ATTENTION ON ATTENTION FOR TEXT TO IMAGE SYNTHESIS USING MODE-SEEKING LOSS FUNCTION

Naitik Bhise

A thesis
in
The Department
of
Engineering and Computer Science

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Computer Science
Concordia University
Montréal, Québec, Canada

February 2021
© Naitik Bhise, 2021

Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Naitik Bhise**

Entitled: **Attention on Attention for Text to Image Synthesis using Mode-seeking Loss Function**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

_____ Dr. Lata Narayanan

_____ Dr. Tristan Glatard

_____ Dr. Thomas Fevens

_____ Dr. Tien D. Bui

_____ Dr. Adam Krzyzak

Approved _____
Chair of Department- Dr. Lata Narayanan

_____ 20 _____ _____

Dr. Hovhannes Harutyunyan,
Graduate Program Director,
Faculty of Engineering and Computer Science

Abstract

Attention on Attention for Text to Image Synthesis using Mode-seeking Loss Function

Naitik Bhise

Text to Image Synthesis is a burgeoning field that has sprung up in the research community in the last few years. Generative Adversarial Networks form the basic component of modern research as many architectures are built around this particular type. This thesis is an attempt to develop a new technique and architecture to compete with the recent state-of-the-art models. The research around the text to image synthesis is conducted using two approaches by working with the modification of the loss and then the change in architecture. In the first approach, we sample two noise vectors to generate two different output images. The model is trained by a mode-seeking loss function which maximizes the ratio of the l2-norm of difference between the two images to the l2 norm of difference between noise vectors. The second approach deals with increasing the attention between the text and the image by introducing the Attention on Attention architecture in the AttnGAN network. We find that a combination of two approaches produces good quality images and better attended on their text descriptions. The metric scores of Frechet Inception Distance and Inception Scores are used to evaluate the results. The datasets used in this research are Microsoft COCO and CUB Birds. A comparison study of the obtained results with the past state-of-the-art models is conducted and presented in this thesis.

Acknowledgments

I would like to thank all who have helped me to this point in the Concordia program. First of all, I would like to thank Professor Dr. Tien D. Bui for giving me a chance to work under his supervision for the project and also for introducing me to this great domain of text to image generation. Professor Bui inspired me to explore many topics in the Deep Learning field and understand its research in greater detail. I have my sincere gratitude towards him for giving me the autonomy of research and financial support for the research. I can't imagine living in Montreal without financial assistance. I really appreciate his support during the period of my master's study. Secondly, I would like to thank all my lab-mates and my group mates who have been there at every point of struggle and helped me navigate through the research as well as logistic issues. Next, I would like to thank the course instructors at MILA where I completed the Deep Learning course and Reinforcement Learning course. Last but not the least, I want to thank my parents, for their selfless love and continuous encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

CHAPTER 1

INTRODUCTION

Text to Image Synthesis is a burgeoning field in recent years with the use of recent GAN architectures and LSTM for generating good quality images from text descriptions. Our research focuses on some of the previous architectures and constructs a new architecture with a modified loss function. This chapter provides a brief introduction to our research work. In the first part, the report will discuss the work on applying a mode-seeking function [11] to existing text to image synthesis model DM-GAN [8]. Then, research on modifying the AttnGAN [7] architecture will be described in detail. I will also give an outline of this thesis and future work in this field of research.

1.1 Text to Image Synthesis

Images come with captions all the time, but there are times while reading a caption strikes our imagination, and an abstract scenario is generated. Text to Image Synthesis field deals with the latter part where a deep learning network is used for generating an image aligned with the caption details. This research field utilizes a combination of two research fields in Deep Learning [13] - Natural Language Processing and Computer Vision. Natural Language Processing works with the text analysis by converting raw text to word vectors and embeddings while Computer Vision transforms them to clear images.

There have been past strides in this field where the work was done initially in the other direction for image captioning tasks. Our research is motivated by the same architectures. Image Captioning [14, 15] deals with transforming RGB images into image features which are then used in caption generation tasks. There are few techniques employed in the image captioning architecture that are used as well for the text to image synthesis. In this thesis , we will introduce some of the important past architectures and their significance before arriving at the construction of our research models.

Deep Learning models understand the mathematical relation between input and output [13]. The loss function is responsible for the effective training of the network or giving a direction for the back-propagation to find a suitable model. Back-propagation is an iterative gradient-based

weight learning procedure aimed at minimizing the loss function. The capacity of a network refers to the range or scope of the types of functions that the model can approximate. Two approaches are employed in the whole research with one involving modification of the loss function and the other with the change of the architecture. Past architectures in Text to Image Synthesis are explained in our thesis with the links to their research and Github code.

Generative Adversarial Networks [16] are relatively new architecture in Deep learning. It consists of a Discriminator and a Generator. The generator generates fake images from a random uniform distribution while the discriminator tries to disqualify them by comparing them with real images. cGANs [17] are frequently used in Image to Image translation [18, 19, 20, 21, 22, 23], text to image synthesis [4, 5, 6, 7, 8, 11, 22] and semantic manipulation [24]. They take prior inputs for the generator to generate an image.

In a general text to image synthesis architecture, text embeddings are extracted from a sequential model and passed through a GAN network for image generation. The sequential model is generally a Bi-directional LSTM [25]. Single-stage methods [4, 26] were the first to use Generative Adversarial Networks [16] to generate images from the text descriptions. Initially used for small resolution images by the single-stage methods, this developed into multistage methods. Multistage methods [5, 7, 6, 8] appended more GAN networks to the single-stage to augment the image resolution as well as improve the quality of the images generated. Multistage architectures [5, 6, 7, 8, 4, 26] were used as a baseline for my research. The mode-seeking loss function [11] was used for the retraining of the DM-GAN [8] to generate new results.

1.2 Mode-Seeking Loss Function

Our first approach is focused on the modification of loss function for the best DM-GAN[8] architecture. The work starts from the introduction of the basic multimodality problem of the GAN and resulting issues. GANs have multimodal output but during training, some of the modes are suppressed relative to others, affecting the diversity of the output [27]. The mode-seeking loss function [11] finds a way to prevent the mode-collapse by increasing the number of output modes of the GAN. It samples two noise functions and maximizes the difference between the two resulting output images. This improves the diversity of the images generated by architecture by strengthening the output data distribution.

1.3 Refined Attention in AttnGAN

Attention network [28] is a deep learning architecture that tries to focus on specific parts of a text sentence or an image [26]. The attention network of the attnGAN [7] provides a way to find

the semantic relationship between the text description and the image features. Refined attention attempts to improve the attention on the image features by including the global sentence vector with the word embeddings [9]. A new method of Attention on Attention(AoA) is also used in the network. AoA [10] generates an information vector and an attention gate using the attention result and the attention query and adds another attention by applying the gate to the information and obtains the attended information.

1.4 Refined Attention with Mode-Seeking Function

In this small section within the Refined Attention, we describe the importance of the mode-seeking function and the difference it casts on the image quality as well as description clarity. First approach is considered as a baseline for the determination of the importance of the mode seeking function.

The results from the experiments are evaluated based on metric scores. The metrics used for the evaluation are the Frechet Inception Distance [29] and the Inception Score [27]. FID and IS are used frequently in the GAN image evaluation. Inception score is used for the evaluation of the quality and the diversity of the image generation while FID supplementarily has the benefit of matching it to the real data distribution.

The output images are displayed for the observation of its quality and correlation with the text. The images used with the attention are displayed for the application of the attention features and the different states of the images in an attention network. Our research attempts to understand the text to image synthesis work and tries to improve the quality and correlation of the images with the text through a series of experiments. The experiments can be broadly classified into loss based modification and architecture based modification. The datasets used for the research are the Caltech Birds dataset [2] and the Microsoft Coco dataset [3].

The thesis starts with the literature review which presents the basic understanding of the deep learning modules involved in the research and the previous research in the text to the image domain. The third chapter explains the first experiment with the implementation of the loss and the results extracted from the datasets. Chapter 4 describes the refined attention GAN module, and the last chapter contains the summary of the thesis.

CHAPTER 2

LITERATURE REVIEW

We review the basics of module components and past architectures used for the research. This chapter starts with an explanation of deep learning and neural networks. The second part of this chapter deals with the previous research in the Text to Image Synthesis domain. We also mention some more architectures relevant to our research.

2.1 Basic Modules

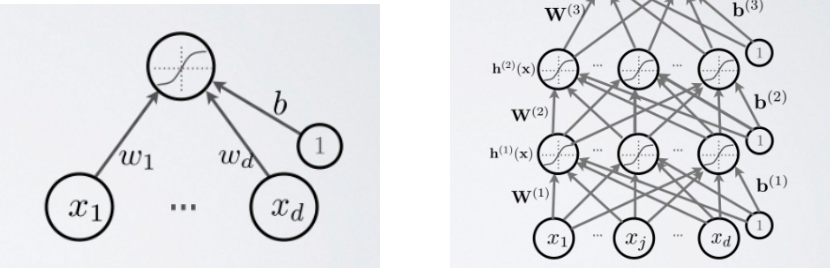
This section is an introduction to the basic Deep Learning modules used in the research.

2.1.1 Artificial Neural Network

Neuron is the basic building block of any neural network, and its function is similar to a neuron in the human brain. It takes an input and generates an output of either zero or one through an activation function. Here we use a basic neuron with a non-linear activation for the classification. Connection weights w_i are applied to each of the features and a bias b. Non-linear activation function to finally obtain an output d(x). Suppose x is a set of features vectors with a non-linear activation g, we get the result d expressed as :

$$d(x) = g(b + \sum_i w_i x_i) = g(b + w^T x) \tag{2.1}$$

Generally, we are calculating the conditional probability of the decision being a yes or $p(y = 1|x)$. Neural networks have neurons as their basic units. Artificial Neural Networks [30] are a combination of such structures. We have layers of multiple neurons densely packed between the input and the output. The neurons in the inner layers are called the hidden layer neurons. Each layer has a non-linear activation function. Artificial Neural networks are used for approximating any distribution by varying the number of layers and the number of neurons in each hidden layer.



(a) Neuron with its inputs and weights

(b) Artifical Neural Network containing layers of neurons

Figure 2.1: Basic Neuron and Neural Network. Source : Internet

The neuron weights or coefficients are the hyperparameters of the network. We can use different activation functions in the network . Examples of some non-linear activations are sigmoid activation, ReLU activation, leaky-ReLU and hyperbolic tan function. Different non-linear functions serve different purposes for a neural network. A neural network is trained by using backpropagation. The loss function for the classification task is taken according to the problem statement at hand. The hyperparameters of the neural network are updated at each training by differentiating the loss function concerning each hyperparameter.

Increasing the layers or number of neurons in a layer increases the capacity making it possible to interpolate over the data better. Increasing capacity can also lead to overfitting the training data. To avoid overfitting, we need to increase the amount of data or adjust the total number of hyperparameters of the network.

2.1.2 Convolutional Neural Networks

In neural networks, Convolutional neural networks [31] (ConvNets or CNNs) are one of the important models for image recognition and image classification. Objects detections, face recognition etc., are some of the areas where CNNs are widely used.

CNN takes an input image, processes it and classifies it under given categories (Eg., Dog, Cat, Tiger, Lion). The image is presented as an array of pixels and it depends on the image resolution. Dense Neural Networks cover the pixel-wise features of the image and generate the classification but with the augmentation in the resolution of the image, the pixels increase and thus our classification becomes computationally expensive. Convolutional Neural Nets make use of a kernel for capturing the local features of an image and the connectivity between the neighbouring regions. The kernel parameters are trained for the better capture of the local features.

The figure 2.2 explains the basic operation of a kernel on an input filter to generate an output

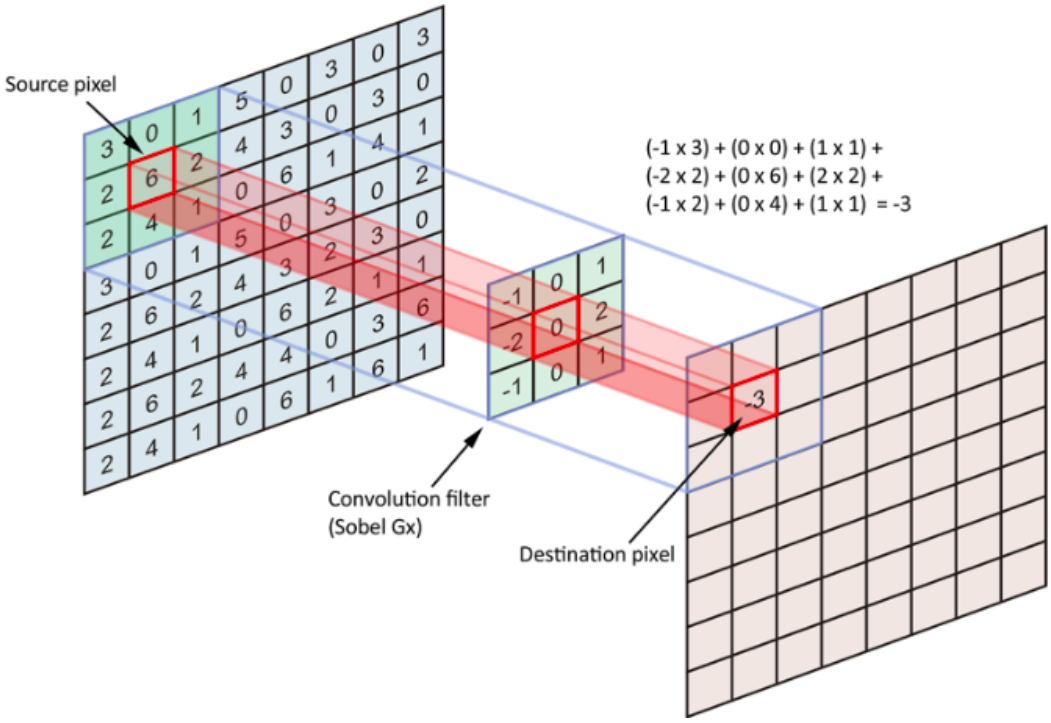


Figure 2.2: Convolutional Neural network makes use of a kernel function for capturing local features. Source :Internet

filter. The kernel matrix moves around the input vector to generate a single value by using convolution. The input can be supplemented with padding- a zero value layer around the source filter. The convolution stride can be adjusted according to the desired output filter resolution. The number of input and output filters can be adjusted as features of a network. A non-linear activation is applied at the output of the convolution to insert the non-linearity.

2.1.3 Recurrent Neural networks

Recurrent Neural Networks [32] or RNN as they are abbreviated, are a very important variant of neural networks heavily used in Natural Language Processing. In a general neural network, the input is processed through several layers and an output is produced, with an assumption that two successive inputs are independent of each other.

Sequential models are used to learn for generating the next information given the data available until then. Recurrent Neural Networks are called recurrent as they perform the same task for every element in the sequence and output elements are dependent on previous elements or states. The hidden vector stores the information of the past data and helps in determining the next output of

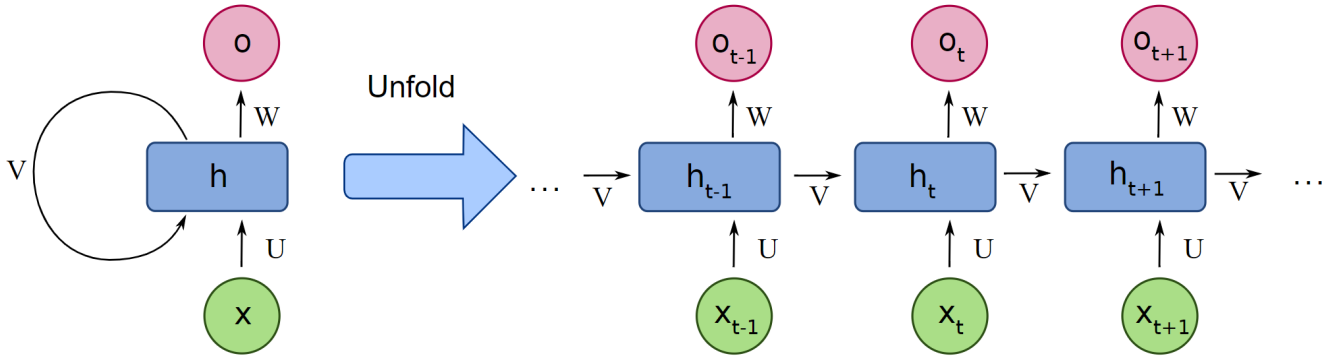


Figure 2.3: Recurrent Neural Network in time progress.
Source:Internet

the RNN. The next hidden vector is calculated by the expression:

$$h_t = tanh(b + Vh_{t-1} + Ux_t) \qquad o_t = Wh_t \tag{2.2}$$

where h_t and h_{t-1} corresponds to the current and previous hidden vectors respectively, w_t is the current input, b is the bias and U and W are the linear networks. We have to train the network over a sequence of data to train the hyper-parameters of the linear networks.

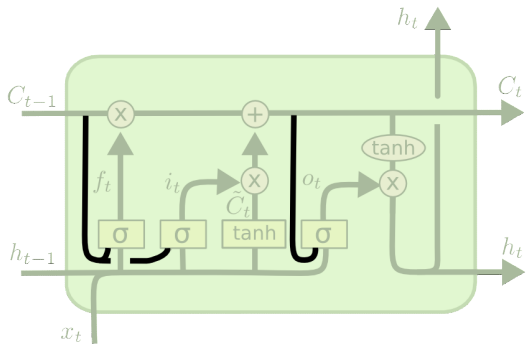
The RNN is trained through back-propagation through a time where the gradients concerning the linear weights are aggregated from the end to start. There are problems with the vanishing gradient [33] and exploding gradient [34] associated with the method. The exploding gradient is the issue of a huge gradient sum while low gradient sums to negligible values in vanishing gradients.

2.1.4 Long Short Term Networks

RNNs can depend on past information but it struggles to cope up with long-term dependencies. LSTMs [25] are explicitly designed for the long-term dependency problem. LSTMs behaviour is to remember information over a long time step, so it needs to know what to remember and what to forget.

The first step in an LSTM is to decide the information to throw away from the cell state. This decision is made by a sigmoid layer called the "forget gate layer." It looks at h_{t-1} and x_t, and outputs a number between 0 and 1 for each number in the cell state C_{t-1}. 0 represents the forget signal while 1 keeps the information.

The next step is to decide what new information to store in the cell state. This has two parts. First, a sigmoid layer called the "input gate layer" decides the values to update. Next, a Tanh layer



$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Figure 2.4: LSTM network with the calculation according to its activations [1] . Source : Internet

creates a vector of new candidate values, $\tilde{C}_{t-1}$, that could be added to the state.

The old state is multiplied by forget gate f_t, forgetting the things. Then we add the quantity $i_t\tilde{C}_{t-1}t$. A sigmoid layer decides the output of the network. Then, the Tanh layer updates the output with the candidate values.

The problems of the vanishing gradient of the RNN can be solved by the use of the LSTM where LSTM stores its information in the cell state from time to time during the training. It can store the necessary information for the long term.

2.1.5 Bi-Directional LSTM

Bidirectional recurrent neural networks(BRNN) [35] have a combination of two independent RNNs. The input sequence is fed in normal time order for one network, and in reverse time order for another. This structure allows the networks to have both backward and forward information about the sequence at every time step.

There are two streams of RNNs running parallel to each other- one in the forward direction where we feed the sentence and the other in the reverse direction where the sentence is reverse-fed. In the case of Bi-LSTM, these are two LSTM. The outputs of the two RNNs are combined using average, sum, concatenation or multiplication operation before passing through a softmax layer.

Bidirectional recurrent neural network (BRNN) [36] can be trained using all available input info in the past and future of a particular time-step.

2.1.6 Generative Adversarial Networks(GANs)

Generative Adversarial Networks [16] consist of two network modules of Generator and Discriminator. The generator generates fake data from latent vectors sampled from a random noise distribution to approximate a data distribution while the discriminator tries to distinguish between the generated samples and original data.

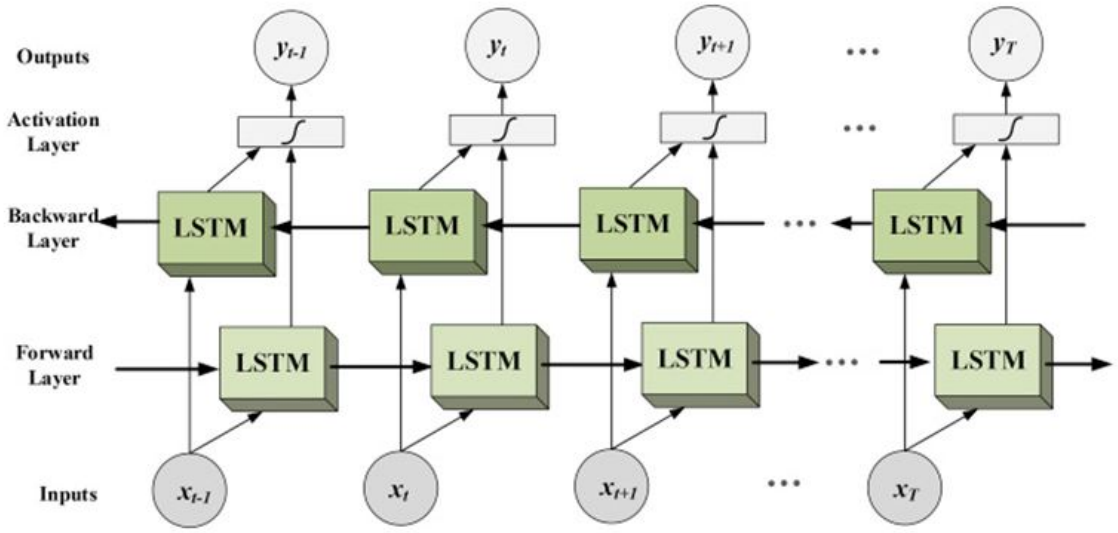


Figure 2.5: Unfolded Layers of a Bi-directional LSTM network.
Source : Internet

- A generative network G(.) that takes a random input z with density p_z and returns an output x_g = G(z) that should follow (after training) the targeted probability distribution
- A discriminative network D(.) that takes an input x that can be a "true" one (x_t, whose density is denoted p_t) or a "generated" one (x_g, whose density p_g is the density induced by the density p_z going through G) and that returns the probability D(x) of x to be a "true" data

At each iteration of the training process, the weights of the generative network are updated to increase the classification error (error gradient ascent over the generator's parameters) whereas the weights of the discriminative network are updated in order to decrease this error (error gradient descent over the discriminator's parameters).

Discriminator D gives a probability of whether the generated data from the generator G distribution is similar to the original data distribution. Their training follows a training scheme to minimize the function.

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[log(D(x))] + E_{z\sim p_z(z)}[log(1-D(G(z)))] \tag{2.3}$$

The input to a GAN is a uniform random variable that is being transformed into a distribution that is tried to compare with the targeted distribution by the discriminator.

The figure 2.6 shows the generation task of an MNIST dataset and its images. The random latent vector sampled from a random distribution is supplied to the generator for generating a

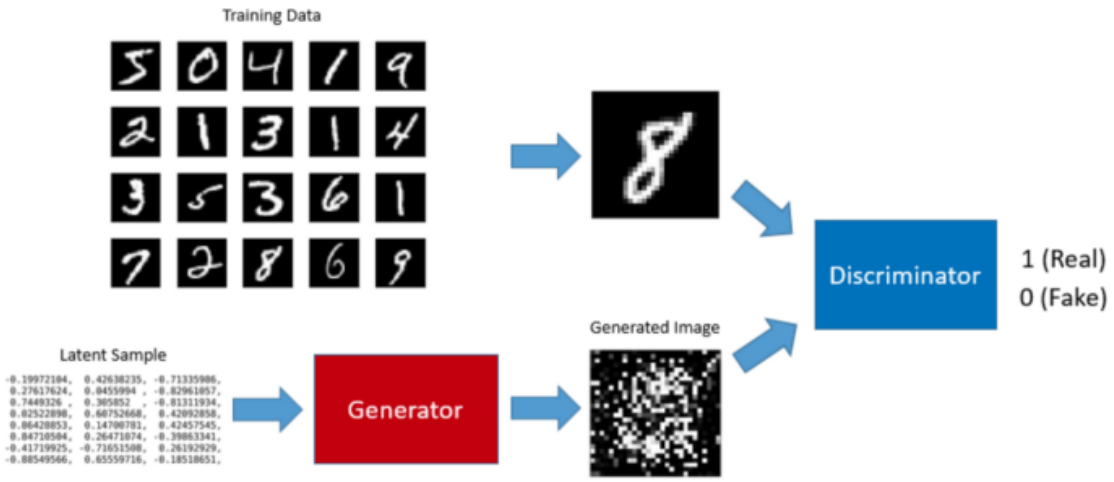


Figure 2.6: GAN model for the task of MNIST generation.
Source: Internet .

distribution. The generated image is compared with the data samples by the discriminator to learn the distribution. Thus, a random uniform function is transformed into distribution by GAN. GAN output has many modes making the system multi-modal. The modes can be trained according to the discriminator.

2.1.7 Conditional GAN

The conditional generative adversarial network [17], or cGAN for short, is a type of GAN that involves the conditional generation of images for a context by a generator model. Image generation can be conditioned on a class label, if available, allowing the targeted generalization of images of a given type of context. In the case of text to image synthesis, the generated images are conditioned on a text sentence. By conditioning, the generated images will be generated based on the context of condition provided at the input with the random function. For example, if the initial condition for the input is a class label of bird, the latent vector will only generate diverse bird images.

Additional information correlated with the input images, like class labels, can be used to improve the GAN. This improvement may come in the form of more stable training, faster training, and/or generated images that have better quality.

Class labels can also be used for the deliberate or targeted generation of images of a given type. A limitation of a GAN model is that it may generate a random image from the domain. There is a relationship between points in the latent space to the generated images, but this relationship is complex and hard to map. The conditioning is performed by feeding y into both the discriminator and generator as an additional input layer.cGANs map the distribution $P(X|y)$ where y is the

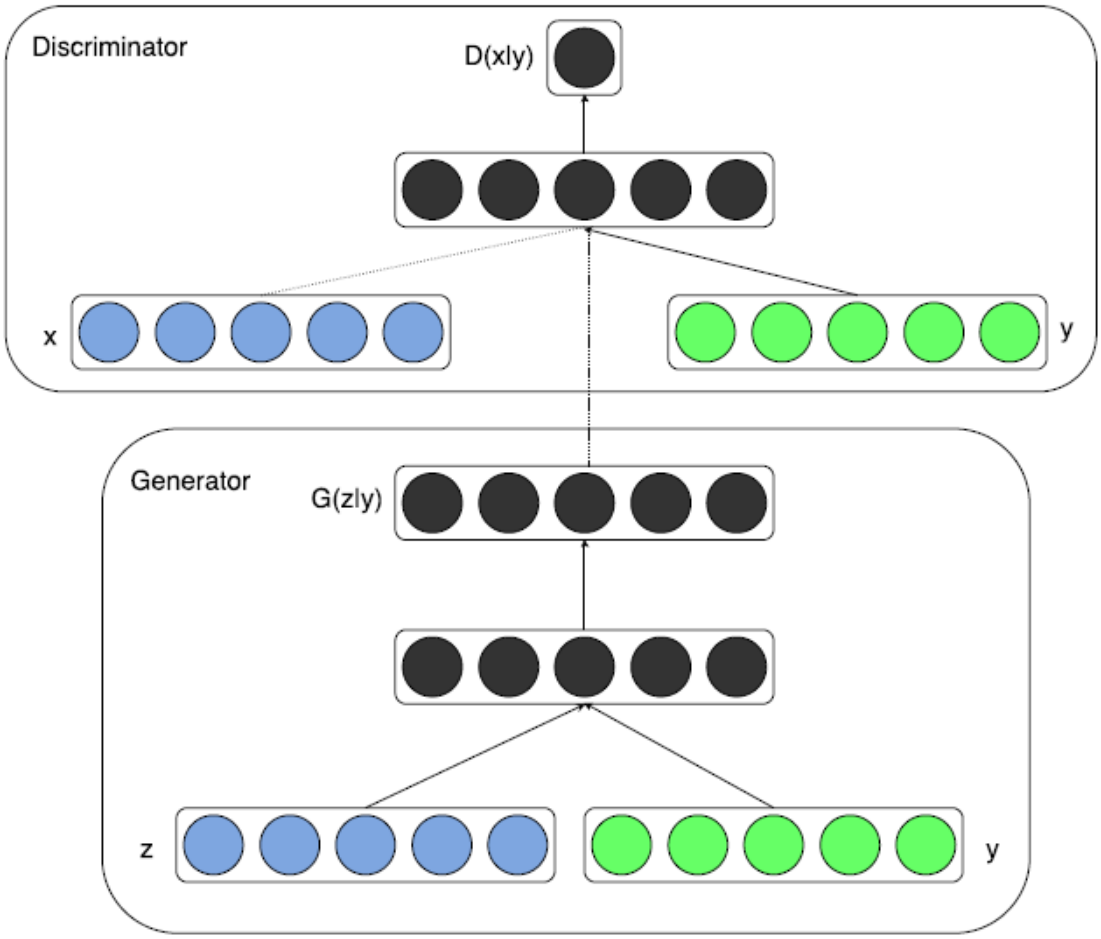


Figure 2.7: Idea of a cGAN where y is the context, z is the random vector and x is the true distribution sample. Generator generates a distribution G(z—y) conditioned on y while discriminator gives an output D(x—y) by conditioned on y. Source - Internet .

context provided at the input of the generator and discriminator. Overall, cGANs have an objective function.

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[log(D(x|y))] + E_{z\sim p_z(z)}[log(1-D(G(z|y)))] \tag{2.4}$$

2.1.8 Attention network

Suppose an RNN is trained on an image dataset that has to translate from one language to another. We know that different languages have different structures. The structure difference requires RNN to remember information for the long term. When a person translates, he keeps essential parts of the input sentences in his mind and applies them to the translation at different positions to form a precisely translated sentence. Thus, we need to give different weights or importances to different words present in the word embedding. Attention mechanism [28] provides the same purpose. Neural processes involving attention have been studied mainly in Computational Neuroscience [37].

An attention model is a method that takes n arguments $y_1,\ldots,\ y_n$ (in the preceding examples, the y_i would be the h_i), and a context c. It returns a vector z which is supposed to be the "summary" of the y_i, focusing on information linked to the context c. More formally, it returns the weighted arithmetic mean of the y_i, and the weights are chosen according to the relevance of each y_i given the context c. The attention network consists of three parts- key, query and the response. The Transformer network uses a particular form of attention called the "Scaled Dot-Product Attention" which is computed according to the following equation:

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d^K}})$$

We can see that the Query Q and key K of dimension d^K can enhance the elements of the value V. In some cases, the value V can be inserted in the place of the key K. The Query has the relevant information or the weightage given to every element of the V. We can apply attention to the outputs of the LSTM for better translation of sentences. Attention is also used in the vision problems for the identification of the significant parts of the image.

2.1.9 Memory networks

Memory networks [38, 39] are used for the storage and retrieval of information. There are 4 elements of a memory network that can be understood for the smooth operating of the memory systems.

- A memory, m, an indexed array of objects (e.g. vectors or arrays of strings).

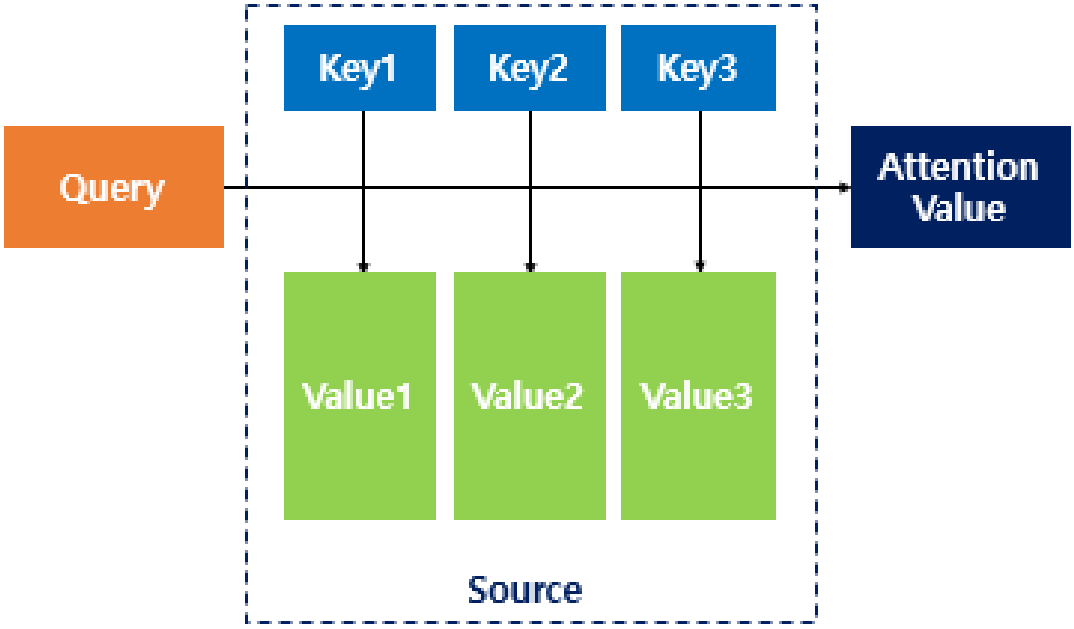


Figure 2.8: Attention mechanism with Key, Query and Value. Source: Internet

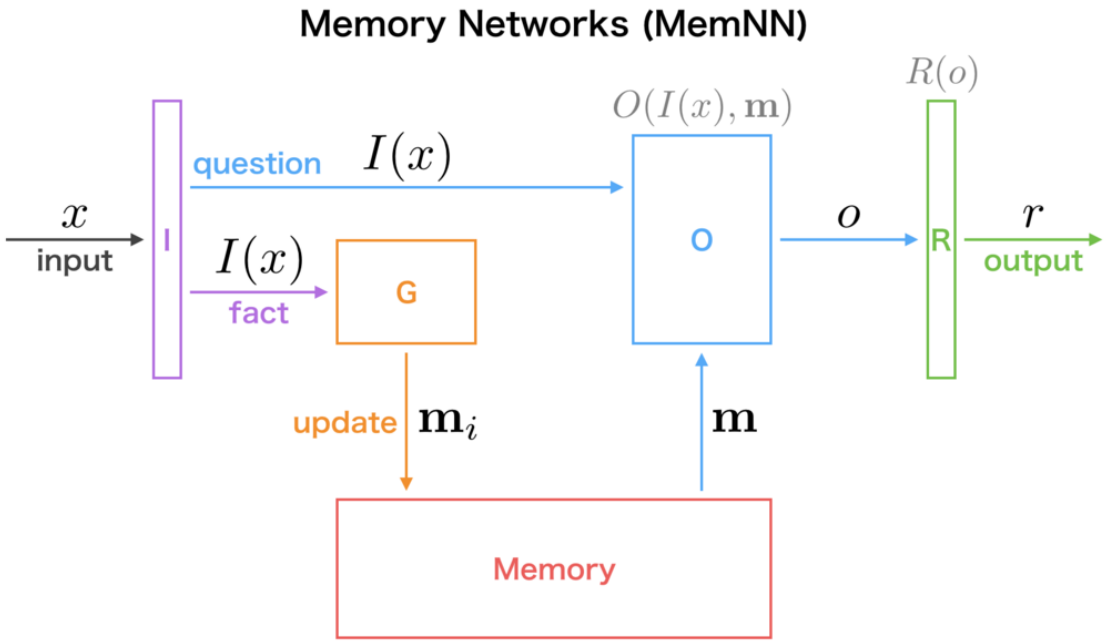


Figure 2.9: Memory architecture with layout of input processing through the layers. Source: Internet.

- An input feature map I, which converts the incoming input to the internal feature representation
- A generalization component G which updates old memories given the new input. It is called generalization as there is an opportunity for the network to compress and generalize its memories at this stage for some intended future use.
- An output feature map O, which produces a new output in the feature representation space given the new input and the current memory state.
- A response component R which converts the output into the desired response format – for example, a textual response or action.

The I module is used for the conversion of the input to a feature representation which is read by the Generalization module. The G module maps the query to an appropriate memory slot by applying a function H. The output feature reads the memory state and performs inference to pass it on to the Response module. The Response module converts the information into a readable format.

In general, case, when someone mentions a topic in a random discussion, our brain processes the topic and finds the information relevant to the topic in the memory. The retrieved memory is converted into a response. Thus, the working function of memory nets resembles the ones of the human brain.

2.2 Datasets

Text to Image synthesis requires a dataset that can be used for the training of the architecture. The datasets for the text to image synthesis used frequently in the domain are the Caltech Birds [2] and Microsoft Coco. These datasets are also seen frequently in the literature on image captioning.

2.2.1 Caltech Birds

Caltech-UCSD Birds 200 [2] (CUB-200) is an image dataset with photos of 200 bird species (mostly North American). There are a total of 11,788 images of bird species distributed according to their classes. Each image has a short description or caption attached to it with a text file and the bounding box coordinates of the bird. The birds images were downloaded from the website Flickr and filtered by workers on Amazon Mechanical Turk [2].


Figure 2.10: Caltech Birds images [2].


Figure 2.11: Microsoft Coco images [3]

2.2.2 Microsoft Coco

COCO [3] is large-scale object detection, segmentation, and captioning dataset. Common Objects in Context is abbreviated as Coco. The dataset is a huge compilation of 83000 labelled train images and 41000 labelled validation images. This dataset descriptions cover the everyday descriptions like the description of a house or a general scenario in a park. The dataset images are achieved by gathering images of complex everyday scenes containing common objects in their natural context [3].

2.2.3 Oxford Flowers

The Oxford flower dataset [40] is a compilation of 102 different categories of labelled flower images. The images in each category range from 40 to 102.

2.3 Previous Papers on Text to Image Synthesis

This section provides an overview of the papers used for the reference in the text to the Image domain. Word representation vectors are generated from the text by bidirectional LSTM. GAN networks take these text descriptions as input and generate output images. The network is trained on the sentence representation according to the datasets. The main idea behind each of the research papers is to find a semantic relation between the generated image features and the sentence embeddings. The architectures are divided into two categories notably on the amount of GANs used in the architectures. Single stage methods [4, 26] are concerned with the use of one generative network

to generate images while multi-stage consist of the utilization of two or more GAN networks. This section aims to give an understanding of the benefits and drawbacks of each of the architecture. I have compiled some of the papers imperative to my research and my research is a continuation of these papers. The architectures are coupled with their generated images to give an overview of the improvement done throughout the years.

2.3.1 Generative Adversarial Text to Image Synthesis

Reed et. al. 2016 [4] was the first attempt at text to image synthesis by using GAN to generate images. This architecture is also called the first single-stage model as this model uses a single GAN.

The captions of a text dataset are taken and reduced to their word embedding. The encoders used in the network are trained by using deep convolutional and recurrent text encoders to learn a correspondence function with images. The architecture is mentioned in the figure 2.12. It uses a bi-directional LSTM for the generation of short word embedding. The embedding is reduced to a sentence vector of dimension 128 by a linear layer and a leaky ReLU activation. It is coupled with a random distribution and passed through a generator consisting of upsampling stages. The generated image is treated with a discriminator to form a representation concatenated with the description formed from the caption to have a result.

The Generative Adversarial Text to Image Synthesis considers training the network under 3 conditions:

- GAN - direct way to train the GAN with real images and fake generator images. The discriminator is trained on the real images with matching texts and synthetic images with the arbitrary text while Generator is trained by Discriminator output.

- GAN-CLS - Training the discriminator on real images with matching texts, synthetic text with the same text and synthetic text with arbitrary text.

$$
\begin{aligned}
s_r &\leftarrow D(x, h)(real \quad image, right \quad text) \\
s_w &\leftarrow D(x, \hat{h})(real \quad image, wrong \quad text) \\
s_f &\leftarrow D(\hat{x}, h)(fake \quad image, right \quad text) \\
L_D &= log(s_r) + \frac{log(1 - s_w) + log(1 - s_f)}{2} \\
L_G &= log(s_f)
\end{aligned}
\tag{2.5}
$$

- GAN-INT: Generator is trained with images generated from interpolation between two texts

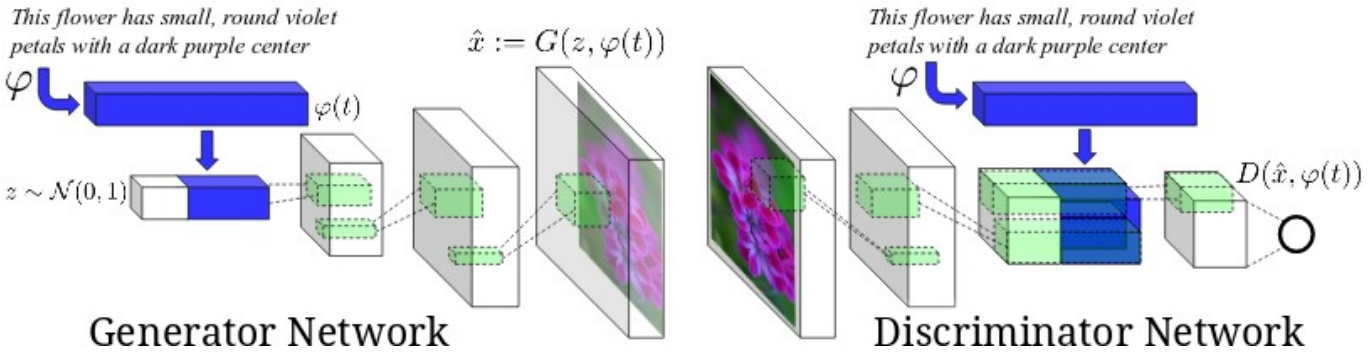


Figure 2.12: Single stage GAN architecture for Text to Image Synthesis [4]

and trained for better robustness of the image generation.

$$E_{t1,t2\sim p_{data}}[log(1 - D(G(z, \beta t_1 + (1 - \beta)t_2)))] \tag{2.6}$$

where z is drawn from the noise distribution and β interpolates between text embeddings t_1 and t_2. The value for the β is set to 0.5.

The datasets used for the training are the Caltech Birds and the Oxford flowers. The images generated were of the format of 64X64 by the single-stage architecture.

2.3.2 Stack GAN

The Stack-GAN[5] is the first architecture in the series to have a multi-stage network for image generation. The sentence embedding generation procedure generates the sentence embeddings which are used for the generation of the single-stage image.

The Stack-GAN paper [5] introduces a technique of Conditional Augmentation where latent variables are sampled from a normal distribution. The latent distribution is constructed from a mean and a variance formed from the same text embedding. The conditional augmentation increases the perturbation for a text embedding increasing the training pairs making the model robust. The generator is trained with the help of a Kullback Liebler divergence between the final text description and standard Gaussian normal distribution as a regularization.

$$Loss = D_{KL}(N(\mu(\psi_t), \sum(\psi_t))||N(0, I)) \tag{2.7}$$

where $\mu(\psi_t)$ is the mean of the latent text description and $\sum(\psi_t))$ is the standard deviation.

The Conditional Augmentation is applied to the input of the generator of the single-stage GAN to generate preliminary 64X64 images. Low-resolution images generated by stage-I GAN gener-

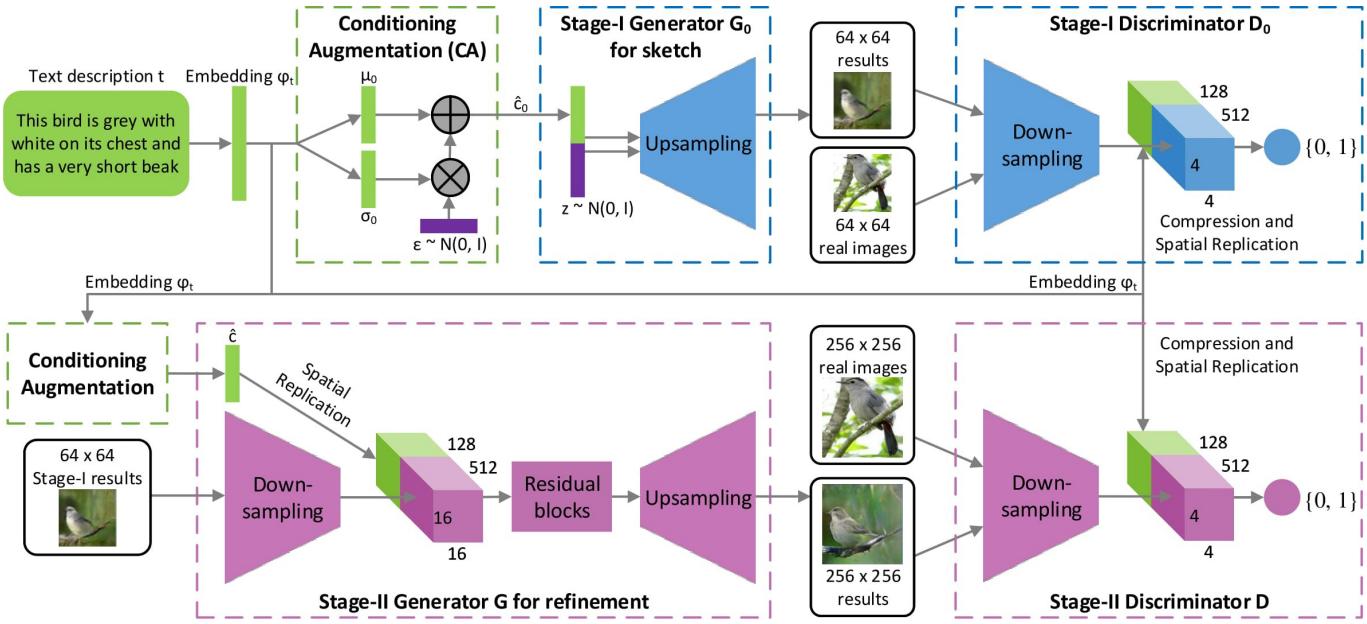


Figure 2.13: Stack-GAN [5] architecture

ally lack vivid object parts and might contain shape distortions. The second stage improves the information of the latent vector by using a new conditional augmentation on the original text embedding. The conditionally augmented vector is combined with image features downsampled from stage-I GAN by concatenation. This helps in capturing new information about the image omitted by the stage-I GAN.

The encoded image features coupled with text features are fed into several residual blocks, which are designed to learn multi-modal representations across image and text features. The second stage image is generated from the upsampling layers further added to the network. The conditional augmentation, concatenation, residual blocks and the upsampling layers form the second generator which can be trained on a different loss function. The images extracted from this stage are of the resolution 256X256. There can be a cascade of stages for better image generation. The image features are enhanced by the new techniques employed in the second stage.

The generator objective function comprises a Discriminator output and the conditional Kullback-Liebler Divergence.

$$L_D = E_{(I,t)\sim p_{data}}[logD(I_o, \varphi_t)] + E_{s_0\sim p_{G_0}, t\sim p_{data}}[log(1 - D(G(z, \hat{c}), \varphi_t))],$$
$$L_G = E_{z\sim p_{G_0}, t\sim p_{data}}[log(1 - D(G(z, \hat{c}), \varphi_t))] + \lambda D_{KL}(N(\mu(\varphi_t), \Sigma(\varphi_t))||N(0, I))$$

where the real image I_o and the text description t are from the true data distribution p_{data}. z is a noise vector randomly sampled from a given distribution of p_z (Gaussian distribution in this paper). λ is a regularization parameter that balances the two terms.

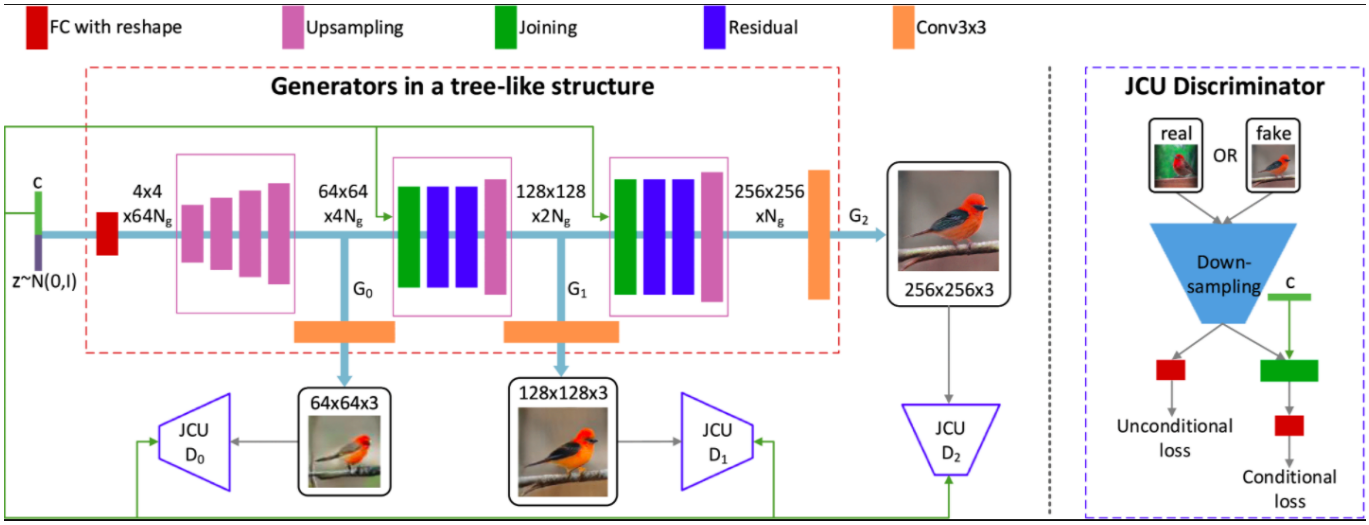


Figure 2.14: Stack-GAN++ [6] architecture

The datasets used for the training are the Caltech Birds, Microsoft Coco and the Oxford flowers.

2.3.3 Stack-GAN ++

The Stack-GAN v2 [6] focuses on generating sharper and better images from the generators than stack-GAN-v1. The idea behind the architecture is to smoothen the image distributions- first based solely on image and second based on the text descriptions. The purpose is achieved by a cascade of generator networks with a gradual increase in the resolution of the generated image descriptions. Cascade of GANs allows the network to account for the features of the text descriptions by modelling data distributions at multiple scales. If any of the model distributions shares support with the real data distribution at that scale, the overlap could provide a good gradient signal to expedite or stabilize the training of the whole network at multiple scales.

The Down-sampling module output of the discriminator as in Stack-GAN-v1 is sourced separately to generate an unconditional loss which compares the generated image distribution with real data distribution. This loss acts along with the distribution loss concerning the text descriptions for the training of the generators. The objective function used for the training of the discriminator is also supplemented with an additional loss. The unconditional loss determines whether the image is real or fake while the conditional one determines whether the image and the condition match or not.

Due to the addition of the generators, it is unlikely that the images from the adjacent generators share a similar basic structure and colour. The Stack-GAN-v2 implements a new regularization based on the RGB maps of the image. The mean of the pixels of the output image is taken and their standard deviations are calculated from the means. The purpose of the regularization term is

to reduce the difference between the mean and standard deviation between the image generated by the last generator and the current one. Thus, it tries to impart the same colour and structure to the two images by regularizing the generator objective function.

The generator objective function for the Stack-GAN++ consists of the introduced conditional loss, unconditional loss and colour regularization. The discriminator objective function is also composed of two conditional and non-conditional losses.

$$
\begin{aligned}
&L_{D_i} = -E_{x_i \sim p_{data_i}}[logD_i(x_i)] - E_{s_i \sim p_{G_i}}[log(1 - D(s_i))] \\
&- E_{x_i \sim p_{data_i}}[logD_i(x_i, c)] - E_{s_i \sim p_{G_i}}[log(1 - D(s_i, c))], \\
&L_G = -E_{s_i \sim p_{G_i}}[log(D(s_i))] - E_{s_i \sim p_{G_i}}[log(D(s_i, c))] + \\
&\alpha \frac{1}{n} \sum_{j=1}^{n} \left(\lambda_1 ||\mu_{s_i^j} - \mu_{s_{i-1}^j}||_2^2 + \lambda_2 ||\Sigma_{s_i^j} - \Sigma_{s_{i-1}^j}||_2^2 \right)
\end{aligned}
$$

where c is the context, x_i is the input , $s_i = G(x_i)$ is generator distribution and G and D are the generator and discriminator respectively. The μ and Σ in the last term correspond to the mean and the standard deviation respectively of the average of the colour component of the j^{th} element of the batch size n and i^{th} generator. α is the scaling factor for the regularization parameter.

The method is applied to the same datasets as those of the StackGAN-v1. The two research papers on the StackGANs led to the further development of the research architecture on the multi-stage structure for image generation.

2.3.4 Results

The image results carved out by the different architectures can be clearly seen in the figure 2.15. The architectures of GAWNN and GAN-INT-CLS are the single-stage ones while the others belong to multistage architectures. The 64X64 images generated by the GAN-INT-CLS show the size and shape of the bird and its background. Thus, they lack some vivid parts. This is enhanced by the results of GAWNN[26] which is conditioned on the location constraints. The images contain better information and better resolution than the first GAN architecture.

The StackGAN[5] generates better images with the conditional augmentation and the multi-stage architecture. The conditional augmentation allows it to have multiple images conditioned on the same text description. The single-stage sketches the general features of the images which are enhanced by the second conditional and residual models to give better well-defined object features like beak and tail. Stack-GAN++ generates better images with the colour consistency regularization and end to end training scheme.

In the CUB birds data from the 4 architectures displayed in figure 2.15, it can be observed that

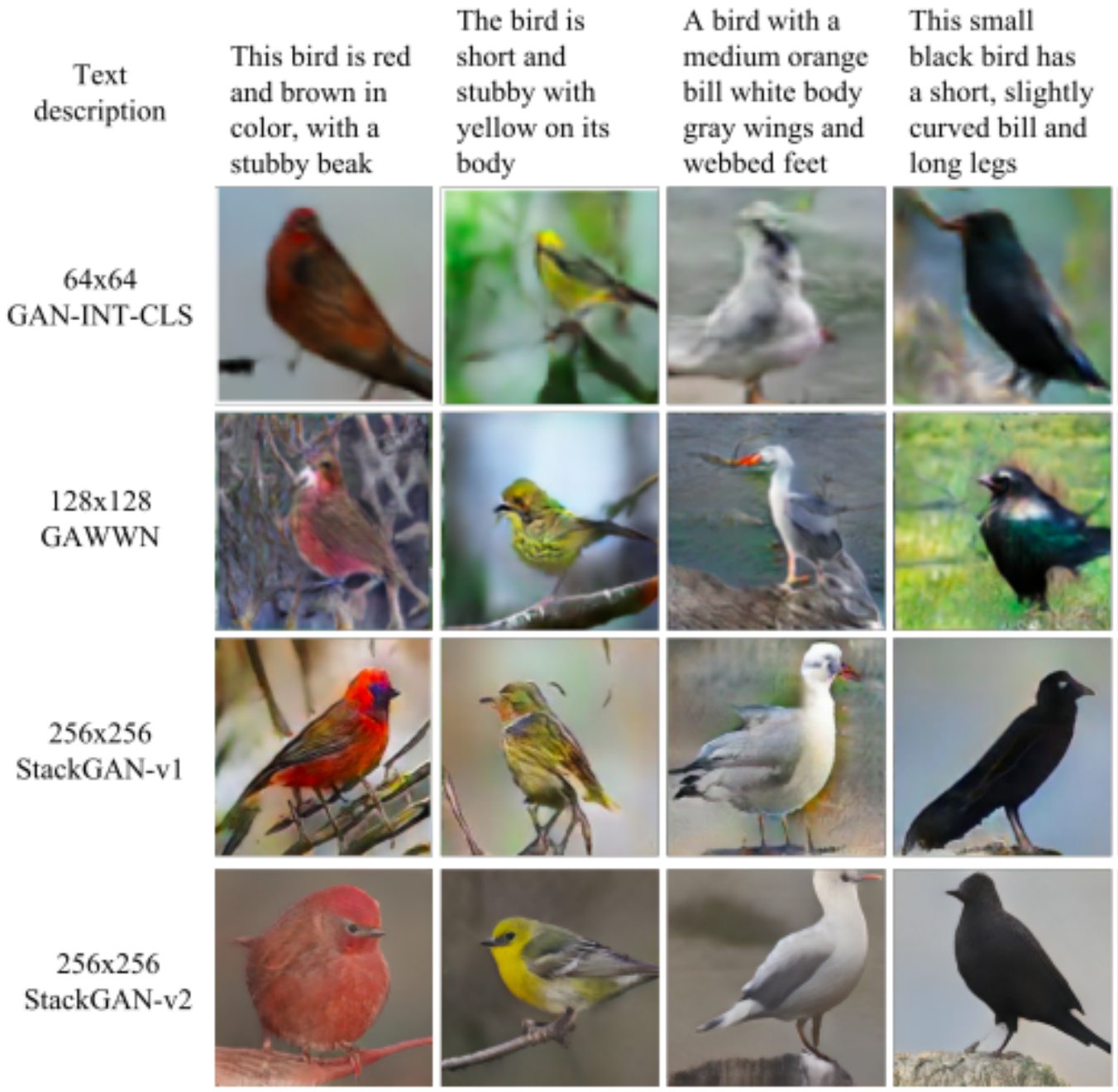


Figure 2.15: Comparison of single stage and multi-stage architectures by generating Caltech Birds images taken from the Stack-GAN++ research paper [6]

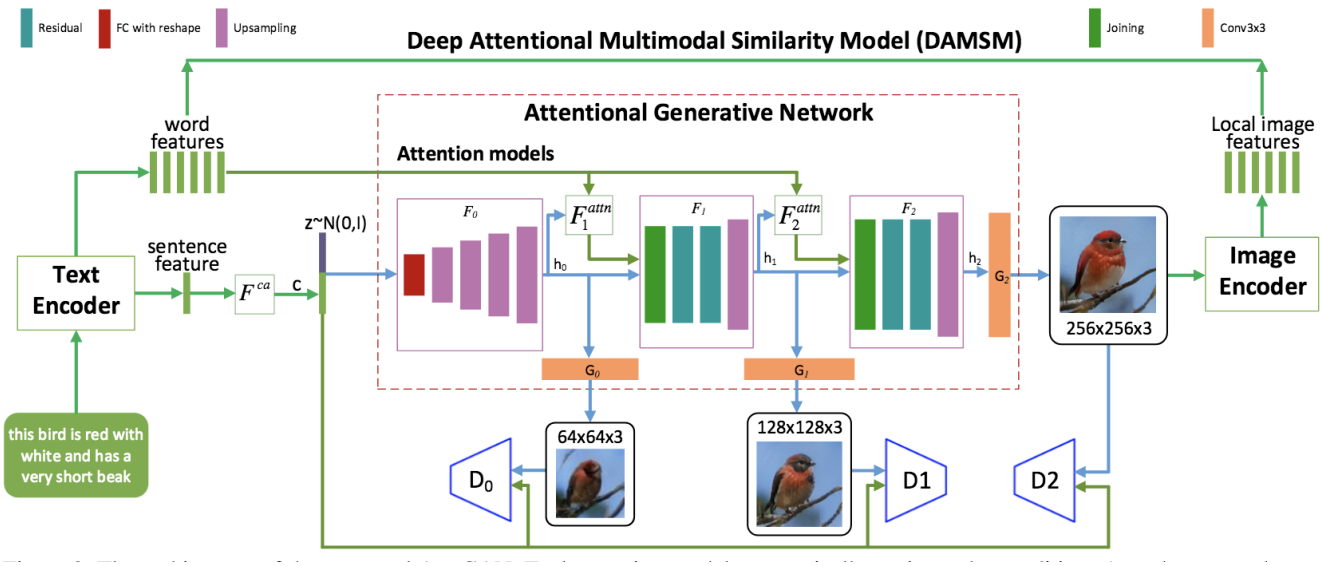


Figure 2.16: AttnGAN [7] architecture

the multistage architectures are effective in generating plausible images with better resolution and distinct features conditioned well on the texts. The Stack-GAN++ image results feel more realistic as compared to the artistic canvas drawings of the Stack-GAN.

2.3.5 AttnGAN

AttnGAN[7] forms part of the multistage architecture constructed on the concept similar to the StackGAN[5]. But it serves the purpose of improving the image quality by generating fine-grained images from the text descriptions. It applies an attention-driven approach for extracting information from the text embedding and enriching the information of the visual features.

The text features are extracted using a bi-directional LSTM encoder. The last two hidden vectors form the word embeddings while sentence embeddings are extracted from the output of the encoder. The attnGAN tries to implement fine-grained word-level information with the help of an attention network. The network consists of a set of hidden vectors that are being processed by the GANs to generate an image. The hidden vectors contain the image-level information like the sentence feature of the single-stage GAN[4]. The hidden vector information is improved with the help of an attention network between the word embeddings and the currently hidden vector.

The sentence feature is passed through a conditional augmentation network before upsampling it. The upsampled vector forms the first hidden vector. The further hidden vectors are found by upsampling the concatenated vectors of attention output and the previously hidden vector. The attention is calculated between the word embedding s and the hidden vector. The word embed-

dings are passed through a fully connected network before being applied as an attention key to the query(hidden vector). The output of the attention network is the corresponding extracted value.

The GANs take a hidden vector as input for the image generation. Higher the hidden vector number, the better the information stored and thus could be applied to a generator with a better image resolution. The resolution can be doubled as the information is enriched by attention in the hidden vector. The GANs give images of 64X64,128X128 and 256X256 in succession. The Discriminator and Generator are similar to that of the StackGAN-v2 model. Therefore, the loss function of the Discriminator is the same as that for the StackGAN-v2.

In terms of the Generator objective function, AttnGAN introduces a new model for the training of the attention network. The loss model is defined by the architecture of the Deep Attention Multimodal Similarity model.

The Generator objective function for the AttnGAN consists of the Image loss, Image-context loss, conditional augmentation regularization and the DAMSM loss. The generator is trained on the datasets of Caltech Birds and the Microsoft Coco.

2.3.5.1 Deep Attention Multimodal Similarity Model Loss

The DAMSM[7] model is a new model introduced in AttnGAN[7] to understand the images generated and their relation to the word features. A normalized symmetry matrix is calculated between the image vectors and the word embeddings. Then, an attention model is introduced for the computation of the regional context for each word. A relevance score is calculated between the word and the image using the cosine similarity function.

The DAMSM loss is designed to learn the attention model in a semi-supervised manner, in which the only supervision is the matching between entire images and whole sentences. The word vectors and sentence vectors are matched with the image and vice versa using the posterior probability function and added to form the complex loss function of the DAMSM loss. This loss serves to train the network to increase the relationship between the word-level information and the image features.

2.3.5.2 Objective function

The generator objective function for the AttnGAN includes the conditional loss of the image, unconditional loss and the DAMSM loss component L_{DAMSM}. The discriminator loss function consists of the unconditional and conditional loss components. λ is used to depict the varying

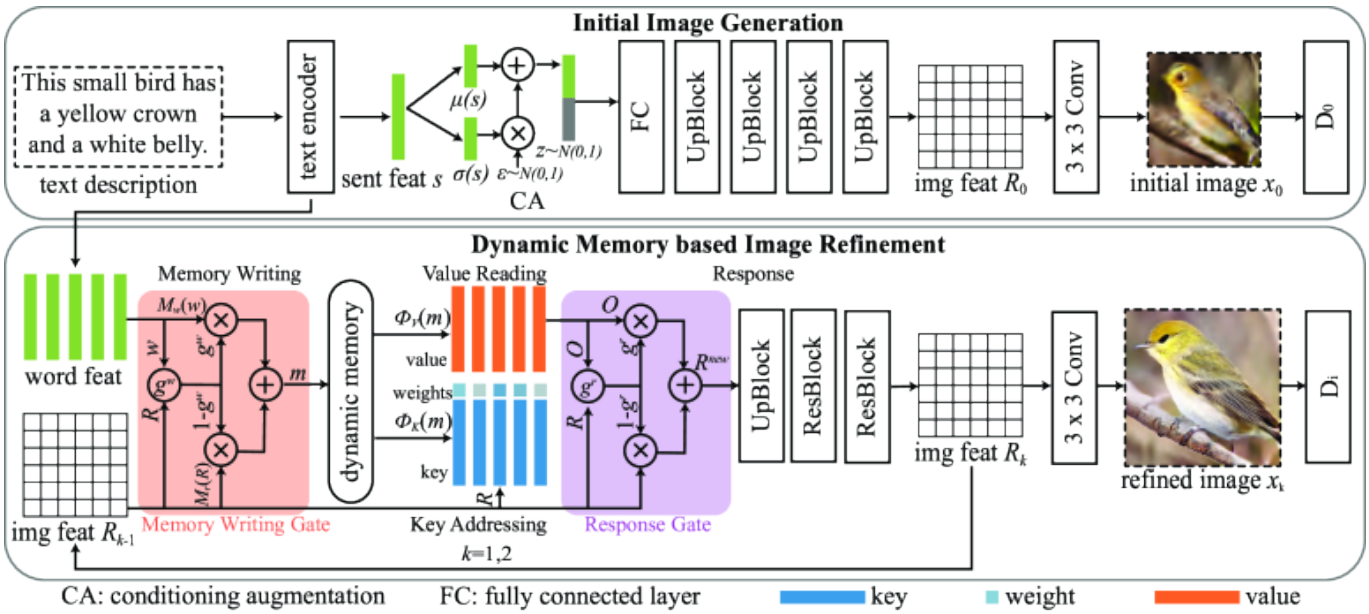


Figure 2.17: Dynamic Memory GAN architecture [8]

strength of the DAMSM function.

$$L_{D_i} = -E_{x_i \sim p_{data_i}}[logD_i(x_i)] - E_{s_i \sim p_{G_i}}[log(1 - D(s_i))] - E_{x_i \sim p_{data_i}}[logD_i(x_i, c)]$$
$$- E_{s_i \sim p_{G_i}}[log(1 - D(s_i, c))],$$
$$L_{G_i} = -E_{s_i \sim p_{G_i}}[log(D(s_i))] - E_{s_i \sim p_{G_i}}[log(D(s_i, c))] + \lambda L_{DAMSM}$$

Here, D_i and G_i is the ith Discriminator and Generator respectively. $s_i = G(x_i)$ is the generator output, c is the context, p_{G_i} is the generator distribution and p_{data} is the data distribution.

2.3.6 DM-GAN

Dynamic Memory GAN[8] is an architecture based on the multi-stage concept. It can be thought of as being based on the concept of Stack-GAN. Attn-GAN uses the Attention to improve the semantic information of the image features. DM-GAN uses Dynamic Memory to improve semantic information of the image features.

The input for the single-stage is taken by concatenating the hidden states of the Bi-LSTM encoder. It passes through a Conditional augmentation network, linear and UpBlock layers to form R_0 image features. The image feature is used to generate the first image of 64X64 resolution by a 3X3 convolution.

The image feature is used in the dynamic memory along with the word embeddings for generating the next image feature. Thus, the network is a set of dynamic memories cascaded improving

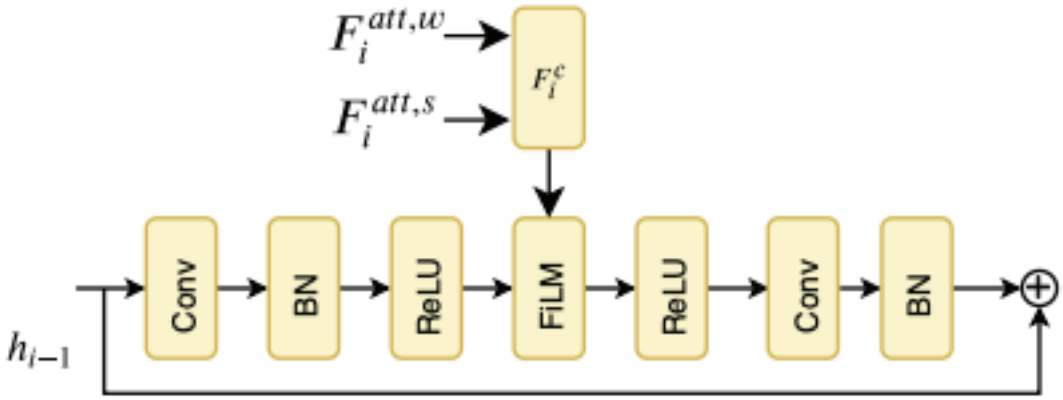


Figure 2.18: Changed architectura1 unit of the e-AttnGAN [9]

the image feature knowledge by using the word embeddings as the input to the memory network and the image features as the memory. The increase in the image information can be converted into better resolution images.

The Generator objective function consists of the image loss concerning the original image, image loss concerning the context, conditional augmentation loss and the DAMSM loss.

$$
\begin{aligned}
L_G &= \sum_i L_{G_i} + \lambda_1 L_{CA} + \lambda_2 L_{DAMSM} \\
L_{D_i} &= -E_{x_i \sim p_{data_i}}[logD_i(x_i)] - E_{s_i \sim p_{G_i}}[log(1 - D(s_i))] - E_{x_i \sim p_{data_i}}[logD_i(x_i, c)] \\
&- E_{s_i \sim p_{G_i}}[log(1 - D(s_i, c))]
\end{aligned}
\tag{2.8}
$$

where $s_i = G(x_i)$ is the generated image, c is the context or the global sentence vector, x_i is the input, p_{G_i} is the generated distribution and p_{data} is the data distribution.

The DAMSM loss tries to train the Memory network in the Dynamic Memory GAN architecture as well as the other weights.

2.3.7 e-AttnGAN

e-AttnGAN[9] is a new branch on the attnGAN network. The architecture of the AttnGAN is tweaked to capture the global vector dependencies of the image features. The hyperparameters of the attention network are trained on the DAMSM loss originally introduced in the AttnGAN. The global sentence vector is applied to an attention network along with the image features to form a new attention output. This output is concatenated with the attention output of the word embeddings and then applied to the upsampling layer to form a new hidden vector.

The author of the e-AttnGAN has applied new techniques to ease the training of the network

as well as improving the information of the hidden vectors. The previously hidden vector and the concatenated attention vectors are applied to a convolutional network to form a new network that can be used to extract a new hidden vector.

The objective function or the components considered is the same as the AttnGAN but there are classification losses appended. The classification loss applied at the discriminator's output ensures that the network pays more attention to the significant attributes of the image. It also uses the feature matching loss used in a generator for easy training of the network.

The author used the Fashion dataset for the application of the network.

2.3.8 Attention on Attention

Attention mechanisms are widely used in everyday applications like encoder-decoder networks for image captioning. It generates a weighted average on encoder vectors for the image captioning process. However, the decoder has less information about the relation between the initial vectors and the attended vectors are related.

AOA network[10] is constructed to extend the conventional attention network to determine the relevance between the keys and the queries. AOA first generates an "information vector" and an "attention gate" using the attention result and the current context, then adds another attention by applying element-wise multiplication to them and finally obtains the "attended information", the expected useful knowledge. It acts like another attention to the attended network with the help of context.

AoA generates an attention gate and an information vector via two linear transformations and applies the gate to the vector to add second attention, where the techniques are similar to some other work: GLU, which replaces RNN and CNN to capture long-range dependencies for language modelling; multi-modal fusion, which models interactions between different modalities (text and image) and combines information from them.

AoA is used in image captioning as the process requires the image features to be deciphered in a way to extract the relationship between different parts of the image. AoA determines the relevance between the attention result and query, while multi-modal fusion combines information from different modalities.

2.3.9 Mode-collapse

It is difficult to draw an object than to recognize one. Generative models are tasked with the generation of images from random latent vectors and a context(for cGANs). Thus, they encounter some problems during generation. Some of them could be the non-convergence issue, mode-collapse[11] and diminished gradient issue. Non-convergence deals with the unstable training of

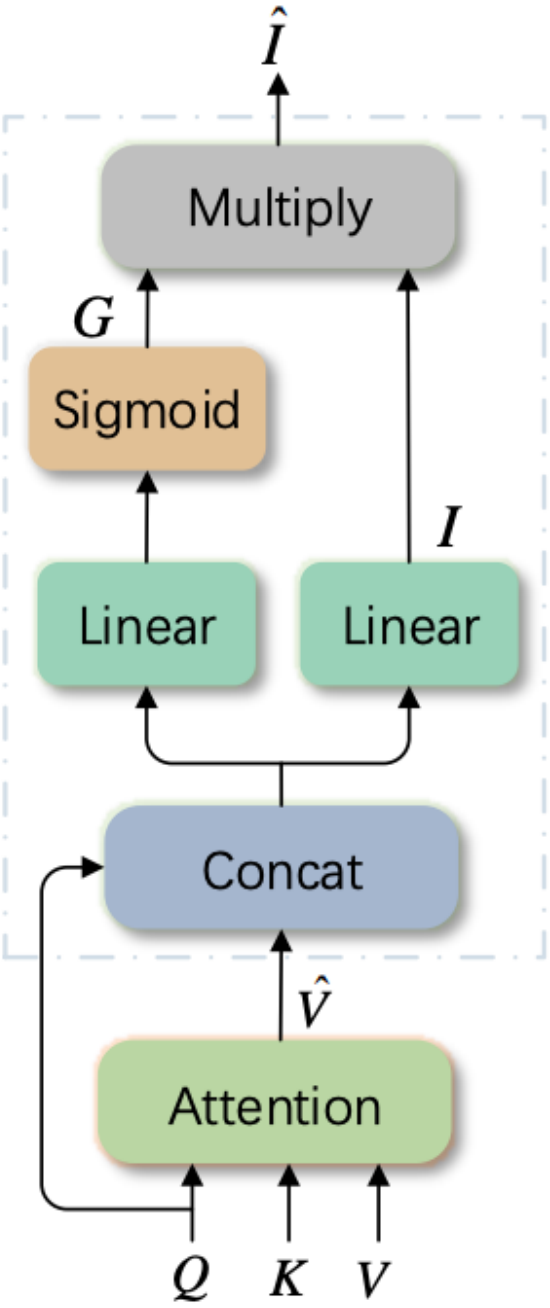


Figure 2.19: Attention on Attention network with its respective components [10]

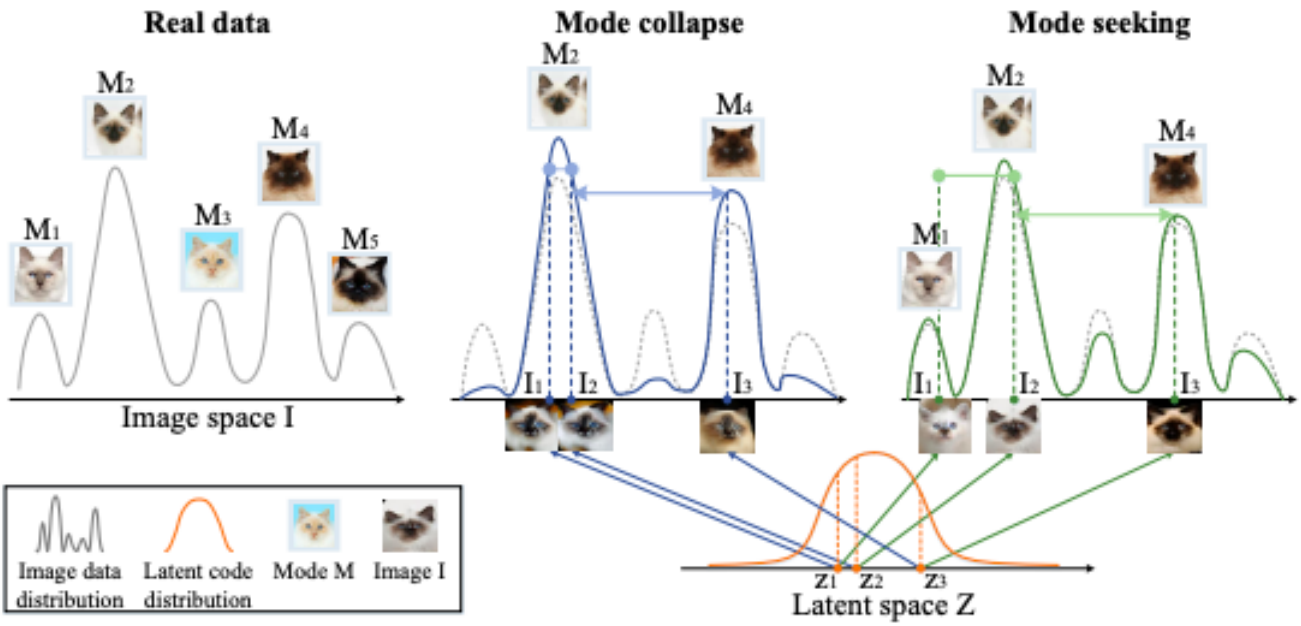


Figure 2.20: GAN modes for mode-collapse and after applying mode-seeking function. [11]

the GANs where the mode parameters oscillate, destabilize and never converge. Mode collapse is an issue where generator training gives rise to a limited number of output modes.

Real data distribution contains numerous modes. However, when mode collapse occurs, generators only produce samples from a few modes. From the data distribution when mode collapse occurs, we observe that for latent vectors z_1 and z_2, the distance between their mapped images I_1 and I_2 will become shorter at a disproportionate rate when the distance between two latent vectors is decreasing. We present on the right the ratio of the distance between images to the distance of the corresponding latent vectors, where we can spot an anomalous case where mode collapse occurs. The observation motivates us to leverage the ratio as the training objective explicitly.

GAN tries to learn the data distribution by using the reverse cumulative distribution function on the random uniform distribution. Thus, two close points on the uniform scale might be mapped to two image modes with less difference between them.

2.3.10 Mode-seeking GAN

MSGAN[11] is a GAN constructed based on improving the intrinsic property of a GAN. The multimodal output of the GAN is responsible for the generation of many output modes. But the output image is selected from some of these modes as some modes become more dominant during the GAN training as compared to the others. This affects the diversity of the GAN output.

The mode-seeking loss function implemented by the MSGAN paper concerns itself with the

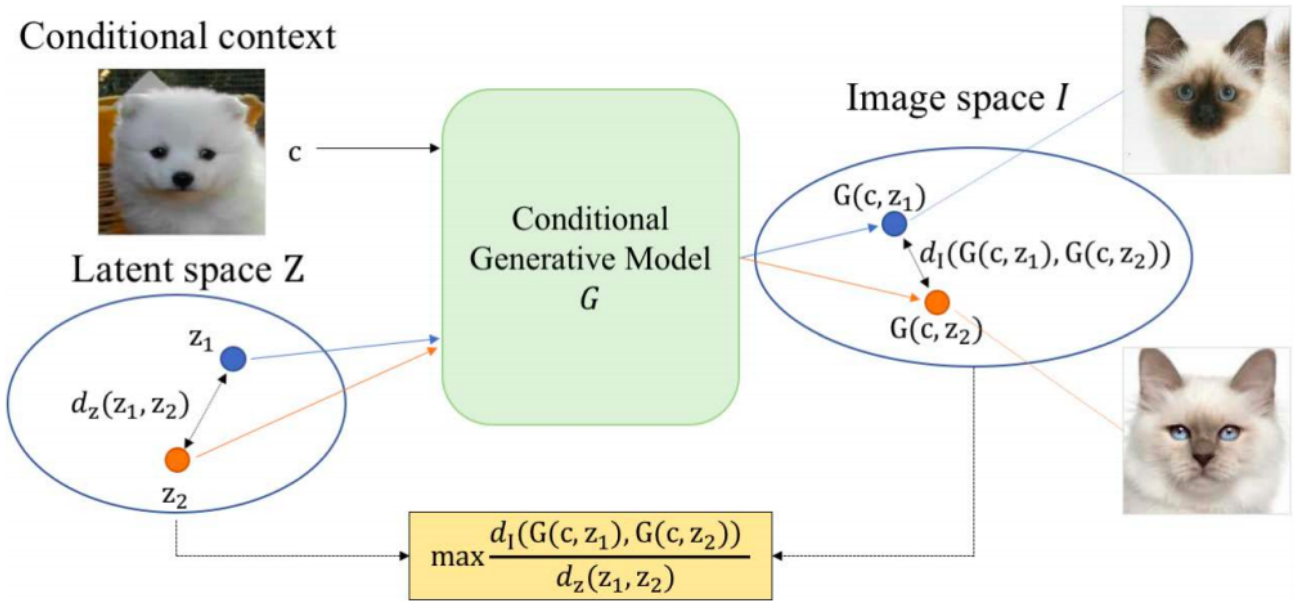


Figure 2.21: Application of mode-seeking loss function to a generative model. [11]

strengthening of the hidden modes of the GAN. The objective function of the GAN is given by :

$$L_{ms} = \max_{G} \left[\frac{d_I(G(c, z_1), G(c, z_2))}{d_z(z_1, z_2)} \right] \tag{2.9}$$

where $G(c, z_i)$ is the image from the noise vector z_i, c is the global sentence vector and D is the distance between the two tensor arguments. The MSGAN method samples the GAN with two instances of the random noise in the latent vector which gives rise to two output images. The expression maximizes the distances between the output images to the noise functions.

The MSGAN is can be used for the image to image, image to text and text to image architectures.

2.4 Results

The results displayed in figure 2.22 are the images extracted from the text to image architectures for the multi-stage. Stack-GAN starts with the conditional augmentation feature. AttnGAN features the attention network for the image-text relation. DM-GAN uses the memory network for finding a semantic relation between the text feature and image vectors.

After the observation of the results for the CUB birds, the difference between the quality of images is quite evident for the architectures. The GAN-INT-CLS single-stage architecture generates

Figure 2.22: Text to Image Synthesis architectures and their results on Caltech Birds dataset.

a blurry and rough design for the birds and the colour of the background but they don't have the proper object boundary. The Stack-GAN[5] images show the object boundaries very well with the features described well.

The AttnGAN results show that the images are trying to take into account specific details of the text by focusing on specific words in the text. The images depict the vivid descriptions of the text. The DMGAN clears out the fuzzy background and enhances the quality of the images generated with good detail of the text description.

The images are evaluated based on metrics used for the GAN images. A GAN output is generally measured by checking it through a human evaluation and verifying its image quality. The training of GANs is done similarly. But few metrics have tried to quantify the training of GANs. The metrics used for the verification of the text to image output are the Inception Score and Frechet Inception Distance.

2.4.1 Inception Score

The inception score involves using a pre-trained deep learning network Inception-v3 model used for image classification to gauge the quality of the generated images. The probability of the image belonging to each class is predicted and then calculated into an inception score.

Inception Distance calculates the score according to the conditional probability of a generated image being in one of the class labels.

$$I = exp(E_x D_{KL}(p(y|x)|p(y))) \tag{2.10}$$

where y is the class label corresponding to the image, x is the generated image, E is the expected value and D_{KL} is the Kullback-Leibler divergence. The inception score tries to measure two qualities in an image :

- Image Quality: it checks whether the image resembles an object.
- Image Diversity: whether a wide range of images is being generated.

Image Diversity is a metric to measure the inherent property of multimodality of GANs or variety of output modes of GANs. The inception score tries to measure the amount of diversity in a model. Higher the inception score, the better is the quality of the images generated and the number of modes produced.

In the original calculation of the Inception Score, the images, owing to their quantity, are randomly split into 10 equal groups. Inception scores are calculated over the groups which are then averaged and the standard error is calculated. Thus the inception score of a real dataset like CIFAR-10 is 11.24 ± 0.12.

2.4.2 Frechet Inception distance

FID[29] considers not only the synthetic data distribution but also how it compares to the real data distribution. It directly measures the distance between the synthetic data distribution and the real data distribution.

Like the inception score, the FID score uses the inception v3 model. Specifically, the coding layer of the model (the last pooling layer before the output classification of images) is used to capture computer-vision-specific features of an input image. These activations are calculated for a collection of real and generated images.

Assuming the feature embeddings follow a multidimensional Gaussian distribution, the synthetic data's Gaussian with mean and covariance (m, C) is obtained from p(.) and the real data's Gaussian with mean and covariance (m_r, C_r) is obtained from p_r(.). The FID is calculated as :

$$FID = ||m - m_r||_2^2 + Tr(C + C_r - 2(CC_r)^{\frac{1}{2}}) \tag{2.11}$$

where Tr is the trace of the matrix. A lower FID indicates better-quality images; conversely, a higher score indicates a lower-quality image and the relationship may be linear. The FID metric is focused on the relationship between the real data distribution and the generated fake image data distribution.

2.4.3 Metric scores

The table 2.1 gives a detailed overview of the results of the above architectures. the results are classified according to the Inception score and FID score. As the architecture develops, the Inception score increase as the image quality and diversity increase. The FID decreases with the increase in Image quality. Table 2.1 shows the comparison of the metric scores for the images generated by the different architectures studied during the literature review. There is a gradual increase in the metric scores as we go down the table. The metric scores improve as the semantic word-level information improves through the architectures. The main objective of our research is to understand this growth and try to introduce techniques to improve the quality of the existing architectures.

Architecture	CUB-FID	CUB-IS	Coco-FID	Coco-IS
GAN-INT-CLS[6]	68.79	2.88 ± .04	60.62	7.88 ± .07
StackGAN[6]	51.89	3.70 ± .04	74.05	8.45 ± .03
StackGAN++[6]	15.3	3.84 ± .06	33.88	8.30 ± .10
AttnGAN[7]	23.98	4.36 ± .03	35.49	25.89 ± .47
DMGAN[8]	16.09	4.75±0.07	32.64	30.49±0.57

Table 2.1: FID scores and Inception score for SOTA architectures

CHAPTER 3

DM-GAN WITH MODE-SEEKING LOSS FUNCTION

This chapter describes the work done during research in the construction phase. The construction phase involves building models, obtaining the datasets, understanding the dataset structure, model search and improvement. I will delve deep into my approach towards the research before explaining the construction of the architectures. My work can be found in the research paper "Improving Text to Image Generation using Mode-seeking Function"[12]

My perspective behind the problem of text to image synthesis was to improve the quality of generated images and check their dependence on the text descriptions. The improvement could be achieved in two kinds of approaches in a deep learning problem. One approach could be involving the modification of the loss function and the second approach can focus on the architectural change which might bring about better attention or better enrichment of the semantic information.

Mode-seeking loss modifies the loss function of the generator objective function while refined attention changes the architecture of the AttnGAN architecture.

3.1 Mode-Seeking GAN Effect on Dynamic-Memory GAN

The Dynamic Memory GAN[8] is a recent state-of-the-art models in the text-to-image synthesis domain. I try to explain the code of the DM-GAN in better detail here before the explanation of the introduced modification. The original loss function of DM-GAN is appended with the mode-seeking GAN objective.

The DM-GAN network consists of the Bi-LSTM encoder, conditional augmentation, 3 GANs and memory architecture. The Bidirectional LSTM encoder has two LSTMs running in opposite generating two hidden vectors and the word embeddings. The global sentence vector is formed by concatenating two hidden vectors while the word embeddings are the output of the encoder.

The text is preprocessed before the application of the encoder by tokenization and stopword

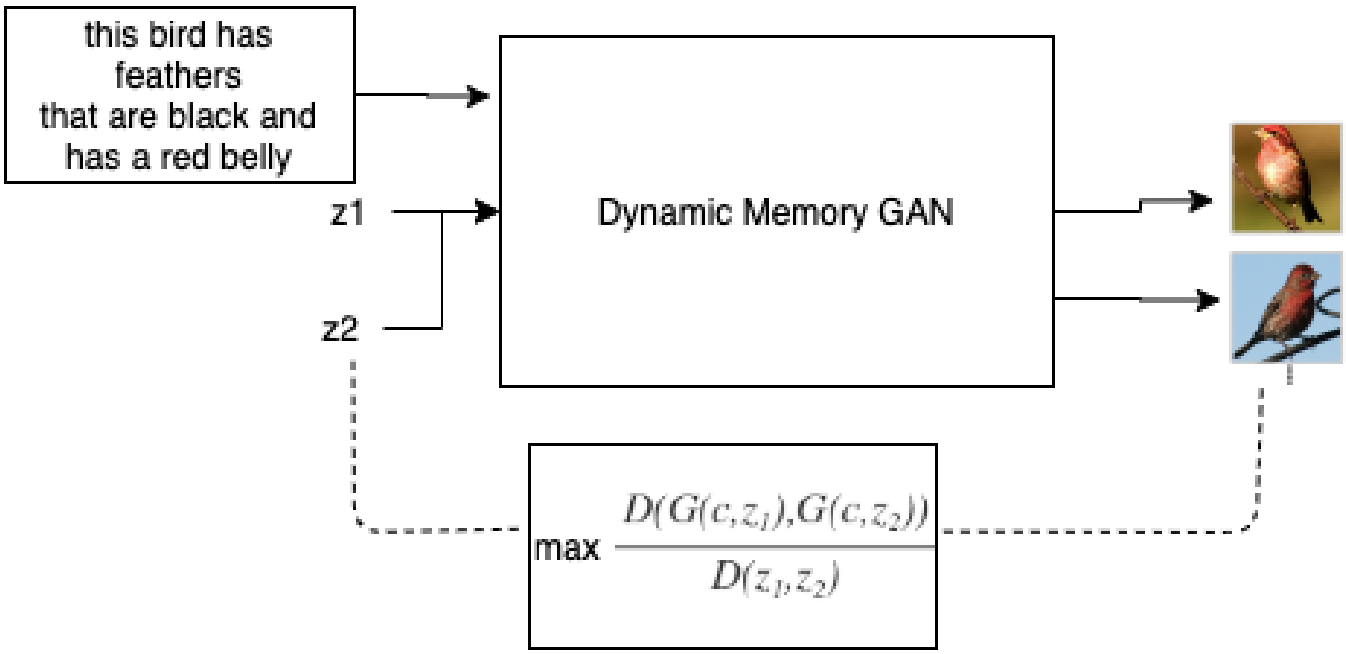


Figure 3.1: Application of mode-seeking function to the DM-GAN architecture [12]

removal. The words are converted to one-hot encodings before the application of the encoder. The word embeddings are extracted from the initial vectors using the bi-directional LSTM encoder.

The sentence embedding vector or the global vector is used in the conditional augmentation stage and the single-stage generator for the generation of 64X64 images. The word-level information is encoded in the word embedding and is used for the improvement of the image features in the next stages through a memory module. The updated image features are inputs to the higher generators.

I would like to explain the DM-GAN memory module in the context of my understanding of memory. The DM-GAN memory consists of 4 parts :

- Memory writing: this module encodes prior information from the image vectors and the test vectors through the convolutional operation. That is, obtain attention between feature map and word vector.
- Key addressing: In the Key addressing section, key memory is used to search related memories by utilizing the similarity function.
- Value reading: Value reading is the process of leveraging the key memories to prepare the output memory representation.
- Response gate: The response rate is the section for the concatenation of the image and formed vectors.

Generally, during multimodality, there is a loss in some of the output modes as they have less tendency to occur than the majority modes having high intensity in the output distribution. The objective function for the mode-seeking loss is the magnitude of the change in the images to the magnitude of change in the noise function.

Thus, two noise functions are sampled through the network and extract two images. These images are compared and the loss function tries to strengthen the modes so that there is more diversity in the model results.

3.1.1 Objective Function

The objective function for the discriminator consists of conditional and unconditional loss.

$$\begin{aligned} L_{D_i} = -\frac{1}{2}[&\underbrace{E_{x\sim p_{data}} log(D_i(x)) + E_{x\sim p_{G_i}} log(1 - D_i(x))}_{unconditional \quad loss} \\ &+ \underbrace{E_{x\sim p_{data}} log(D_i(x,s)) + E_{x\sim p_{G_i}} log(1 - D_i(x,s))}_{conditional \quad loss}] \end{aligned} \tag{3.1}$$

where x is the generator output and s is the context. The generator objective function depends on the conditional output of the discriminator and non-conditional one. The DAMSM loss is responsible for the semantic attention between the text features and image features. The last equation comes from the mode-seeking function which involves working with the final generated images and the initial noise functions.

$$\begin{aligned} L &= \sum_i L_{G_i} + \lambda_1 L_{G_{CA}} + \lambda_2 L_{DAMSM} + \lambda L_{mode-seek} \\ L_{G_i} &= -\frac{1}{2}\left[E_{x\sim p_{G_i}} log(D_i(x)) + E_{x\sim p_{G_i}} log(D_i(x,s))\right] \\ L_{CA} &= D_{KL}(N(\mu(s), \Sigma(s))||N(0,I)) \\ L_{mode-seek} &= \max_G \left[\frac{D(G(c,z_1), G(c,z_2))}{D(z_1,z_2)}\right] \end{aligned} \tag{3.2}$$

where λ_1 is the coefficient for the Conditional Augmentation loss, λ_2 is the coefficient for the DAMSM loss [7] , λ is the coefficient for the mode-seeking loss $L_{mode-seek}$ and z_i are the noise vectors. The λ_1 and λ_2 are fixed and present as in the original DMGAN objective function for the corresponding loss.

3.1.2 Implementation

For text embedding, a pre-trained bidirectional LSTM text encoder is employed by AttnGAN[7] and their parameters are fixed during training. Each word feature corresponds to the hidden states of two directions. The text, image and memory features are set to be 256, 64 and 128 respectively. The hyperparameters are set to be λ_1 = 1 and λ_2 = 5 for the CUB dataset and λ_1 = 1 and λ_2 = 50 for the Coco dataset. All networks are trained using ADAM optimizer[9] with batch size 8 for Coco dataset and 5 for CUB .Other parameters of β_1 = 0.5 and β_2 = 0.999 as given . The learning rate is set to be 0.0002. The DM-GAN model was trained with 600 epochs on the CUB dataset and 120 epochs on the Coco dataset. The models were saved every 50 epochs for the Birds dataset and every 5 epochs for the Coco dataset. The setup was run on a single Tesla v100 GPU in the virya cluster of Concordia University. The training of the dataset on 8855 CUB bird training images took 3 days and the same on 80k Coco scenes took 12 days. The timing was the same for both the parameter values of λ. The values for the λ_1, λ_2 are fixed through all runs according to the original DMGAN parameters [8]. The code for this task was constructed from the original architecture of DM-GAN [8] and mode-seeking GAN [11].

3.2 Results

The datasets used for the training of the GAN were the Caltech Birds and the Microsoft Coco. The DM GAN network was trained on two values of the λ for each of the Datasets. The trained models generate 30000 images of the validation dataset separate from the training data. The generated images were used for generating the FID metric score. A small set of generated training images per epoch was displayed on a Tensorboard visualization. The epoch of first generation of the best quality images was noted and the saved models closer to that epoch was taken for generation of test images. The model was allowed to run for given number of epochs in the Implementation.

3.2.1 Image Quality

From the figure 3.2, the images generated from the new DMGAN have better quality in terms of finer features and better characteristics of the birds for the CUB dataset than the original DMGAN images. It can also be seen that the bird is well separated from the background and there are fewer instances of having a double head. Also in many images, the background is clearly described and can be distinguished very well from the bird. There are few images with well distinguishable colours and brighter to correlate well with the description. In the case of the Coco dataset in the figure 3.3, more distinguishing features can be seen in the same way as birds with better clarity. By increasing the value of λ from1 to 1.5, there is an improvement in the image quality and better

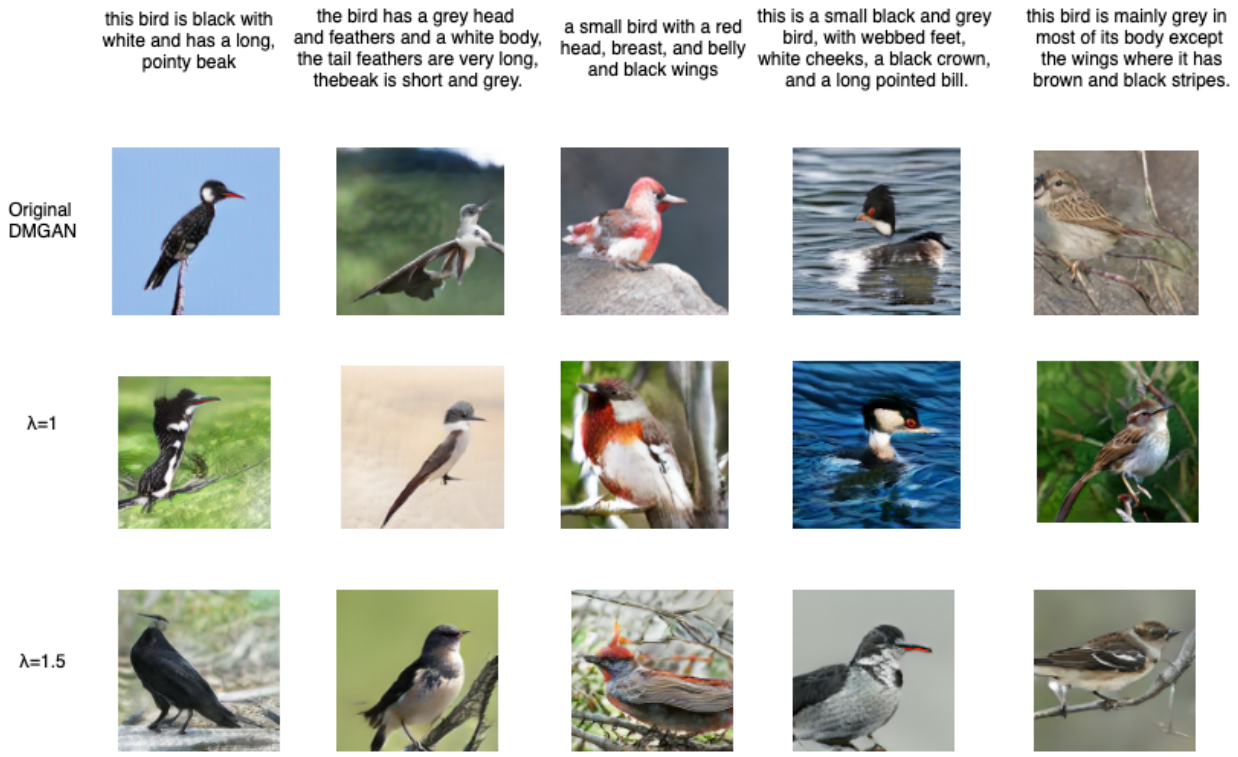


Figure 3.2: Results of our models on the CUB dataset with the values of the λ coefficient and their respective images

Figure 3.3: Results of our models on the Microsoft Coco dataset with the values of the λ coefficient and their respective images

Architecture	CUB Dataset	Coco Dataset
StackGAN++[6]	35.11	33.88
AttnGAN[7]	23.98	35.49
DMGAN[8]	16.09	32.64
OPGAN[41]	-	24.7
Ours @ $\lambda = 1$	14.27	24.3
Ours @ $\lambda = 1.5$	13.91	26.01

Table 3.1: Evaluation results (FID score) on CUB-200-2011 and Coco dataset with two different values of λ. Our FID is better than DMGAN.

Architecture	CUB Dataset	Coco Dataset
StackGAN++[6]	3.84 ± 0.06	4.77 ± 0.06
AttnGAN[7]	4.36 ± 0.03	25.89 ± 0.47
DMGAN[8]	4.75 ± 0.07	30.49 ± 0.57
Ours @ $\lambda = 1$	4.6 ± 0.06	28.55 ± 0.53
Ours @ $\lambda = 1.5$	4.66 ± 0.06	32.66 ± 0.71

Table 3.2: Evaluation results (Inception score) on CUB-200-2011 and Coco dataset with two different values of λ. Ours is better than DMGAN and higher λ has better results. Inception score values and standard errors are calculated accordign to the description given in the section 2.4.1

colours in the image. Thus, this training method produces better distinctive features in the images by improving contrast.

3.2.2 Comparison with other models

The objective of this work is to understand the importance of the mode seeking a term in the loss function by tweaking its amplitude. FID score decreased from 16.09 for the original DMGAN network to 14.27 for the $\lambda = 1$ while it reduced further to 13.93 for $\lambda = 1.5$. The Coco Dataset saw a significant change as the value of the FID score was 24.3 for the $\lambda = 1$, a 25.5% decrease from the original DMGAN FID and 1.6% from the OPGAN FID. But the dataset gave a higher score for $\lambda = 1.5$ with a value of 26.01 beating the basic model of DMGAN. It is a bigger improvement on the previous AttnGAN model with a 42% decrease in FID on the CUB dataset and a 31.5% decrease in FID on the Coco dataset. In summary, it can be said that the DMGAN network can be trained with the new objective function to achieve the best experimental results as compared to the original loss scheme. Table 3.1 compiles the values of FID scores with changing parameters with the FID values of the previous architectures.

In the case of the Inception score, there is a downward trend observed for the CUB dataset where the inception score slightly shifted downward. It increased as with the rise in the λ factor

as can be seen in the values of the core. The increase in score was significant for the higher λ on the coco dataset where there is an increase of 4 points and even more than the original DM-GAN. Initially, the score decreased for the Coco dataset for the $\lambda = 1$ case.

The Inception score measures the conditional classification of images to their classes and diversity. An increase in the score depicts an increase in diversity which can be seen in higher datasets but an increase in FID at the same time also means that by increasing diversity, it is moving away from the real data distribution. For higher classes dataset with fewer images, higher diversity might mean a brunt on the conditional image classification to their classes leading to their balanced nature and less deflection of inception score from the original DM-GAN.

3.2.3 Analysis

There is a variation in FID with an increase of λ factor. The λ factor is responsible for the strength of the mode seeking loss function in the overall loss expression and thus, it influences the training of the network. It manages the intensity of the mode collapse loss function in the loss term. Table 3.1 shows that the results have different FID scores on different values of parameters. The FID score decreases with the increase in the amplitude of the mode-seeking term for the Coco dataset but it decreases for the CUB dataset. There might be some new modes that resulted in the new images having different shades and better quality. Therefore, a substantial amount of diversity in the network is achieved and thus, the architecture could generate better images for descriptions.

Qualitatively, an increase in λ improves the image quality with the increase in clarity. But it also increases the strength of the mode-seeking term which depletes the importance of the DAMSM loss and the conditional loss in the original loss function. Some of the effects can be seen in the first example of the bird in the figure 3.2. In the example, the bird being black is magnified better in the higher λ and other features are rendered less significant. In the fourth example of the figure 3.3, the bus being white is not evident in the $\lambda = 1.5$.

Though the architectures improve in their FID, their image quality might go for a spin and so it's necessary to observe their images. The DAMSM loss is the function holding up the semantic relationship between image and text. The FID is a metric measuring the quality and diversity of the images which definitely increases due to the introduction of the mode-seeking loss function. Therefore, the value of $\lambda = 1$ is kept for further experiments.

3.2.4 Discussion

Higher Inception scores and lower FID values are better for the multimodality as they show that there is an increase in the GAN output modes and quality of the generated images. Increase in multimodality or output modes shows that the model might have access to better modes of images

like different orientations, better contrasting and vivid backgrounds. If we check the images of Coco from the figure 3.3, we can see that the $\lambda = 1.5$ images show a great contrast in their image quality and sharp features as well but they deviated a bit from their original text descriptions as explained in the Analysis. This can be explained by their improvement in the Inception scores as it checks for diversity and quality. But if we check the FID value for the same $\lambda = 1.5$, we get an increase in FID than $\lambda = 1$ making it an image distribution less comparable with the real data distribution.

CHAPTER 4

REFINED ATTENTION IN AttnGAN

The second approach used in the text to image synthesis is the new architecture for a network. The network used for this model was the AttnGAN model with modified attention. The attention modification was thought about with a vision to improve the attention of the network to the correct words. Improvement of the attention would lead to a better understanding of the vector and thus, generate better image features.

The attention procedure of e-AttnGAN was used to increase the attention in the original AttnGAN network. It consists of two attentions- one for the word-embeddings and another for the sentence embeddings or the global vector. In the e-AttnGAN network, the output from the two attentions is concatenated before presenting to the up-sampling block of the network. Initially in the AttnGAN, they use the attention output and the original image feature vector for the up-sampling block.

Suppose the proposed attentional generative network has m generators (G_0, G_1, ..., G_{m-1}), which take the hidden states (h_0, h_1, ..., h_{m-1}) as input and generate images of small-to-large scales (x_0, x_1, ..., x_{m-1}).

$$
\begin{aligned}
&h_0 = F_0(z, F_{ca}(e));\\
&h_i = F_i(h_{i-1}, F_i^{attn-new}(w, h_{i-1}, h_0))i = 1, 2, ..., m1;\\
&F_i^{attn-new}(w, h_{i1}, h_0) = F^{aoa}[F_i^{attn}(e, h_{i-1}), F_i^{attn}(w, h_0)]\\
&x_i = G_i(h_i)
\end{aligned}
$$

where F_{ca} is the conditional augmentation module, z is the random uniform vector, e is the sentence embedding or the global vector, w is the word embedding vector, F_i are the upsampling modules, $F_i^{attn-new}$ is the combined new refined attention module and F^{attn} is the original attention network.

This architecture merges two attentions through an Attention-on-Attention layer improving the semantic relation of the text and the image. The result of the AOA module is concatenated with the original image feature vector before passing through the upsampling block. Thus, I construct

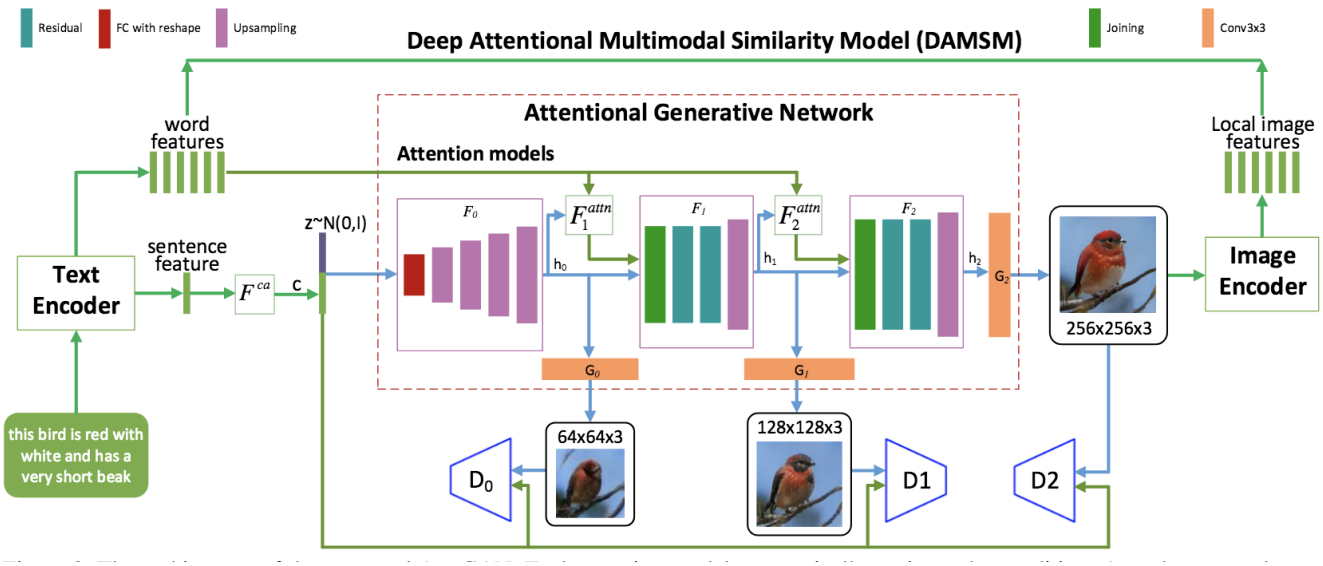


Figure 4.1: AttnGAN original architecture [7] and the F_1^{attn} and F_2^{attn} is replaced by the architecture in figure 4.2

a new attention module comprising of two attentions joined by an AOA block. The output of the model is extracted in 3 resolutions- 64X64, 128X128 and 256X256.

The AOA layer provides better attention on the attended features of the attention GAN and it merges the attention coming through the word embeddings with the attention on image features through the sentence embeddings.

The objective function for the architecture is the same as the AttnGAN as it has already the loss function responsible for the training of the Attention network- the Deep Attention Multimodal Similarity Model objective.

$$L_{refined} = L_{attnGAN}$$

The model is trained with the same parameters as the original AttnGAN model.

4.1 Merging Improve Attention with Mode-Seeking Loss

The third short experiment is to apply both methods to the same problem. The work is implemented on an AttnGAN with the changed attention module. The objective function is appended with the new function based on the mode-seeking loss $L_{mode-seek}$.

$$L_{new_refined} = L_{attnGAN} + L_{mode-seek}$$

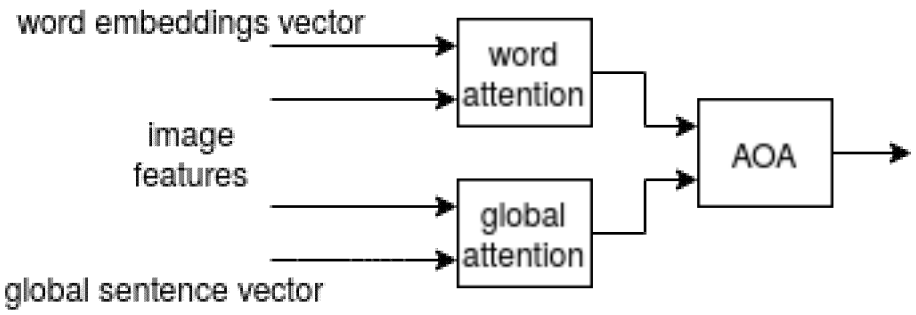


Figure 4.2: Refined attention uses this new architecture in place of the word attention in the AttnGAN network in the original network

The mode-seeking loss tries to improve the diversity of the output modes.

4.1.1 Implementation Details

The value of λ=1 is set for the coefficient of the mode-seeking function as concluded in the first work on DM-GAN. The other parameters for the conditional augmentation objective and the DAMSM loss are not changed. The model is trained with a single core in the Concordia cluster for 600 epochs on the CUB dataset and 120 epochs on the Coco dataset. The total training time for CUB dataset was 3 days while Coco dataset took 9 days to train. The model was constructed with the help of the codes taken from the AttnGAN original architecture [7], e-AttnGAN [9] and Attention on Attention [10]

4.2 Results

The result is extracted after the training of the 88000 images on both the datasets in the AttnGAN. There are 2 stages of multi-stage GAN, each for the 128X128 and the 256X256 resolution images. The excerpt tries to delve into the ablation study of the application of functions, comparison with the parent model-the AttnGAN and then the image quality of the generated testing images. The training is done till 100 epochs in the case of the Coco dataset and 600 epochs in the case of the Birds dataset.

4.2.1 Image Quality

In terms of the images generated by the refined attention function, there is a slight improvement in terms of the images from the AttnGAN for the bird's dataset. There are more instances that the object is well separated from the background and the object is complete in both cases like the

Figure 4.3: Birds images for different text descriptions for original and Refined GAN architectures

Figure 4.4: Coco images for different text descriptions for refined GAN architectures

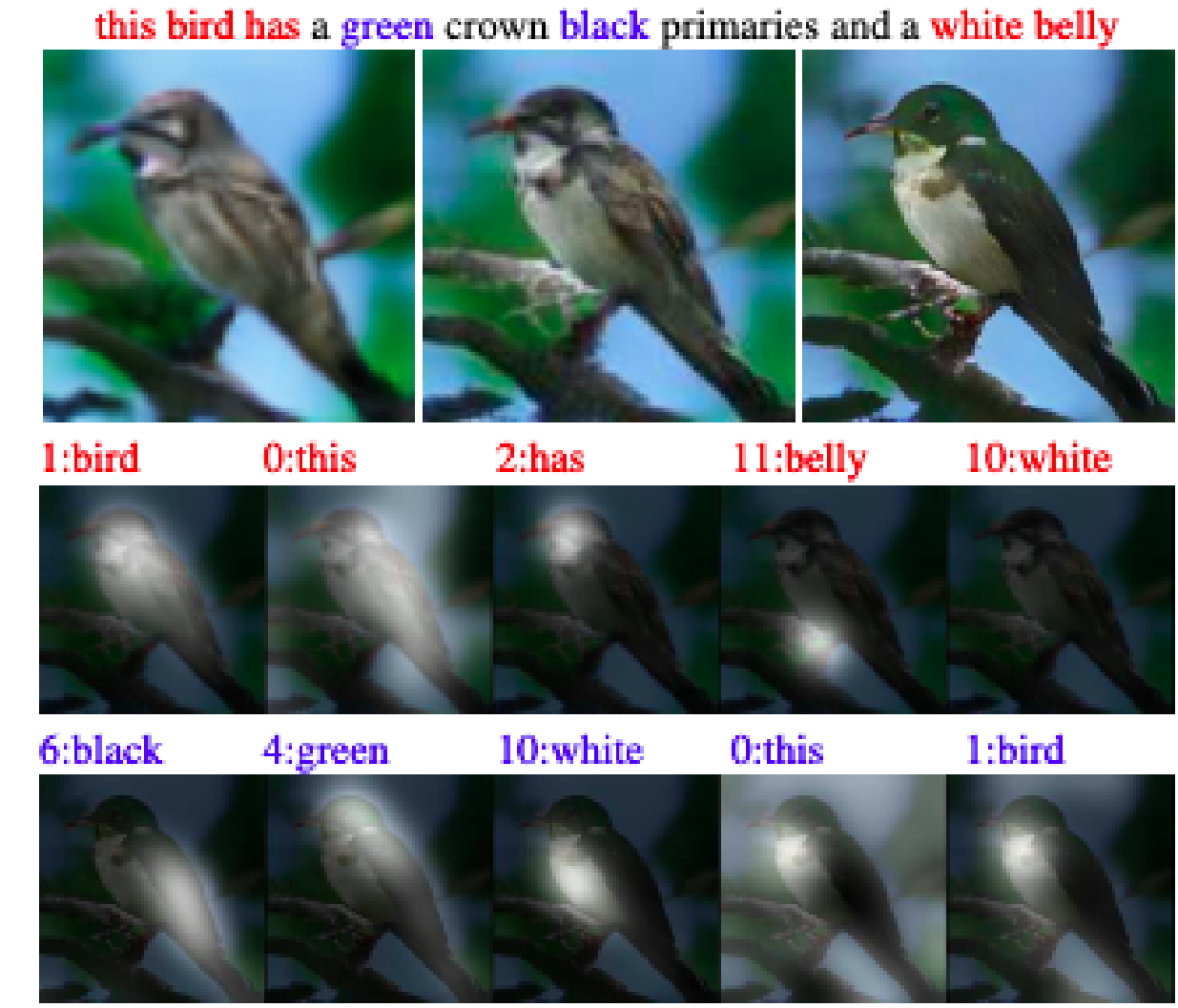


Figure 4.5: AttnGAN bird (original architecture) - Intermediate results of the bird as it passes through the GAN and the attention results

original AttnGAN. It is easier to say that the improved attention has brought about more elaborate features within birds where it concentrates on the small parts of the sentence and thus executes the generation of a better image. The attention effect can be seen in the case of Coco results as the wooden flooring and the home decor clearly in the images. Thus, the attention improvement has caused a better look at the detail of the image and the mode-seeking function has led to the improvement of the image quality by strengthening the inferior modes of the network.

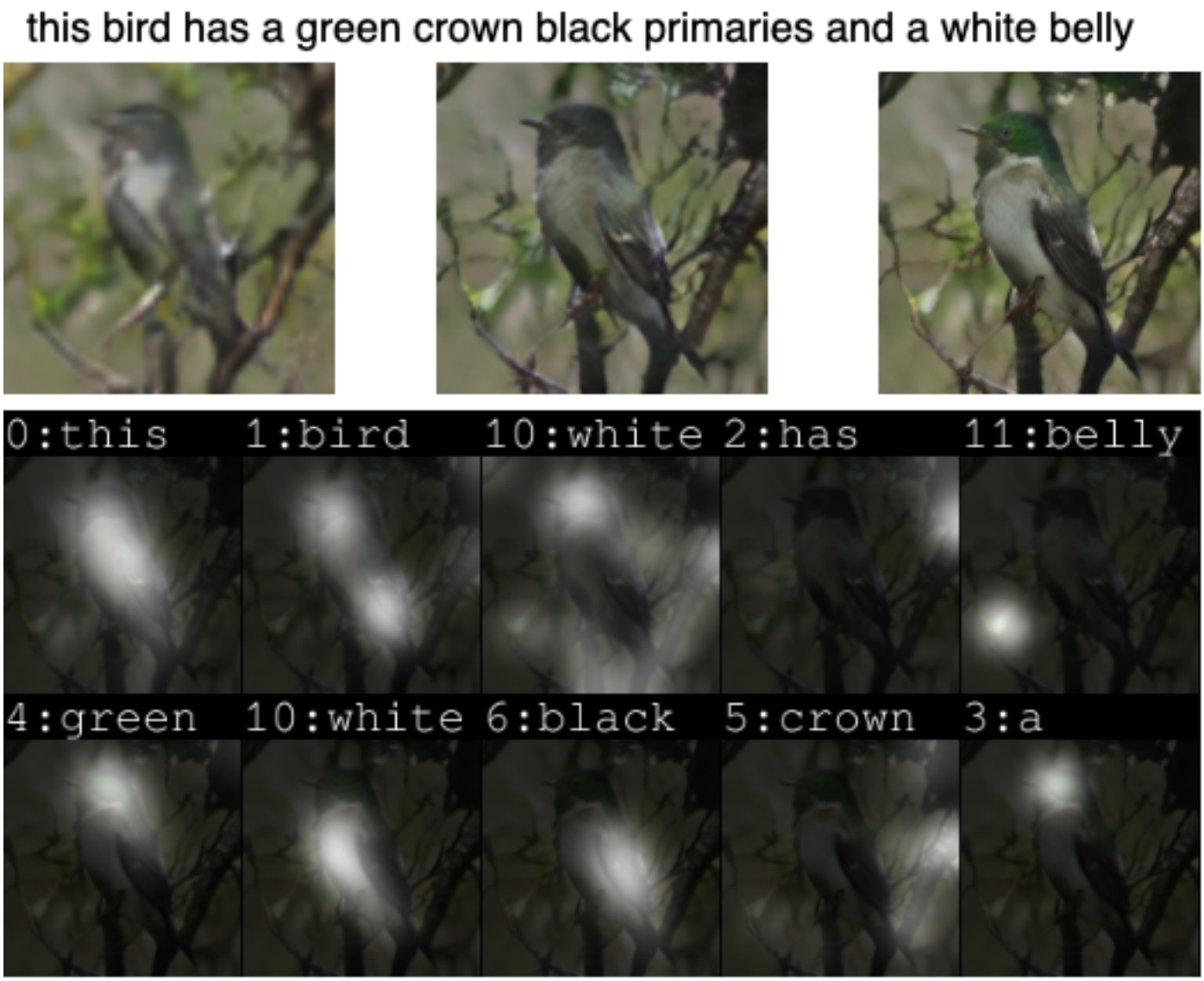


Figure 4.6: Refined Attention bird- Intermediate results of the bird as it passes through the GAN and the attention results

Figure 4.7: Refined Attention with mode-seeking function bird - Intermediate results of the bird as it passes through the GAN and the attention results

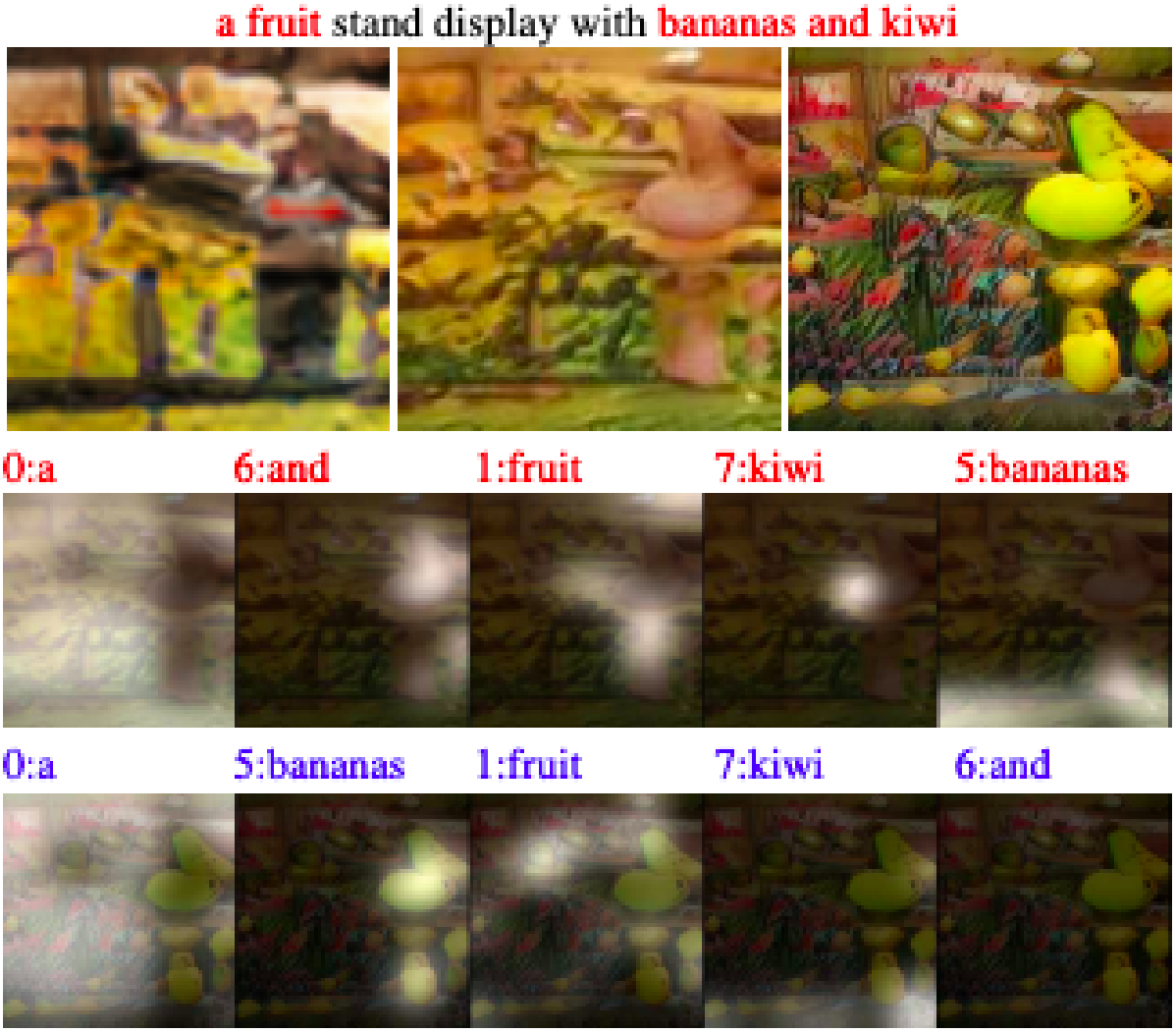


Figure 4.8: AttnGAN(original architecture) coco- Intermediate results of the coco as it passes through the GAN and the attention results

Figure 4.9: Refined Attention coco- Intermediate results of the coco as it passes through the GAN and the attention results

Figure 4.10: Refined Attention with mode-seeking function Coco- Intermediate results of the coco as it passes through the GAN and the attention results

Architecture	CUB Dataset	Coco Dataset
StackGAN++[6]	35.11	33.88
AttnGAN[7]	23.98	35.49
Refined Attn	23.78	31.68
Refined Attn + mode-seeking	20.73	27.64

Table 4.1: Evaluation results (FID score) on CUB-200-2011 and Coco dataset with two different architectures of refined attention and refined attention with mode-seeking

Architecture	CUB Dataset	Coco Dataset
StackGAN[6]	3.84 ± 0.06	4.77 ± 0.06
AttnGAN[7]	4.36 ± 0.03	25.89 ± 0.47
Refined Attn	4.32 ± 0.19	24.32 ± 0.58
Refined Attn + mode-seeking	4.33 ± 0.15	26.23 ± 0.42

Table 4.2: Evaluation results (Inception score) on CUB-200-2011 and Coco dataset with different architecture variant and one mode-seeking addition.

4.2.2 Ablation study

Attention weights are pulled from the original bird to analyze the development of the attention in the network. The less relevant attentions are equated to 0 while the prominent 5 are kept at every stage in the AttnGAN process. These are arranged in the order of priority as they indicate the way a model processes an image. The images from 3 generators and the attentions are arranged in the figures above for each of the original AttnGAN, refined AttnGAN and refined AttnGAN with mode-seeking function. The generated images improve in terms of quality and clarity of the objects as one goes from left to right by increasing resolutions.

It can be seen that the refined attention on the bird does focus on the birds part of the image and its features like belly being white and crown being green. The attention focuses on drawing an image of the bird which was initially painted yellow in the upper belly portion later painted to white. Also in the case of the mode-seeking term in the figure 4.7, there is an improvement in the image quality and also, a different configuration of the bird can be seen due to its diversity.

In the case of the Coco dataset, it can be seen that the images are changing with the help of renewed attention. In the figure 4.9 , Bananas can be seen as well as kiwis position is defined with the help of the extracted attention output. It tries to display a fruit stand with the many fruits being placed on the platform. In the figure 4.10, a green pigment is at the bottom of the second generator image. It can be seen that the placement of the kiwis and the big bananas on the cart is in a different view than the refined attention.

4.2.3 Comparison with other models

The FID for the bird's dataset decreased from 23.98 to 23.78 for the refined attention module. The difference is less significant but it can be seen in the experiment with mode seeking function. The FID decreases to 20.73 in the most refined case. Similarly, there is an 4 point drop in the case of the Coco dataset where the FID decreased from 35 to 31.2 and then the mode-seeking function reduced the FID to 27.6. The table presented shows an analysis of the FID scores and the Inception scores for both the datasets. The coefficient of the mode-seeking function is selected to be 1 as it is the value concluded in the mode-seeking loss.

The inception score for the new architectures can be seen to be staying constant with a slight dip for the CUB dataset. The Coco dataset responds well to the inception score with an initial dip and then a rise due to the mode-seeking function. The quality improvement with the mode-seeking function is evident from the images as well as the metric scores. The Refined attention attends better to the details improving the granularity of the image as it concentrates better on the word description. The mode-seeking function improves the clarity of the image and thus, their combination paves the way for a clear and granular generation of images from text.

CHAPTER 5

SUMMARY

This chapter discusses the summary of the observations and the future work in regards to the work.

5.1 Conclusion

The research aimed at approaching the topic of the text to image synthesis from multiple directions. The mode-seeking function modified the loss function by appending it to the original loss function of the generator. The refined attention architecture modifies the attention architecture by introducing a new method and using the Attention on Attention(AOA) architecture. The original architecture of Dynamic Memory GAN was used for loss modification while the refined attention, as well as mode-seeking, was applied to the AttnGAN architecture. The results for the GAN were evaluated according to the basic metrics of FID and Inception Score. Inception Score evaluates the quality and diversity of the images while the FID checks also the data distribution. Good results are signified by low FID scores and high Inception scores. GAN architectures can also be verified by human evaluation.

The objective function for the generator was changed by the mode-seeking function as it tried to apply two random uniform vectors to a GAN architecture to strengthen the hidden modes of the Generator output. Its application on the GAN increases the output modes also improving the diversity and the clarity of the images. The significance of the mode-seeking term was verified by changing the magnitude of the objective function. The FID metrics increase signifying the increased diversity and clarity as to the mode-seeking term gains. The increase in the magnitude of mode-seeking might affect the DAMSM term responsible for the semantic attention between text and image.

The new architecture of Refined attention in the AttnGAN is tested with the standalone mode as well as its loss modification with the mode-seeking term. By improving the attention, the attention network improves the image features with more important words resulting in the generation of a better-attended image for the word features. The mode-seeking adds minor modes improving the

quality and diversity of the image. It is manifested in the further decrease of FID and increase of Inception score. Coco dataset had the maximal change in FID and Inception score with the introduction of mode-seeking function. The increased attention improves the FID as it tries to bring the generated image to focus more on the sentence and it comes closer to the real image.

Both the qualitative and quantitative measures show that aforementioned modifications improve the quality of the images generated. Attended images have their image well conditioned with the text and improved loss works well with the diversity of images. We have submitted a paper entitled by "Refining Attention in AttnGAN using Attention on Attention" to ICIP 2021 [42]. Also, a paper entitled by "Improving Training of Text-to-image Model Using Mode-seeking Function" [12] is present on the arxiv website regarding the first approach.

5.2 Future Work

There are a few places where the work could be improved or modified in the future.

5.2.1 Architecture

Dynamic Memory-GAN and AttnGAN dealt with improving the semantic relation between the image and the text. DM-GAN has a memory model to induce text dependencies in the image while AttnGAN uses attention to improve image features. There could be new architectures that focus on the semantic relationship and try to strengthen it as the semantic relationship is the crux of the text to image synthesis.

5.2.2 Loss function

DAMSM loss tries to gauge the similarity between the generated images and the text descriptions. Application of improved techniques like the Mirror GAN[43] which combines the image captioning as the generator objective function.

5.2.3 Metrics

Metrics for the output quality of the images could be changed as the FID and Inception score check for the quality and diversity but they don't take into account the semantic information of the images. There is a necessity for a metric more suited to the problem statement which also checks for the correlation between the text and the image output. Also, there are researches for better metrics of GAN.

BIBLIOGRAPHY

[1] Colah, "Lstm networks: https://colah.github.io/posts/2015-08-understanding-lstms/," 2015.

[2] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.

[3] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*, pp. 740–755, Springer, 2014.

[4] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv preprint arXiv:1605.05396*, 2016 Github: https://github.com/reedscot/icml2016.

[5] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proceedings of the IEEE international Conference on Computer Vision*, pp. 5907–5915, 2017 Github : https://github.com/hanzhanggit/StackGAN.

[6] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan++: Realistic image synthesis with stacked generative adversarial networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1947–1962, 2018 Github: https://github.com/hanzhanggit/StackGAN-v2.

[7] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1316–1324, 2018 Github : https://github.com/taoxugit/AttnGAN.

[8] M. Zhu, P. Pan, W. Chen, and Y. Yang, "Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5802–5810, 2019 Github: https://github.com/MinfengZhu/DM-GAN.

[9] K. Emir Ak, J. Hwee Lim, J. Yew Tham, and A. Kassim, "Semantically consistent hierarchical text to fashion image synthesis with an enhanced-attentional generative adversarial network," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0, 2019 Github: https://github.com/k-eak/e-AttnGAN.

[10] L. Huang, W. Wang, J. Chen, and X.-Y. Wei, "Attention on attention for image captioning," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4634–4643, 2019 Github: https://github.com/husthuaan/AoANet.

[11] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, "Mode seeking generative adversarial networks for diverse image synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1429–1437, 2019 Github: https://github.com/HelenMao/MSGAN.

[12] N. Bhise, Z. Zhang, and T. D. Bui, "Improving text to image generation using mode-seeking function," *arXiv preprint arXiv:2008.08976*, 2020.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[14] X. Li, X. Yin, C. Li, P. Zhang, X. Hu, L. Zhang, L. Wang, H. Hu, L. Dong, F. Wei, *et al.*, "Oscar: Object-semantics aligned pre-training for vision-language tasks," in *European Conference on Computer Vision*, pp. 121–137, Springer, 2020.

[15] K. Desai and J. Johnson, "Virtex: Learning visual representations from textual annotations," *arXiv preprint arXiv:2006.06666*, 2020.

[16] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *Annual Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2672–2680, 2014.

[17] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[18] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 172–189, 2018.

[19] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1125–1134, 2017.

[20] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *Proceedings of the European conference on Computer Vision (ECCV)*, pp. 35–51, 2018.

[21] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, pp. 700–708, 2017.

[22] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," in *Advances in Neural Information Processing Systems*, pp. 465–476, 2017.

[23] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2223–2232, 2017.

[24] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8798–8807, 2018.

[25] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with lstm," in *1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470)*, vol. 2, pp. 850–855 vol.2, 1999.

[26] S. E. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee, "Learning what and where to draw," in *Advances in Neural Information Processing Systems*, pp. 217–225, 2016.

[27] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, pp. 2234–2242, 2016.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[29] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, pp. 6626–6637, 2017.

[30] S.-C. Wang, "Artificial neural network," in *Interdisciplinary Computing in Java Programming*, pp. 81–100, Springer, 2003.

[31] Prabhu, "Understanding of convolutional neural network (cnn) — deep learning Medium," 2018.

[32] L. R. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications/ CRC press*, vol. 5, 2001.

[33] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 6, no. 02, pp. 107–116, 1998.

[34] R. Pascanu, T. Mikolov, and Y. Bengio, "Understanding the exploding gradient problem," *CoRR, abs/1211.5063*, vol. 2, p. 417, 2012.

[35] Y. Yao and Z. Huang, "Bi-directional lstm recurrent neural network for chinese word segmentation," in *International Conference on Neural Information Processing*, pp. 345–353, Springer, 2016.

[36] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[37] G. W. Lindsay, "Attention in psychology, neuroscience, and machine learning," *Frontiers in Computational Neuroscience*, vol. 14, p. 29, 2020.

[38] J. Weston, S. Chopra, and A. Bordes, "Memory networks," *arXiv preprint arXiv:1410.3916*, 2014.

[39] Adrian, "Adrian colyer blog - memory networks: Adrian blog," 2016.

[40] M.-E. Nilsback and A. Zisserman, "Delving deeper into the whorl of flower segmentation," *Image and Vision Computing*, vol. 28, no. 6, pp. 1049–1062, 2010.

[41] T. Hinz, S. Heinrich, and S. Wermter, "Semantic object accuracy for generative text-to-image synthesis," *arXiv preprint arXiv:1910.13321*, 2019.

[42] N. Bhise, A. Krzyzak, and T. D. Bui, "Refining attngan using attention on attention network," *Submitted to IEEE International Conference on Image Processing*, 2021 (ICIP 2021).

[43] T. Qiao, J. Zhang, D. Xu, and D. Tao, "Mirrorgan: Learning text-to-image generation by redescription," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1505–1514, 2019.