layout encoding. At each higher level, objects are conditioned, and features are extracted repeatedly in the same way as from the previous level for each respective resolution. In the process, features are reconditioned by applying several convolutional layers and up-sampling technique.

The discriminator also contains global pathway and object pathway. In object pathway, features are extracted using STN and applies several convolutional layers with respect to the location of the bounding box. In global pathways, an input image is processed by applying convolutional layer several times with stride two to reduce the resolution of the image and the process is repeated continuously until it reaches 4 X 4 resolution. Conditional and unconditional losses are calculated for both generator and discriminator and are optimized by comparing the extracted features with the text embeddings of the image. Thus, the output is a fine-grained image of multiple objects with high resolution.

# Glossary

**AttnGAN**

Attentional Generative Adversarial Network. A machine learning model, generating images from textual description allowing attention-driven, multi-stage refinement to generate images.

**COCO**

Common Objects in Context. A large-scale dataset used for applications like captioning, object-detection and segmentation.

**CUB**

Caltech-UCSD Birds. An image dataset of two hundred bird's species.

**GAN**

Generative Adversarial Network. A machine learning model which has two neural networks, namely Generator responsible to generate realistic images, and a Discriminator responsible to discriminate real and fake images generated by Generator.

**OPGAN**

Object Pathway Generative Adversarial Network. A machine learning model, which specifically models individual objects based on text description to generate images.

**STACKGAN**

Stacked Generative Adversarial Network. A machine learning model which consists of two stages to generate images from text.

# References

[1] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, Z., Huang, X., and Metaxas, D. (2017b). "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks",in *IEEE International Conference on Computer Vision (ICCV), Venice, pages.*

[2] Zhang, G., Tu, E., and Cui, D. (2017a). "Stable and improved generative adversarial nets (gans): A constructive survey", in *Proc. The International Conference on Image Processing.*

[3] Zhang, Han., Xu, Tao., Li, Hongsheng., Zhang, Shaoting., Wang, Xiaogang., Huang, Xiaolei., & N. Metaxas*, Dimitris. (2018, June). "StackGAN++: Realistic image synthesis with stacked generative adversarial networks", in *IEEE, 2018*.

[4] Xu, Tao., Zhang, Pengchuan., Huang Qiuyuan., Zhang, Han., Gan, Zhe., Huang, Xiaolei., & He, Xiaodong. (2017, Nov). "AttnGAN: Fine-grained text to image generation with attentional generative adversial networks", *in Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2018, pp. 1316-1324.

[5] Hinz, Tobias., Heinrich, Stefan., & Wermter, Stefan. (2019, Jan). "Generating multiple objects at spatially distinct locations", in *Int. Conf. Learn. Representations*, 2019.

[6] Zhu, Minfeng., Pan, Pingbo., Chen, Wei., Yang, Yi., (2019, Apr). "Dm-gan:Dynamic memory generative adversial networks for text-to-image synthesis", in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*., 2019, pp. 5802-5810.

[7] Agnese, Jorge., Herrera, Jonathan., Tao, Haicheng., & Zhu, Xingquan. (2019, Oct). "A survey and taxonomy of adversial neural networks for text-to-image synthesis", in *IEEE*, 2019.

[8] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016b). "Generative adversarial text to image synthesis", in *Proc. The International Conference on Machine Learning (ICML).*

[9] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. (2018b). "Stackgan++: Realistic image synthesis with stacked generative adversarial networks", in *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 1801.05091v2."

[10] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2018). "Gans trained by a two time-scale update rule converge to a local nash equilibrium", in *CoRR,* 1706.08500v6.

# Appendices

## Appendix A. Description of Implementation Repository

- Install all the required packages using pip install –r requirements.txt command. All the required packages are listed in the requirements.txt file.
- All the model hyperparameters are maintained in separate files. Two different yaml files are defined for model training and model evaluation.
- The name of the dataset, GPU IDs, number of workers, number of branches, maximum epochs, generator learning rate, discriminator learning rate, and Gamma values for label smoothing is defined in training yaml file.
- Similarly, the model hyperparameters like generator model, batch size, pre-trained text embedding network, and dimensions for GAN and embedding network are defined in the evaluation yaml file.
- The model.py file contains the code for RNN encoder, CNN encoder, Generator networks, and Discriminator networks for different kinds of image resolutions like 64 X 64, 128 X 128, and 256 X 256.
- The trainer.py file contains the code for the training pipeline. Preparing training data and to compute text embeddings is the first step. Second, it generates synthetic images. Third, it updates the discriminator network. Finally, it updates the Generator network to maximize its objective function and minimizing the Discriminator's objective function.
- In the miscc folder, we have utility functions defined. Individual module's loss functions are defined in losses.py, matrix operation functions are defined in utils.py and program global variables are defined in config.py.

## Appendix B. Steps to train in SJSU's HPC:

- There are two ways to run the program in compute nodes/GPU nodes.
- Sbatch – Slurm Batch job requires us to define a batch script with correct parameters.
- Srun – Interactive way of running a Job. This has a limitation concerning the lifetime of the program. It supports the program to run only a maximum of 5 hours.
- Start a screen. Linux screens are used to keep the shell programs running even the client terminal gets disconnected.
- Use this Srun command to start the program:
- srun -p gpu --gres=gpu --pty /bin/bash - This command will help us to land in the GPU node that is currently available.
- Use this command to select python2 as our environment – "module load python3"
- Virtualenv is available only after selecting the python module. Now use virtualenv to select our project-specific virtual environment.
- Do this command in python shell to verify if the device <device-id> is GPU:

- torch.cuda.set_device(device-id). Upon executing this command, no error is expected.
- Start the training using “sh train.sh 0”. The 0 indicates the device id.
- Later re-attach to the same screen to check the run logs.