

1-1-2023

## Reconnoitering Generative Deep Learning Through Image Generation From Text

Vishnu S. Pendyala  
*San Jose State University*, [vishnu.pendyala@sjsu.edu](mailto:vishnu.pendyala@sjsu.edu)

VigneshKumar Thangarajan  
*PayPal*

Follow this and additional works at: [https://scholarworks.sjsu.edu/faculty\\_rsca](https://scholarworks.sjsu.edu/faculty_rsca)

---

### Recommended Citation


Vishnu S. Pendyala and VigneshKumar Thangarajan. "Reconnoitering Generative Deep Learning Through Image Generation From Text" *Deep Learning Research Applications for Natural Language Processing* (2023): 113-131. <https://doi.org/10.4018/978-1-6684-6001-6.ch007>

This Contribution to a Book is brought to you for free and open access by SJSU ScholarWorks. It has been accepted for inclusion in Faculty Research, Scholarly, and Creative Activity by an authorized administrator of SJSU ScholarWorks. For more information, please contact [scholarworks@sjsu.edu](mailto:scholarworks@sjsu.edu).

# Chapter 7

## Reconnoitering Generative Deep Learning Through Image Generation From Text

**Vishnu S. Pendyala**

 <https://orcid.org/0000-0001-6494-7832>

*San Jose State University, USA*

**VigneshKumar Thangarajan**

*PayPal, USA*

### ABSTRACT

*A picture is worth a thousand words goes the well-known adage. Generating images from text understandably has many uses. In this chapter, the authors explore a state-of-the-art generative deep learning method to produce synthetic images and a new better way for evaluating the same. The approach focuses on synthesizing high-resolution images with multiple objects present in an image, given the textual description of the images. The existing literature uses object pathway GAN (OP-GAN) to automatically generate images from text. The work described in this chapter attempts to improvise the discriminator network from the original implementation using OP-GAN. This eventually helps the generator network's learning rate adjustment based on the discriminator output. Finally, the trained model is evaluated using semantic object accuracy (SOA), the same metric that is used to evaluate the baseline implementation, which is better than the metrics used previously in the literature.*

### INTRODUCTION

The objective of this chapter is to explore generative deep learning through the specific example of generating images from text with some control over the placement of objects in the generated image. Expressing ideas in text is often much easier than doing the same in pictures or figures. Coming up with figures is an important skill that is often a formidable challenge even for human beings. In a sense, figures capture the latent semantic space of the corresponding text. Deep learning has been quite ef-

DOI: 10.4018/978-1-6684-6001-6.ch007

fective in processing natural language by capturing its latent space in the language models. With this background, we framed our research question: *to what extent can deep learning systems capture a pictorial representation of a given text and exercise control over the generated image?* We went on to search the literature to survey the existing work in this area and tried to leverage some of it as detailed in the following sections. In general, synthesizing high-quality photo-realistic images is a challenging computer vision problem and has a plethora of practical applications. Generating multiple objects in an image is an even more challenging problem, as there are high chances of missing out on the objects and overlapping of created objects in the image.

Generative Adversarial Networks (GANs) have shown major improvements and capabilities in generating photo-realistic images given an input of textual description. GANs have achieved superlative performance in synthesizing images containing a single object given the textual descriptions as input to the model. Figure 1 shows real-looking images of fictitious people that were generated using a GAN on the website, <https://thispersondoesnotexist.com/>. As can be seen, it is hard to tell the images are fake. The generator and the discriminator are neural networks that play a minmax game, acting as adversaries (Karras et al., 2020) to produce the astoundingly real-looking images. The generator starts with random Gaussian noise and improves in generating

*Figure 1. Images of fictitious people generated on <https://thispersondoesnotexist.com> using a GAN*



real-looking images over several iterations of feedback from the discriminator. The generator tries to make the discriminator believe that the image is that of a real person, while the discriminator acts as an adversary by denying the generator's claim as much as possible. Eventually, when the generator produces images like in Figure 1, the discriminator agrees with the generator's claim and the cycle stops. Given the outstanding performance of GANs combined with the advances in using pretrained deep learning models at text understanding, generating images conditioned on a given textual description seems feasible.

Generating images conditioned on textual description has many applications such as generating 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. The images generated can depict anything from anywhere imagination can take one. For instance, images of flying lions, four-eyed tigers, and flowers that can see with eyes can all be generated by specifying the appropriate words in the text. There were attempts in the past to generate images from directed scene graphs, but they did not achieve significant results. In addition to that, the existing evaluation metrics like Inception Distance do not align with human eye

evaluation. Although GANs are successful in these tasks, they are difficult to train, are unstable, and are sensitive to hyperparameters. Previous models like StackGAN faced 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 image generated at the first attempt is of poor quality.

Object Pathway GAN (OP-GAN) is often the baseline model, which generates photo-realistic images consisting of multiple objects described in the text description. For example, if the textual description is, “two skiers are posing on a snowy slope”, the objects expected in the image are “two skiers” and the “snowy slope”. The generator network takes labels and other images’ metadata like bounding box location as input to generate fine-grained images and the discriminator network feeds on the real image and fake image to classify them into the right bucket. For evaluating the model, we use Semantic Object Accuracy (SOA) (Hinz et al., 2020), which the original implementation in the current literature also uses. SOA is more practical in evaluating the synthetic images compared to other metrics like the Inception score (IS) (Salimans et al., 2016), which fails to significantly indicate the semantic content hidden in the images.

## **BACKGROUND**

In earlier research literature, the process of creating new images from text-based natural language descriptions also known as image synthesis heavily relied upon analysis of a word-to-image correlation. However, recent advancements in deep learning methods and deep generative models can create real-looking images using neural network models and can be extended to generate captivating images based on descriptions in natural language. A successful approach to generate realistic images based on text descriptions uses StackGAN (Stacked Generative Adversarial Network) (Zhang, et al., Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks, 2017), and defines a model with two stages. StackGAN++, which is the second version of StackGAN has a noise vector that is introduced along with the conditioning variables, which is input to the first generator. A novel approach that is like StackGAN++ is AttnGAN, Attentional Generative Adversarial Network (Xu, et al., 2018).

In addition to generating text, embedding with conditional variables like previous work (Zhang, Tu, & Cui, 2017), (Reed, et al., 2016), and (Zhang, et al., Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks, 2017), AttnGAN’s text encoder component generated an additional separate text embedding based on individual words. In addition to the attentional generative network, Deep Attentional Multimodal Similarity Model abbreviated DAMSM (Zhang, et al., Stackgan++: Realistic image synthesis with stacked generative adversarial networks, 2018) is also the major contribution of AttnGAN (Xu, et al., 2018). 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 compute the closeness of the generated image embedding and the text embedding at both sentence and word level.

Most of the existing text-to-image synthesis models/methods tend to depend heavily on the quality of the initial image (Reed, et al., 2016), (Zhang, et al., Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks, 2017), (Zhang, et al., Stackgan++: Realistic image synthesis with stacked generative adversarial networks, 2018) and then 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 will not be of acceptable quality. Despite the recent advancement in Generative Ad-

versarial Networks to synthesize synthetic images, there are some challenges presented. Visual realism is hard to achieve. The standard of the image quality has scope for improvement. 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 look for ways to resolve these challenges by combining the technique of text embedding and the capabilities of Generative Adversarial Networks to synthesize images. Several research articles discussed in the last one or two years have focused on text-image pairing. A taxonomy and timeline of various approaches to generating images from text (Agnese et al., 2020) shows that the enhancements in the approaches can be categorized into semantic, resolution, diversity, and motion. However, the work described in this chapter is predominantly based on the key idea of using object pathways in GANs (Hinz, et al., 2018), which is not covered in the taxonomical survey (Agnese et al., 2020). To generate specified object at the specified locations in an image, an object pathway is added to the generator and discriminator of the GAN. 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 use multiple discriminators to produce images with high resolution. This work tries out an efficient deep learning model that demonstrates the capability to generate multiple objects in an image with in-depth details that are seemingly plausible.

## **State of the Art**

A significant breakthrough in connecting visual images with natural language descriptions is CLIP (Radford et al., 2021). The images generated using CLIP have an artistic value (Smith, 2021). CLIP stands for Contrastive Language–Image Pre-training. Using two encoders, one for text and the other for images, the framework can zero-shot learn to associate images with textual descriptions. The idea is to maximize the similarity between the embeddings generated by the two encoders. However, optimizing the similarity between the embeddings is formidable and off-the-shelf packages are not good at it (Liu, et al., 2021). The authors use a pipeline and newer optimization techniques available in the literature to implement zero-shot text-to-image generation without needing extensive training infrastructure. CLIP has been used for zero-shot generation of text from images and vice versa (Galatolo, et al., 2021). A genetic algorithm has been used in the work to maximize the similarity of the embeddings generated by the CLIP framework. The DALL.E system from OpenAI (Ramesh, et al., 2021) is an exhaustive work using an autoregressive transformer with 12 billion parameters that is trained on 250 million pairs of images and the corresponding text description. The images generated can be tuned to match a user-given textual description.

## **Datasets**

CUB (Wah et al., 2011), Oxford, and COCO (Lin et al., 2014) are a few of the datasets used for image synthesis with GAN models. The CUB dataset contains 200 bird images along with text descriptions. The COCO (Common Objects in Context) dataset contains 328k images with 91 different object types. The objects in each image are at unique locations and are annotated in multiple ways. The annotations help with tasks such as object detection, keypoint detection, and semantic segmentation. The images come with five captions each. The annotations for object detection are in the form of bounding boxes. The bounding boxes and captions make the COCO dataset ideal for training the GAN models in this project.

The Oxford dataset contains 102 categories of flowers with 40-258 images each and text descriptions. Additionally, the COCO dataset contains images with multiple objects whereas the CUB and Oxford datasets have images with only a single main object with its text descriptions.

### Evaluation Metrics

GAN models are assessed based on certain evaluation metrics, each of which has its own advantages and disadvantages (Borji A., 2022) (Borji A., 2019). Out of the many evaluation metrics used in the literature, Fréchet Inception Distance (FID) (Heusel, Ramsauer, Unterthiner, Nessler, & Hochreiter, 2017) is commonly used as an evaluation metric, which compares the images generated with the real images from the data distribution. A low FID (Zhu, Pan, Chen, & Yang, 2019) 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) (Salimans et al., 2016) calculate the randomness of the 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 the same data distribution and high randomness means that the images generated are diverse. The performance of different GAN models used for Text-to- Image synthesis for datasets discussed above along with the evaluation metrics is reported in Table 1.

Table 1. Model comparison sorted by SOA-C

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

### Image Synthesis from Text

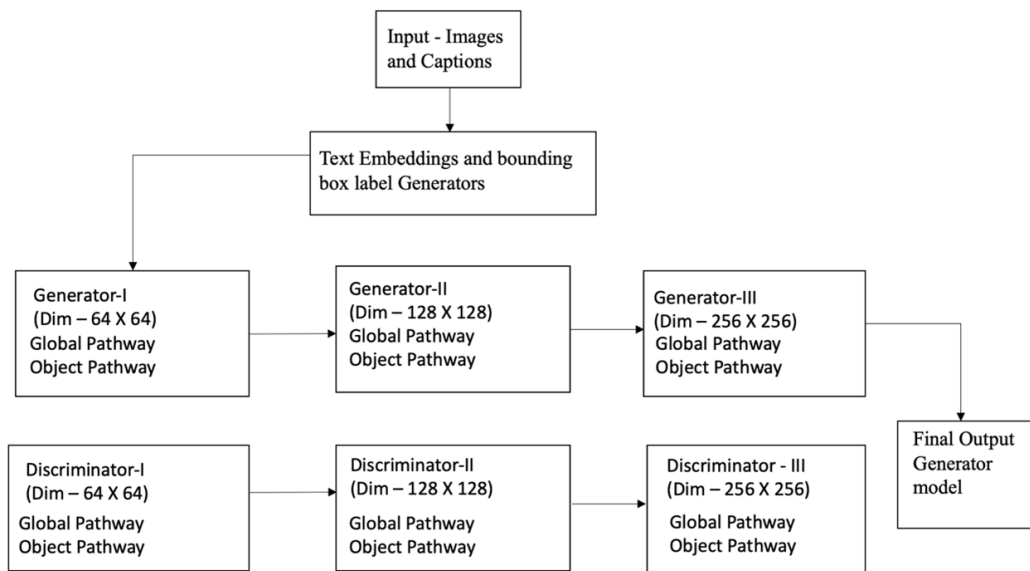
The primary objective of this work is to develop a GAN model which can generate images containing multiple objects conditioned on text descriptions. The model is aimed to demonstrate semantic intelligence to decipher what part of the text should be used to generate the corresponding parts of images. The model aims to develop realistic images that are closely related to the description. . In Figure 1, random Gaussian noise is converted into photos that look genuine. Similarly, for this project, text description that can be converted into vectors, which embed the meaning of the words can be converted into images that capture the meaning of the text description. The multi-modal transformation in Figure 1 was from meaningless numbers from a random Gaussian distribution to photo-realistic images. The multi-modal transformation we are targeting for this project is from meaningful text to images that represent the meaning of the text.

## Architecture

At a high level, the architected system transforms one kind of numbers to another kind of numbers. Recurrent Neural Networks (RNN) used in the transformer architecture are used to convert the given text description into vectors of numbers that capture the word meanings. Images are essentially numbers – a set of pixel values. Convolutional Neural Networks (CNN) are used to extract the features of the images using a mathematical operation called convolution on the pixel values. The GAN architecture used for this project essentially converts the vectors of numbers representing the given text description into the vectors of numbers that represent the generated image. At a high level, this is primarily done by backpropagating the loss to adjust the weights of the neural networks. The numerical transformations need to go through a stack of GANs that pay attention to different parts of the text description, while improving the resolution of the image. The attention to different words in the text description is staged through a series of GANs.

The architecture of our work is predominantly based on the object pathway GAN (Hinz et al., 2018) and AttnGAN (Xu et al., 2018). In general, like mentioned earlier, GANs have two different networks: a generator that eventually generates images similar to the one given in input samples and a discriminator to differentiate between the real and generated images. AttnGAN architecture (Xu, et al., 2018) is used as a base architecture for our project. It works based on conditional GAN where attention and additional information are conditioned on the generator and three discriminators. Attention is giving importance to different words from the caption to the specific regions of the image. The architecture is illustrated in Figure 2. Captions are converted into word embeddings and bounding boxes are converted into labels. For instance, if the bounding box is around a dog in the image, the label for the bounding box is “dog,” which is then converted into a number by using one-hot encoding. The one-hot encoding of the label and the word embeddings for the caption are then combined using a fully connected layer to produce a label. These labels are passed to the first generator, which works with its corresponding discriminator to produce the inputs to the second generator, and so on.

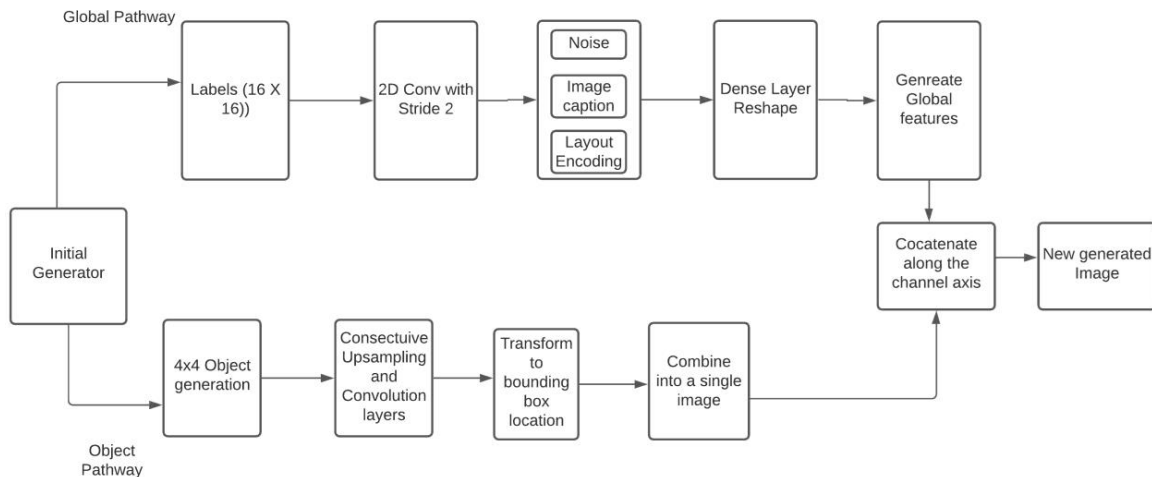
Figure 2. High-Level Architecture



## Generator Network

Generators perform up-sampling of the latent space, in this case, the word embeddings of the text description to data in real space, in this case images that capture the intent of the text description. The generator network consists of two different pathways – the global pathway and the object pathway. The generator architecture is illustrated in Figure 3. As their names indicate, the object pathway generates individual objects based on the labels from the pre-trained encoder network. The global pathway generates the “global” scenario - other surrounding common areas of the image which is also conditioned on the text input (Hinz et al., 2018).

Figure 3. Low-level Generator Architecture



Objects that are associated with the caption are represented in a 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 features at specific locations 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 the first object pathway in the generator to extract the features and transform them around the bounding box location using a spatial transformer network (STN). Global pathways generate the features by receiving the individual object’s location and size and applying a convolutional layer 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 an up-sampling technique.

## Discriminator Network

The discriminator can be thought of as a teacher and the generator, its student who learns from the feedback that the former provides. If the discriminator is trained incorrectly, the generator will be incorrect as well. If the discriminator is trained on incorrect text to image mappings, the generator produces

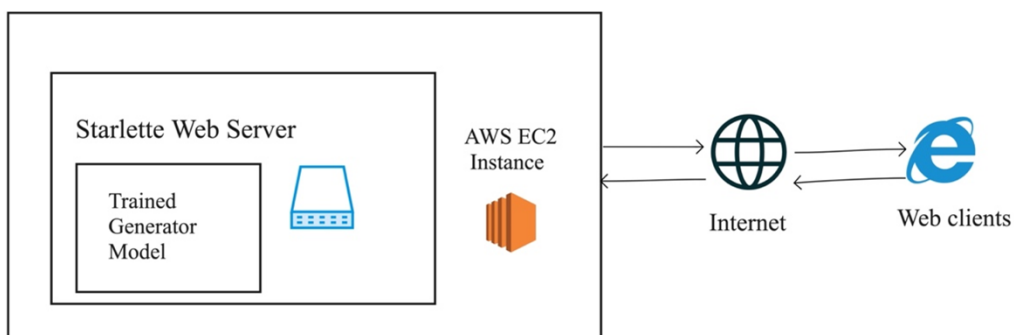


incorrect images for the given text descriptions as well. It is therefore logical that the discriminator also contains the global pathway and object pathway. The object pathway extracts the fine-grained details of individual objects using a Spatial Transformer Network (STN). In the global pathway, an input image is processed by applying a 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. It consists of a 2D convolution layer with stride 2, to extract the common global features of the image. Conditional and unconditional losses are calculated for both the 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.

## Web Application Architecture and Deployment

For better usability and comprehensive implementation, we deployed the entire application to be accessible from a web browser. Figure 4 shows the deployment schematic. In a browser client, when the deployment URL is requested, an HTTP request is sent to the web server, where the trained model is deployed. The web server sends back the responses to the web browser, which will be handled by the web browser and display the requested web content. We used the Starlette web application framework for designing the web application for our project. This follows a micro-framework pattern. It is a lightweight WSGI web application framework that provides rapid development, reliability, and scalability to build complex applications. It provides server-side rendering which enables a faster response from the web application when a request is made.

*Figure 4. Deployment Diagram*



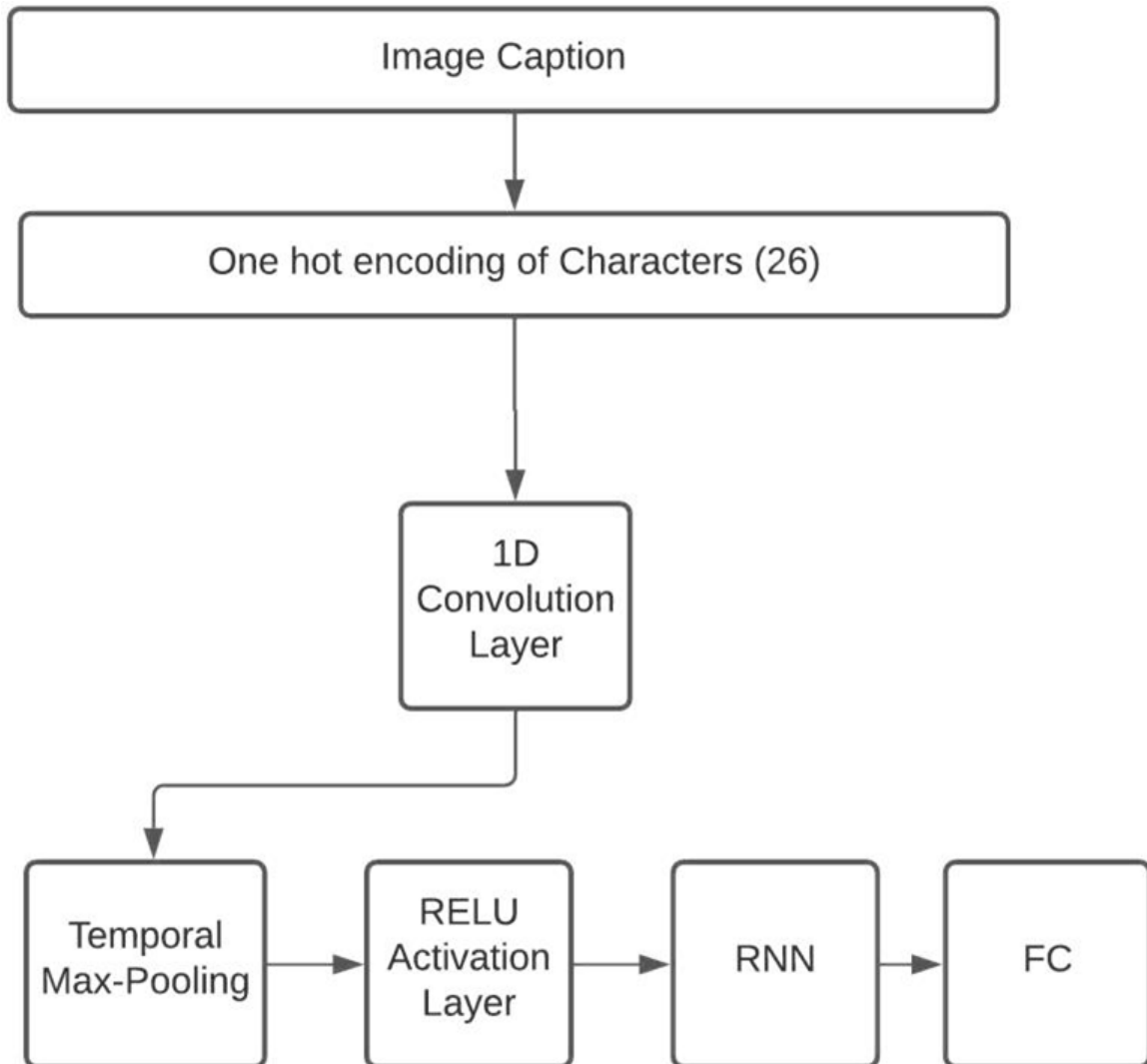
## Natural Language and Image Correlation

We use an encoder model for representing fine-grained visual descriptions. The fine-grained visual description denotes how well we can model the words in the caption that describes a particular section of the image. Each part of the caption text contains information about some parts of an image which is why it is crucial to model this before we can construct a generator and discriminator model. Text-based CNN is very similar to the standard image-based CNN. The difference is that the image width is 1 pixel (for a character) and the channel width is the number of unique characters (26 in the case of English). This

## Reconnoitering Generative Deep Learning Through Image Generation From Text

is followed by a temporal max-pooling layer and a RELU activation layer. Text inputs are characterized by temporal dependencies which are not always captured by CNN models. To address this problem, the hidden layers of the network are constructed with CNN and the top layers are designed to be an RNN to capture the temporal structures in text data. The architecture is illustrated in Figure 5.

Figure 5. Char-CNN-RNN Encoder



## **CLIENT IMPLEMENTATION**

### **Starlette**

We implemented the Starlette framework for loading the saved model and rendering the output of the model to the users. Starlette is useful as it deals with a synced service. We developed a web application that deploys the GAN model and servers the request for generating the images. We made the model accessible to the user in the browser by utilizing the capabilities of the deep learning model, by abstracting the model itself.

### **HTML, CSS, and Bootstrap**

The HTML web page contains a text box for getting the input as a textual description from the user. On clicking the submit button, the input from the user is sent to the GAN model for generating the images. A list of images is displayed to the user as the result. Cascading Style Sheets are used for stylizing the web page by providing the color of the form and setting the size of the font, text box, etc., Bootstrap framework is implemented to make the web page responsive and make the web page uniform with the existing template.

## **MIDDLE-TIER IMPLEMENTATION**

### **PyTorch**

The entire GAN model is developed using PyTorch. This open-source package provides various utilities for faster development of code and gives fine-grained model implementation functional APIs. We utilized libraries such as transforms for image transformations, “auto grad” for auto differentiations, “nn” for creating the neural networks, “optim” for optimization of training of the models.

### **Shell Scripting**

We developed a shell script for both model training and evaluation. These scripts are the entry point for code invocation. They also support program arguments to the main python file. Some of the model hyperparameters during training and evaluation are defined inside the respective YAML files. The scripts are also responsible for setting up the necessary system path variables. For example, the CUDA\_VISIBLE\_DEVICES needs to consist of a list of available GPU device indices. These indices will then be referenced by CUDA drivers.

### **Model Training and Objective function**

The model training is initiated from a shell script. The model is trained on a single Nvidia Tesla V100 machine. The objective function of a GAN model is typically a min-max game between the generator and discriminator. However, one difference here is that it is not vanilla GAN but a conditional GAN, conditioned on captions. AttnGAN is used as a baseline model. AttnGAN has achieved significant results

in conditional-GAN domain datasets like CUB. The reason being the generator pays more attention to the keywords that describe the image which is spatially and temporally important.

To make the model robust, the discriminator is fed with multiple combinations of text and image pairs. This includes (real text, real image), (fake text, real image), (fake text, fake image), and (real text, fake image). Equation (1) is the overall objective function of the model (Hinz et al., 2018).

$$\min_G \max_D V(D, G) = \mathbb{E}_{(x,c) \sim p_{data}} [\log D(x, c)] + \mathbb{E}_{z \sim p(z), c \sim p_{data}} \left[ \log \left( 1 - D(G(z, c), c) \right) \right]$$

## IMPLEMENTATION

### Natural Language and Image Correlation

We used a pre-trained text encoder model for this task. The pre-trained model is trained on CUB-200 and Oxford Flower dataset. The model is a Char-CNN-RNN encoder model, which is why it captures fine-grained details of the dependency relationship between the text and image. The pre-trained model is trained for 100 epochs after which the model's accuracy has not improved substantially.

### Generator Network

The Generator model is constructed using multiple python classes. Figure 6 shows the class-level relationships. The G\_NET class is the main class for the generator. It consists of several attributes like CA\_NET, h\_net1, h\_net2, and h\_net3. CA\_NET is the comprehensive Attention Network. H\_net1 is the object of the INIT\_STAGE\_G class which consists of a bounding box net as its first layer. It also consists of several subsequent up-sampling layers. NEXT\_STAGE\_G class is used for h\_net2 and h\_net3 objects. We have also used Gated Linear Unit (GLU) as one of the activation layers in CA\_NET to selectively train the model on what part of data to be given attention to.

The implementation of the Global and Object Specific Pathway is similar to the one in the literature (Hinz, et al., 2018).

### Discriminator Network

Like the generator, the discriminator is also implemented using a set of python classes. The class diagram of the discriminator is shown in Figure 7. The main class is D\_GET\_LOGITS which consists of objects like COND\_DNET and UNCOND\_DNET. These objects are of type D\_NET64 (64x64), D\_NET128 (128x128) and D\_NET256 (256x256). The D\_NET128 and D\_NET\_256 reuse the class attributes of D\_NET64. In class D\_NET64, we have accommodated the changes for the discriminator network from the original implementation. Spectral normalization is a type of weight normalization helpful in cases where batch normalization does not help much in model performance. Another reason is while training using GPUs, due to low GPU memory, many times we are forced to give lesser batch size. Lesser batch size in training causes random movement in gradient descent and takes a longer time to converge.

## Reconnoitering Generative Deep Learning Through Image Generation From Text

Hence, the 2D batch normalization layer is replaced by Spectral normalization. The implementation of the Global and Object Specific Pathway for the discriminator is similar to the one in the literature (Hinz, et al., 2018).

Figure 6. Generator Class Diagram

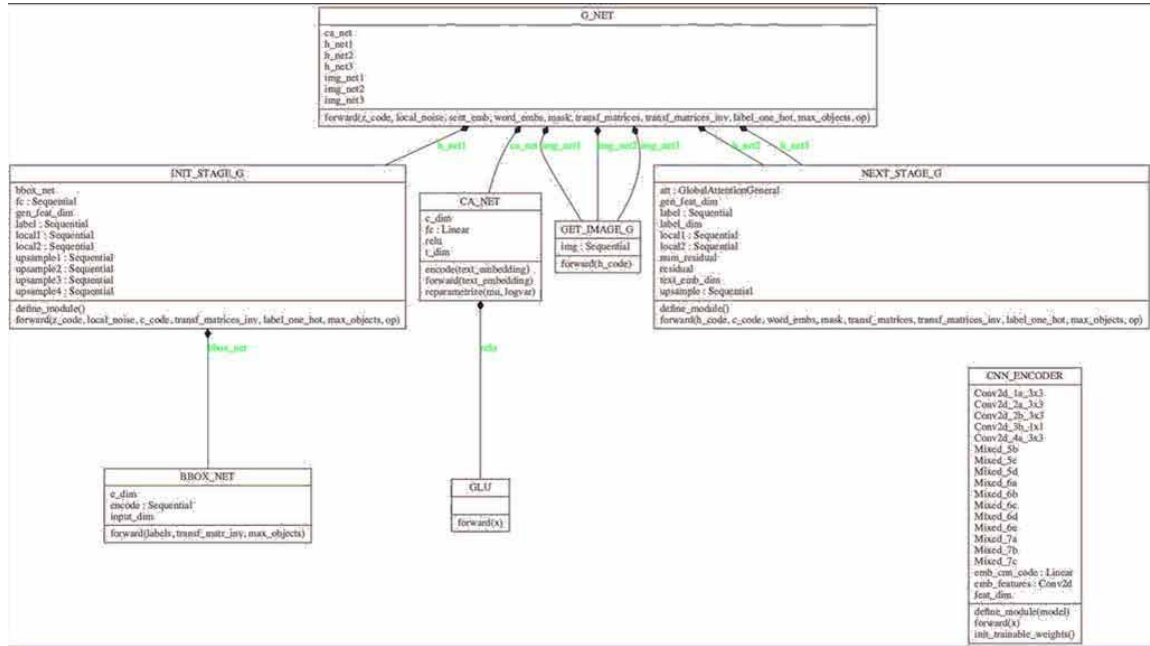
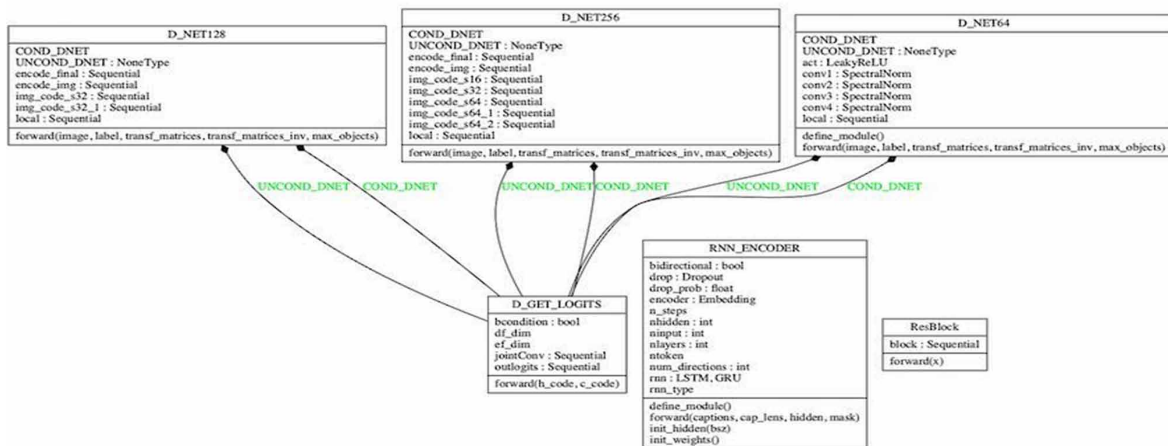


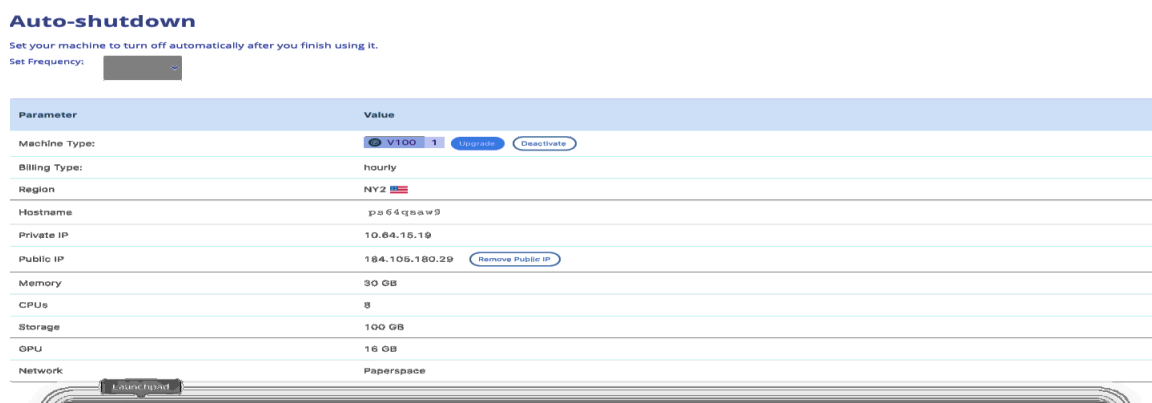
Figure 7. Discriminator Class Diagram



## Model Training

The model is trained in Paperspace (<https://www.paperspace.com/>) cloud provider. We provisioned a machine of type Nvidia Tesla V100. We also used a 100GB SSD hard drive to store the MS-COCO dataset. The SSD drive helps in I/O performance. We have also disabled the Optimized data loading flag from the original implementation to keep the static batch size for every training iteration. Figure 8 shows the snapshot of the machine from Paperspace.

Figure 8. Paperspace NVIDIA Tesla V100 Instance



## MODEL EXPERIMENTS

### Machine Learning Operations with MLflow

We used MLflow for ease of managing our entire Machine Learning project's lifecycle. As per the official website of MLflow, it is an open-source platform for machine learning lifecycle management. It offers four components - MLflow tracking, projects, models, and registry. We leveraged MLflow's tracking component for tracking/logging our GAN model's Generator loss and Discriminator loss. We created a script for auto-logging of the required parameters (batch size, epochs, etc.) for our model. In addition to that, we also tracked metrics such as loss for our generator and discriminator network. Figure 9, 10, and 11 show various screenshots from MLflow.

### Performance and Benchmarks

#### Inception Score (IS) and Fréchet Inception Distance (FID)

For performance metrics and evaluation, we have used the Fréchet Inception Distance score (FID) (Heusel, Ramsauer, Unterthiner, Nessler, & Hochreiter, 2017). FID score takes the feature vector of both real and generated images and calculates the distance between them. The lower the score, the more the similarity between the real and generated/synthetic. That is, FID, evaluates the quality of images generated by the

## Reconnoitering Generative Deep Learning Through Image Generation From Text

Generator network and correlates with higher-quality images. The real and generated images are said to be similar, if the FID score is a perfect score of 0.0, indicating identical real and generated images.

Figure 9. Landing page of Mlflow experiments

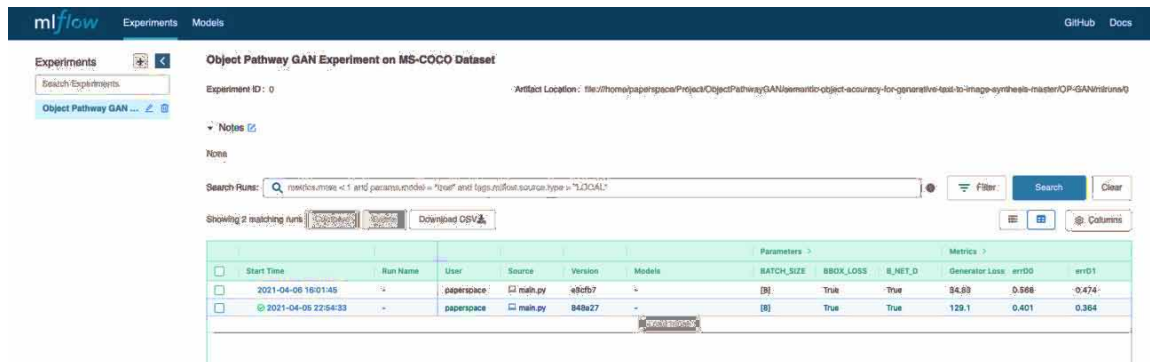
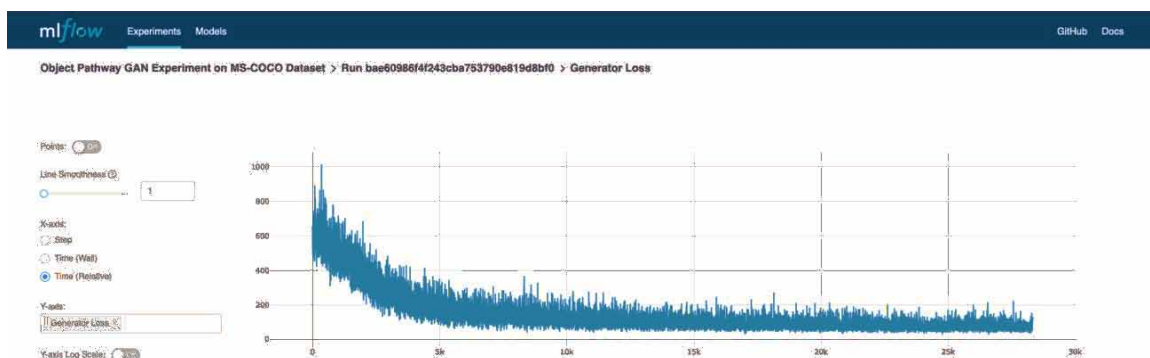


Figure 10. Experiment Parameter



Figure 11. Visualization of Generator loss over time



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

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

In addition to FID, we have also used the Inception Score (IS) (Salimans et al., 2016), as a metric for evaluating performance in terms of the diversity of the images generated and their quality. The inception score specifically evaluates the quality of the generated/synthetic images generated by the generator network of the GAN model. The inception score captures the image quality and image diversity of the generated / synthetic images. In other words, the Inception score captures, how the images look in terms of a specific object as well as the wide range of objects which are generated in the generated images. The range of the inception score lies between 0.0 to the number of classes supported in the model.

### Semantic Object Accuracy (SOA)

SOA (Semantic Object accuracy) is a novel evaluation metric that will evaluate the generated images from the OP-GAN model, given textual description. SOA evaluates if the object in the generated images matches the image caption. For instance, given the caption “A person driving a car” SOA will evaluate if the generated image consists of objects like a person and car. SOA outperforms the two most common evaluation methods: Inception Score (IS) and Fréchet Inception Distance (FID) because they do not take image captions into account when evaluating images.

In the MS-COCO data set, all the text descriptions in the validation set are filtered out based on the keywords which are specified in the labels. There are 80 labels available, and for each label, all the captions are discovered, and three images are generated for each caption. The object detector that we used is the YOLOv3 network, pre-trained on the COCO dataset. It is used to evaluate the given objects on each of the generated images. The process can be simplified using two approaches which are class average SOA-C and image average SOA-I (Hinz et al., 2020). SOA-C gives us the average number of images per class that the given object is detected and SOA-I gives us the average of the number of images in which the desired object was detected. The formula for SOA-I and SOA-C is given below (Hinz et al., 2020),

$$SOA - C = \frac{1}{|C|} \sum_{c \in C} \frac{1}{|I_c|} \sum_{i_c \in I_c} YOLOv3(i_c)$$

$$SOA - I = \frac{1}{\sum_{c \in C} |I_c|} \sum_{c \in C} \sum_{i_c \in I_c} YOLOv3(i_c)$$



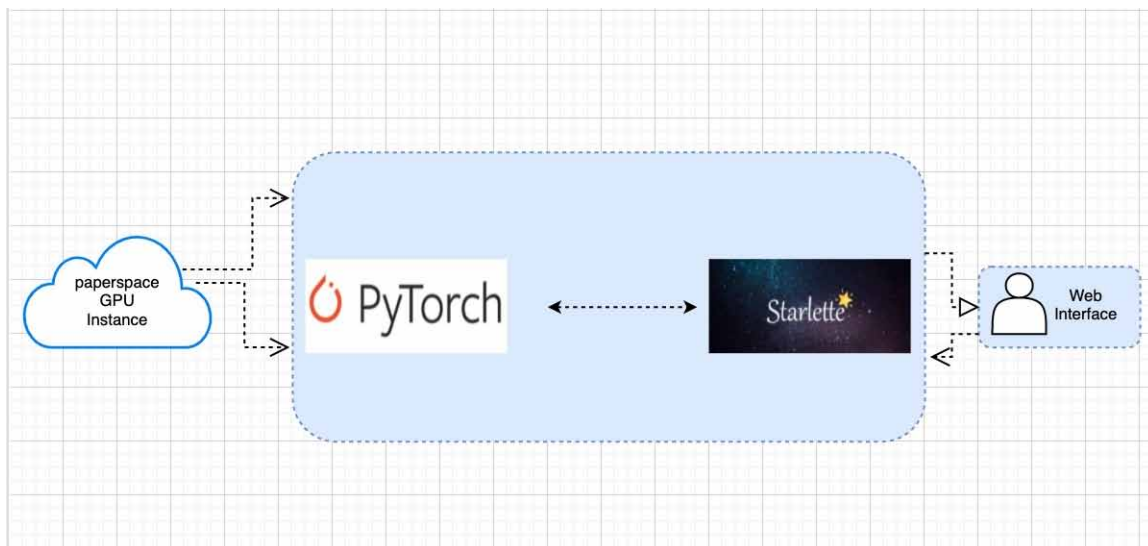
Table 3. Oxford dataset by Inception Score (IS)

Rank	Model	FID	IS
1	StackGAN-V2	48.68	3.26
2	StackGAN-V1	55.28	3.20
3	StackGAN		3.2

## Deployment, Operations, Maintenance

To serve our model on the web, we used the Starlette ASGI web framework. When the end-user clicks on “generate images”, our model serves the request by using our trained GAN model. This process of generating the images is done by passing the text input through the text encoder. This text encoder extracts the semantic features. The saved weights of our model are used to serve the user request and based on these weights and the semantic meaning of text describing the images is generated. Figure 12 below illustrates the deployment architecture.

Figure 12. Deployment Architecture



## CONCLUSION

The area of generative deep learning is vast and expanding. This chapter gave a focused and comprehensive view of the various aspects, including the deployment of generative deep learning for creating synthetic images from descriptions in natural language. We evaluated our work using multiple relevant evaluation metrics. The task has plenty of potentials and is still evolving. From our assessment, currently, the textual descriptions from which images can be generated are limited by their length and scope. With more powerful models trained on humongous datasets on the horizon, a future direction can be to gener-

ate more extensive images from longer texts. Such a possibility opens up a plethora of applications such as illustrating children's story books, generating animated movies from screenplay, and adding figures to research manuscripts. Editing software such as Microsoft Word may be able to provide these services as a value-add, similar to how designer ideas are incorporated into Microsoft powerpoint.

## **ACKNOWLEDGMENT**

The authors are grateful to Sivaranjani Kumar, Akshaya Nagarajan, and Pooja Patil for their contribution to this project.

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors, but was supported by San Jose State University.

## **REFERENCES**

- Agnese, J., Herrera, J., Tao, H., & Zhu, X. (2020). A survey and taxonomy of adversarial neural networks for text-to-image synthesis. *Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery*, 10(4), e1345. doi:10.1002/widm.1345
- Borji, A. (2019). Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, 179, 41–65. doi:10.1016/j.cviu.2018.10.009
- Borji, A. (2022). Pros and cons of GAN evaluation measures: New developments. *Computer Vision and Image Understanding*, 215, 103329. doi:10.1016/j.cviu.2021.103329
- Galatolo, F., Cimino, M., & Vaglini, G. (2021) Generating Images from Caption and Vice Versa via CLIP-Guided Generative Latent Space Search. *International Conference on Image Processing and Vision Engineering*. 10.5220/0010503701660174
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in Neural Information Processing Systems*, 30.
- Hinz, T., Heinrich, S., & Wermter, S. (2018, September). Generating Multiple Objects at Spatially Distinct Locations. *International Conference on Learning Representations*.
- Hinz, T., Heinrich, S., & Wermter, S. (2020). Semantic object accuracy for generative text-to-image synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. PMID:32877332
- Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., & Aila, T. (2020). Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 8110-8119). 10.1109/CVPR42600.2020.00813
- Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740-755). Springer.

- Liu, X., Gong, C., Wu, L., Zhang, S., Su, H., & Liu, Q. (2021). *Fusedream: Training-free text-to-image generation with improved clip+ gan space optimization*. arXiv preprint arXiv:2112.01573.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning* (pp. 8748-8763). PMLR.
- Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., ... Sutskever, I. (2021, July). Zero-shot text-to-image generation. In *International Conference on Machine Learning* (pp. 8821-8831). PMLR.
- Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. (2016). Generative adversarial text to image synthesis. *International conference on machine learning* (pp. 1060–1069). Academic Press.
- Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. *Advances in Neural Information Processing Systems*, 29.
- Smith, A., & Colton, S. (2021). Clip-guided gan image generation: An artistic exploration. *Evo, 2021*, 17.
- Wah, C., Branson, S., Welinder, P., Perona, P., & Belongie, S. (2011). *The Caltech-UCSD Birds-200-2011 dataset*. Academic Press.
- Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., & He, X. (2018). AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1316–1324. 10.1109/CVPR.2018.00143
- Zhang, G., Tu, E., & Cui, D. (2017). Stable and improved generative adversarial nets (GANS): a constructive survey. *2017 IEEE International Conference on Image Processing (ICIP)*, 1871–1875. 10.1109/ICIP.2017.8296606
- Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., & Metaxas, D. N. (2017). StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks. *Proceedings of the IEEE international conference on computer vision*, 5907–5915. 10.1109/ICCV.2017.629
- Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., & Metaxas, D. N. (2018). StackGAN++: Realistic image synthesis with stacked generative adversarial networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8), 1947–1962. doi:10.1109/TPAMI.2018.2856256 PMID:30010548
- Zhu, M., Pan, P., Chen, W., & Yang, Y. (2019). Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5802–5810. 10.1109/CVPR.2019.00595

## **KEY TERMS AND DEFINITIONS**

**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.

## ***Reconnoitering Generative Deep Learning Through Image Generation From Text***

**CUB:** Caltech-UCSD Birds. An image dataset of two hundred bird 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.