CHAPTER 1
INTRODUCTION

1.1 Background

The process of generating photorealistic image from the text is one of the important problems and is the emerging technology and has various tremendous applications including photo editing, computer aided design, interactive computational graphic design, image fine-tunning etc. The process of generating truly plausible looking image has not been easy. The majority of various advanced methods do not produce photorealistic details that describes given text description. The Generative Adversarial Network (GAN) have shown promising result in synthesizing the real world images. Conditional GANs are able to generate the images that are extremely related to the text meanings but it is very hard to train GAN to generate high-resolution photorealistic images from text descriptions. Here adding more upsampling layers in the state-of-art GAN models for generating high resolution images results in training instability and produces nonsensical outputs. The main difficulty for generating high-resolution images by GANs is that supports of natural image distribution and implied model distribution may not overlap in high dimensional pixel space. This problem is extremely severe when the image resolution increases.

In the context to how human draws, the problems of text to photorealistic image is decomposed into two sub-problems with the use of Stacked Generative Adversarial Networks. Low resolution images are first generated by Stage-I GAN and on the top of Stage-I GAN the Stage-II GAN is stacked to generate realistic high-resolution images conditioned in Stage-I results and text description.

For the text to image generation the limited numbers of text-image pairs often results in sparsity in the text conditioning and such sparsity makes it difficult to train GAN. Thus conditioning augmentation technique is used to encourage smoothness in the latent conditioning. This method increases the diversity of the synthesized images.

The basic concept of the project is to convert the given text description into vectors and generating the low quality images from the given vectors using GAN. And similarly in the second stage using the text description and low quality image from first step high quality images will be produced. This types of project will be helpful in designing of various items and similarly can be used in various computer aided design process too.

1.2 Statement of Problem

The problem that this project attempt to solve is concerned with the generation of high-resolution photorealistic images from the given text description. The process is carried out into two sub-problems. Firstly, the text is converted to low quality images and secondly to the high-resolution image.

We have discussed and reviewed some similar types of projects research in this area of text to image synthesis. Generative image modeling is one of the fundamental problem in the computer vision Deep learning techniques. Various models such as autoregressive models that utilized neural networks to model the conditional distribution of the pixel space have also generated appealing images. Variational Autoencoders formulated the problem with probalistics graphical models whose goal was to maximize the lower bound of data likelihood. But the main problem is the training instability which makes it hard for GAN models to generate high resolution images various techniques has been proposed to stabilize the training process and generate the compelling results. Most of the methods utilize simple conditioning variables such as attributes or class labels. This is also work conditioned on image to generate images, including photo editing and super-resolution. But super-resolution can only add limited details to low resolution images and cannot correct large defects.

1.3 Objectives

The general objectives of the project is to convert text to photorealistic image. The list of specific objectives to attain general objectives are:

- To generate low quality image from given text description.
- To generate high resolution image from low quality image.

1.4 Scope

- Photorealistic images are generated form the given text descriptions which will be helpful in designing various items, interactive graphics design & animation and can be used in various computer aided design process.

1.5 Limitation

- Random Noise is also generated for any text description.
- Only capable of generating a photo realistic image of birds only.

CHAPTER 2
LITERATURE REVIEW

On the context of reviewing the similar type of past project we have studied that there are various process introduced for the text to photorealistic image synthesis. Traditionally, people have been using computer algorithms to generate artistic paintings. Those generated work are usually random, abstract, or full of patterns due to the limitations of the algorithm. It's hard to customize those artwork based on different people's needs. Recently, with the development of computer vision, people have been using neural networks to generate artistic images from real world images.

Convolutional neural network (CNN) based method to transfer styles from a painting to a photo [1]. This method has been shown to generate artwork with high perceptual quality. And it can generate very high-resolution images which most of the other neural networks failed to do. However, this still requires a photo as input so generated artwork, and the content of the output image is fully based on input photo. Also, the style photo and the image photo may not work together very well because the color in the content image may affect the output of the style transfer. Causing the generated image to lose some artistic effects.

While deep style transfer provided a way to generate artwork using provided content image, generative adversarial networks (GANs) provided another way of generating highly compelling images.A class of GAN called deep convolutional generative adversarial networks (DCGANs) has been introduced [2]. Their work shows that GAN is able to learn a hierarchy of representations from object parts to scenes. Additionally, GAN can be used to generate near photo-realistic images of bedrooms. However, DCGAN generated images have low resolution and still suffer from different kinds of defects.

Generative image modeling is a fundamental and emerging problem in computer vision. There has been remarkable progress in this direction with the emergence of deep learning techniques. Variational Autoencoders formulated the problem with probabilistic graphical models whose goal was to maximize the lower bound of data likelihood [3]. Autoregressive models that utilized neural networks to model the conditional distribution of the pixel space have also generated appealing synthetic images. Recently, Generative Adversarial Networks (GAN) have shown promising

performance for generating sharper images [4]. Generative Adversarial Networks (GAN) composed of two models that is alternatively trained to compete with each other. The generator G is optimized to reproduce the true data distribution P_{data} by generating images that are difficult for the discriminator D to differentiate from real images. Meanwhile, D is optimized to distinguish real images and synthetic images generated by G. Overall, the training procedure is similar to a two-player min-max game with the following objective function,

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

Where x is a real image from the true data distribute on P_{data}, and z is a noise vector sampled from distribution P_z. Conditional GAN is an extension of GAN where both the generator and discriminator receive additional conditioning variables c, yielding G(z, c) and D(x, c). This formulation allows G to generate images conditioned on variables c.

The training instability makes it hard for GAN models to generate high-resolution images. So the stacked GAN is used to reduce the training instability and generate the high-resolution image. StackGAN bridged resolution gap between text to image synthesis GAN and models like PPGN [5]. In their work, they proposed a novel two staged approach for text to image synthesis. The first stage network is able to generate low-resolution plausible images from text descriptions. The second stage network then takes the generated image from the first stage network, and then refine the image to generate a more realistic and much higher resolution image. In their experiments, they were able to generate 256x256 high-resolution images from just a text description.

CHAPTER 3
METHODOLOGY

3.1 Background

Synthesizing high-quality images from text descriptions is a challenging problem in computer vision and has many practical applications, including photo editing and computer aided design. The various samples which are generated using text to image approaches can roughly reflect the meaning of the given text descriptions, but they fail to contain necessary details and vivid object parts and makes it difficult to train the GAN to generate high resolution photorealistic images.

Stacked Generative Adversarial Networks (StackGAN) is used to generate an image size of 256*256 photo-realistic image with given text description. We divide the hard problem into sub-problems through a sketch-refinement process to minimize the complexity of the project. In this sub-divided problems we have Stage-I GAN which sketches the primitive shape and the colors of the object based on the given text description, which generates the low quality image size of 64x64 pixels. The Stage-II GAN than takes the result of the Stage-I GAN and the text description as input, and generates high-resolution images with photo-realistic details. Stage –II GAN will be able to rectify defects in Stage-I results and add compelling details with the refinement process.

A new Conditioning Augmentation technique is used to stabilize the conditional GAN training and also improves the diversity of the generated image sample. We have use additional conditioning variable, the sample variables is used from an independent Gaussian distribution, where the mean and diagonal covariance matrix are the functions of the text embedding. The Conditioning Augmentation yields more training pairs given a small number of image text pairs, and thus encourages robustness to small perturbations along the conditioning manifold. The randomness introduced in the Conditioning Augmentation will be beneficial for modeling text to image translation as the same sentence usually corresponds to objects with various poses and appearances.

In other hand the discriminator must classify individual elements as being fake or real. The discriminator generates labels (real/fake) for each element in the batch. The loss functions are computed based on those labels. Elements are fed to the discriminator in batches of the same. This is because the fake data batch is directly generated by the

generator as part of the same computational graph. This is so to be able to propagate gradients of the generator parameters through the discriminator.

3.2 Tools and Platform

Tools

- PyCharm 2018.1
- Python 2.7

Used Python Libraries

- Tensorflow 1.0.0
- Torch
- prettytensor 0.7.4
- progressbar 2.5
- torchfile 0.1.0
- easydict 1.7
- pandas 0.23.0
- python-dateutil 2.7.3

Platform

- Linux (Ubuntu 18.04)

3.3 Stacked Generative Adversarial Network

Generative Adversarial Networks (GAN) composed of two models that is alternatively trained to compete with each other. The generator G is optimized to reproduce the true data distribution P_{data} by generating images that are difficult for the discriminator D to differentiate from real images. Meanwhile, D is optimized to distinguish real images and synthetic images generated by G. Overall, the training procedure is similar to a two-player min-max game with the following objective function,

$$\min_G \max_D V(D,G) = E_{x\sim P_{data}}[\log D(x)] + E_{z\sim P_z}[\log(1 - D(G(z)))] \text{----------(1)}$$

Where x is a real image from the true data distribute on P_{data}, and z is a noise vector sampled from distribution P_z. Conditional GAN is an extension of GAN where both the generator and discriminator receive additional conditioning variables c, yielding G(z, c) and D(x, c). This formulation allows G to generate images conditioned on variables c.

To generate high-resolution photorealistic image from the text description, StackGAN decomposes the text-to-image generative process into two stages:-

- **Stage-I GAN:** it sketches the primitive shape and basic colors of the object conditioned on the given text description, and draws the background layout from a random noise vector generating low-resolution image.
- **Stage-II GAN:** it corrects defects in the low-resolution image and completes details of the object by reading the text description again, producing a high resolution photo-realistic image.

3.3.1 Conditioning Augmentation

As shown in Fig 1, the text description is first encoded by an encoder, yielding a text embedding Ψt. Previously the text embedding is nonlinearly transformed to generate conditioning latent variables as the input of the generator. However, latent space for the text embedding is usually high dimensional. With limited amount of data, it usually causes discontinuity in the latent data, which is not desirable.

For learning the generator. To mitigate this problem, we introduce a Conditioning Augmentation technique to produce additional conditioning variables c. In contrast to the fixed conditioning text variable c in, we randomly sample the latent variables c from an independent Gaussian distribution $N(\mu(\Psi t), \Sigma(\Psi t))$, where the mean $\mu(\Psi t)$ and diagonal covariance matrix $\Sigma(\Psi t)$ are functions of the text embedding Ψt. The Conditioning Augmentation yields more training pairs given a small number of image text pairs, and thus encourages robustness to small perturbations along the conditioning manifold. To further enforce the smoothness over the conditioning manifold and avoid overfitting, we add the following regularization term to the objective of the generator during training,

$$D_{KL}\big(N(\mu(\Psi t), \Sigma(\Psi t))||N(0, I)\big) \text{ ----------(2),}$$

Which is the Kullback-Leibler divergence between the standard Gaussian distribution and the conditioning Gaussian distribution. The randomness introduced in the Conditioning Augmentation is beneficial for modeling text to image translation as the same sentence usually corresponds to objects with various poses and appearances.

3.3.2 Stage-I GAN

Instead of directly generating a high-resolution image conditioned on the text description, we simplify the task to first generate a low-resolution image with our Stage-I GAN, which focuses on drawing only rough shape and correct colors for the object. Let Ψt be the text embedding of the given description, which is generated by a pre-

trained encoder. The Gaussian conditioning variables c for text embedding are sampled from $N(\mu(\Psi t), \Sigma(\Psi t))$ to capture the meaning of Ψt with variations. Conditioned on c and random variable z, Stage-I GAN will trains the discriminator D_0 and the generator G_0 by alternatively maximizing L_{D0} in Eq. (3) and minimizing L_{G0} in Eq. (4),

$$L_{D0} = E_{(I_0,t)\sim P_{data}}\left[\log D_0\left(I_0, \Psi t\right)\right] + E_{z\sim P_z, t\sim P_{data}}\left[\log(1 - D_0(G_0(z,c), \Psi t))\right] \text{----------(3)}$$

$$L_{G0} = E_{z\sim P_z, t\sim P_{data}}\left[\log(1 - D_0(G_0(z,c), \Psi t))\right] + \lambda D_{KL}\left(N(\mu(\Psi t), \Sigma(\Psi t)) || N(0, I)\right) \text{----------(4)}$$

Where the real image I_0 and the text description t are from the true data distribution $Pdata$ z is a noise vector randomly sampled from a given distribution P_z (Gaussian distribution in this paper). λ is a regularization parameter that balances the two terms in Eq. (6). We set λ = 1 for all our experiments. Using the reparameterization trick introduced in, both $\mu(\Psi t)$ and $\Sigma(\Psi t)$ will be learned jointly with the rest of the network.

For the generator G_0, to obtain text conditioning variable c, the text embedding Ψt is first fed into a fully connected layer to generate μ and σ (σ are the values in the diagonal of Σ) for the Gaussian distribution $N(\mu(\Psi t), \Sigma(\Psi t))$. c are then sampled from the Gaussian distribution. Our N_g dimensional conditioning vector c is computed by $c = \mu + \sigma \odot \varepsilon$ (where $\odot$ is the element-wise multiplication, $\varepsilon \sim N(0, I)$). Then, c is concatenated with a N_z dimensional noise vector to generate a $W_0 \times H_0$ image by a series of up-sampling blocks. For the discriminator D_0, the text embedding Ψt is first compressed to N_d dimensions using a fully-connected layer and then spatially replicated to form a $M_d \times M_d \times N_d$ tensor. Meanwhile, the image is fed through a series of down-sampling blocks until it has $M_d \times M_d$ spatial dimension. Then, the image filter map is concatenated along the channel dimension with the text tensor. The resulting tensor is further fed to a 1×1 convolutional layer to jointly learn features across the image and the text. Finally, a fully connected layer with one node is used to produce the decision score

3.3.3 Stage-II GAN

When low-resolution images generated by Stage-I GAN usually lack vivid object parts and might contain shape distortions. Some details in the text might also be omitted in the first stage, which is vital for generating photo-realistic images. Our Stage-II GAN is built upon Stage-I GAN results to generate high-resolution images. It is conditioned on low-resolution images and also the text embedding again to correct defects in Stage-

I results. The Stage-II GAN completes previously ignored text information to generate more photo-realistic details. Conditioning on the low-resolution result $S_0 = G_0(z, c_0)$ and Gaussian variables c, the discriminator D and generator G in Stage-II GAN are trained by alternatively maximizing L_D in Eq. (5) and minimizing L_G in Eq. (6),

$$L_D = E_{(I,t)\sim P_{data}}\left[\log D\left(I, \ \Psi t\right)\right] + E_{S_0\sim P_{G_0}, t\sim P_{data}}\left[\log(1 - D(G(S_0, c_0), \Psi t))\right] \text{----------(5)}$$

$$L_G = E_{S_0\sim P_{G_0}, t\sim P_{data}}\left[\log(1 - D(G(S_0, c_0), \Psi t))\right] + \lambda D_{KL}\left(N(\mu\left(\Psi t\right), \Sigma\left(\Psi t\right)\right)||N(0, I)) \text{ ----------(6)}$$

Different from the original GAN formulation, the random noise z is not used in this stage with the assumption that the randomness has already been preserved by S_0. Gaussian conditioning variables c_0 used in this stage and c used in Stage-I GAN share the same pre-trained text encoder, generating the same text embedding Ψt. However, StageI and Stage-II Conditioning Augmentation have different fully connected layers for generating different means and standard deviations. In this way, Stage-II GAN learns to capture useful information in the text embedding that is omitted by Stage-I GAN.

We design Stage-II generator as an encoder-decoder network with residual blocks. Similar to the previous stage, the text embedding Ψt is used to generate the N_g dimensional text conditioning vector c_0, which is spatially replicated to form a $M_g \times M_g \times N_g$ tensor. Meanwhile, the Stage-I result S_0 generated by Stage-IGAN is fed into several down-sampling blocks (i.e., encoder) until it has a spatial size of $M_g \times M_g$. The image features and the text features have concatenated along the channel dimension. The encoded image features coupled with text features is fed into several residual blocks, which are designed to learn multi-modal representations across image and text features. Finally, a series of up-sampling layers (i.e., decoder) are used to generate a W×H high-resolution image. Such a generator is able to help rectify defects in the input image while add more details to generate the realistic high-resolution image. For the discriminator, its structure is similar to that of Stage-I discriminator with only extra down-sampling blocks since the image size is larger in this stage. To explicitly enforce GAN to learn better alignment between the image and the conditioning text, rather than using the vanilla discriminator, we adopt the matching-aware discriminator proposed for both stages. During training, the discriminator takes real images and their corresponding text descriptions as positive sample pairs, whereas negative sample pairs consist of two groups. The first is real images with mismatched text embedding's, while the second is synthetic images with their corresponding text embedding's.

3.4 Algorithm

The stepwise procedure to generate photorealistic image from the given text description is given below:-

Step 1: Take text description of birds as an input.

Step 2: The input is transformed to text embedding Ψt via fully connected layer.

Step 3: The text embedding Ψt then again transformed via fully connected layer to the mean vector $\mu(\Psi t)$ and sigma vector $\Sigma(\Psi t)$.

Step 4: These vectors are interpreted as normal distribution $N(\mu(\Psi t), \Sigma(\Psi t))$, from which output vector c is sampled.

Step 5: KL-divergence term is added to the loss, which prevents each single normal distribution from deviating too far from the unit normal distribution N(0,1).

Step 6: The output vector c from the step 5 is concatenated with the N dimensional noise vector $Z \sim N(0, I)$ to generate 64x64 image through series of upsampling blocks.

Step 7: The text embedding Ψt from step 2 is compressed to N dimension and spatially replicated to form 3D tensor using a fully connected layer.

Step 8: The image generated in step 6 is fed through series of down-sampling blocks until it has 2D spatial dimension.

Step 9: The two tensors from step 7 and step 8 are concatenated and further fed to 1x1 convolution layer to jointly learn features.

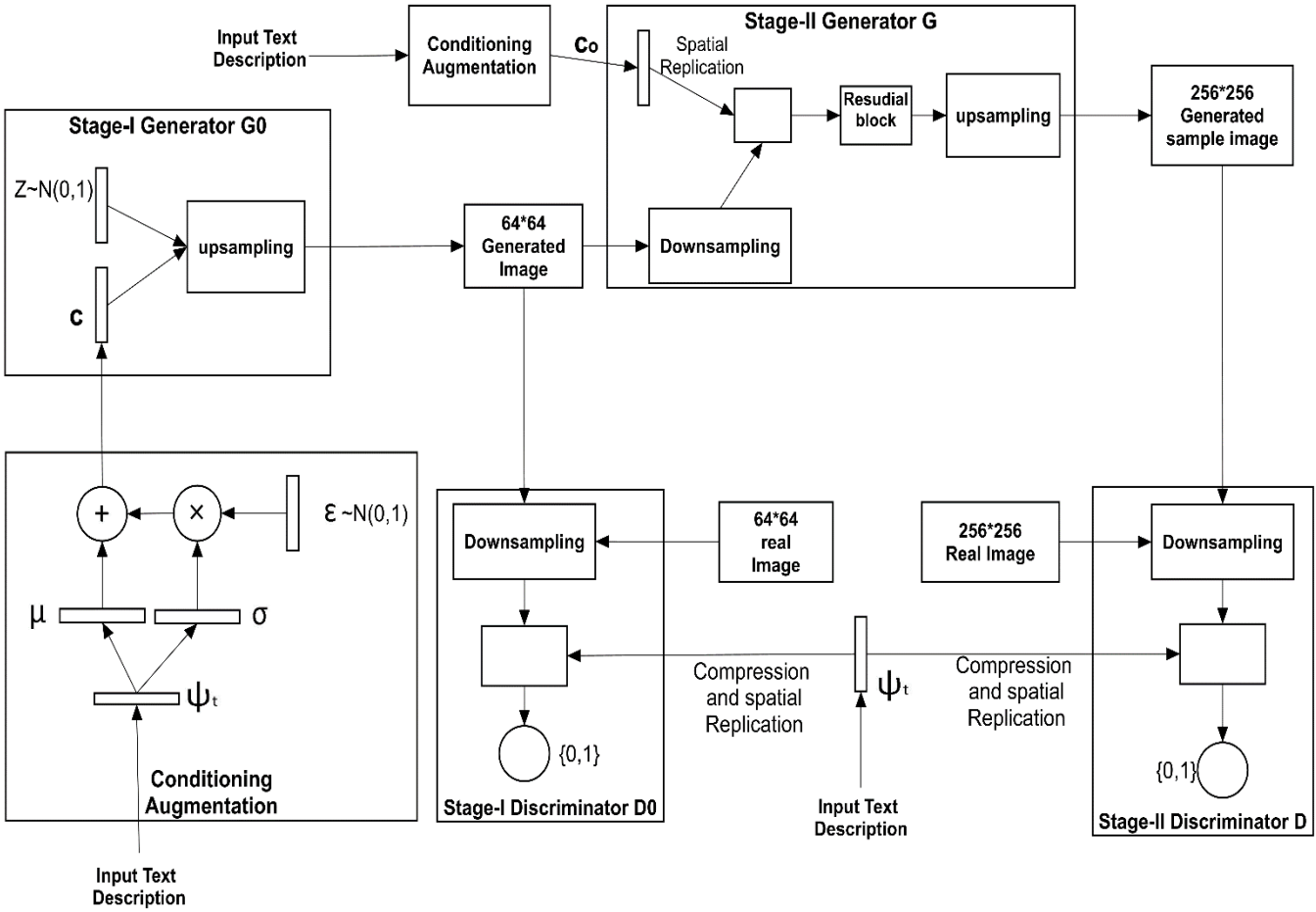
Step 10: Again the text embedding Ψt (generated using different fully connected layer) is used to generate N dimensional vector, which is spatially replicated to 3D tensor.

Step 11: The 64x64 image generated from step 6 is fed into several down-sampling blocks.

Step 12: The image features from step 11 and text features from step 10 are concatenated to generate 256x256 image through series of upsampling blocks.

Step 13: Again step 7, 8 and 9 is repeated with image size of 256x256.

3.5 Architecture



3.6 Implementation and Training

The up-sampling blocks consist of the nearest-neighbor upsampling followed by a 3×3 stride 1 convolution. Batch normalization and ReLU activation are applied after every convolution except the last one. The residual blocks consist of 3×3 stride 1 convolutions, Batch normalization and ReLU. Two residual blocks are used in 128×128 Stack-GAN models while four are used in 256×256 models. The down-sampling blocks consist of 4×4 stride 2 convolutions, Batch normalization and LeakyReLU, except that the first one does not have Batch normalization [10,11]. For training, we first iteratively train D0 and G0 of Stage-I GAN for 600 epochs by fixing Stage-II GAN. Then we iteratively train D and G of Stage-II GAN for another 600 epochs by fixing Stage-I GAN. All networks are trained using ADAM solver with batch size 64 .

3.7 Datasets and Features

We ran the experiment on the Caltech-UCSD Bird (CUB) datasets [12]. CUB contains 200 bird species with 11,788 images. Since 80% of birds in this dataset have object-image size ratios of less than 0.5, as a pre-processing step, we crop all images to ensure that bounding boxes of birds have greater-than-0.75 object-image size ratios. Each image in CUB are provided with 10 descriptions and we split CUB into class-disjoint training and test sets.

CHAPTER 4
RESULT AND DISCUSSION

"Text to Photo Realistic Image using GAN" is a system having an ability to generate photorealistic image from the text description. The result from our model can be seen in the output section below. When the text description is given, stage-I GAN generates 64 × 64 image and further stage-II GAN produce 256×256 image. The output shows that the GAN has generated meaningful shapes, colours and depictions of the objects. However, in some cases, some of the features given in the text is missing in the generated images.In all most all the cases Stage-II generated a highly plausible sample. We also observe that StackGAN has the ability to transfer background from Stage-I images and fine-tune them to be more realistic with higher resolution at Stage-II. Importantly, the StackGAN does not generate good sample only by memorizing the training samples but by capturing the complex underlying language-image relations. We extract visual features from generated images and all training images by stage-II discriminator.The nearest neighbor from the training set can be retrieved for each generated image. By visually inspecting we conclude that generated image have some common features with training sample but are different.

4.1 Testing

4.1.1 Time for execution

Test 1:

We found that the time required for the tested sentence " A black bird with a red tail and yellow beak" to be nearly equal to five seconds.

The output of the test 1 is shown below:

Test 2:

We found that the time required for the tested sentence “this bird has a bright yellow body, with brown on its crown and wings” to be nearly equal to four seconds.

The output of the test 2 is shown below:

Test 3:

We found that the time required for the tested sentence “this bird has a red breast and belly as well as a small bird”to be nearly equal to five seconds.

The output of the test 3 is shown below:

Test 4:

We found that the time required for the tested sentence “this bird has gray crown,belly and white abdomen, with black tarsus and feet”to be nearly equal to four seconds.

The output of the test 4 is shown below:

Test 5:

We found that the time required for the tested sentence “a colorful bird with a yellow body, a black crown and throat, orange bill, and black primries and secondaries” to be nearly equal to five seconds.

The output of the test 5 is shown below:

4.1.2 Average time for training

We have total 600 epoch(loops) in out training phase. We took the first ten epochs and the time taken for each epoch to complete is shown in the below points respectively.

- To execute epoch (loop) zero it took nearly seven and half minutes.
- To execute epoch (loop) one it took six minutes twenty six seconds.
- To execute epoch (loop) two it took six minutes fourty seven seconds.
- To execute epoch (loop) three it took six minutes fifty seconds.
- To execute epoch (loop) four it took six minutes fourty seconds.
- To execute epoch (loop) five it took six minutes fifty eight seconds.
- To execute epoch (loop) six it took six minutes fifty five seconds.
- To execute epoch (loop) seven it took six minutes fourty nine seconds.
- To execute epoch (loop) eight it took six minutes thirty seven seconds.
- To execute epoch (loop) nine it took six minutes fifty five seconds.

$$\text{Avg} = \frac{\sum(i1+i2+i3+\cdots+in)}{n}$$

$$= \frac{\sum(450+380+407+410+400+418+415+409+397+415)}{10}$$

$$= \frac{4101}{10}$$

$$= 410.1\ sec$$

$$= 6\ \text{min}\ 50\ sec$$

We found that the average time for training in each epoch is: six minutes fifty seconds.

4.2 Output

Fig 4.2.1 : Output for the text " a colorful bird with a bright yellow body, a black crown and throat, orange bill, and black primaries and secondaries."

Fig 4.2.2 : Output for the text "a black bird with a bright yellow body, a black crown and throat, orange bill, and black primaries and secondaries."

Fig 4.2.3 : Output for the text "small, roundish bird with off white breast and belly, light brown crown, brown and black colored wings."

Fig 4.2.4 : Output for the text "This small blue bird has a short pointy beak and brown on its wings."

Fig 4.2.5 : Output for the text “This bird is completely red with black wings and pointy beak.”

Fig 4.2.6 : Output for the text “A small sized bird that has a cream belly and short pointed bill.”

Fig 4.2.7 : Output for the text “A small bird with a black head and wings and features grey wings.”

Fig 4.2.8 : Output for the text “This bird is black with green and has a very short beak.”

CHAPTER 5
CONCLUSION AND FUTURE ENHANCEMENT

Stacked Generative Adversarial Networks (StackGAN) with Conditioning Augmentation is good method for generation of plausible samples. It generates the photorealistic image for given text description. This method decomposes the text-to-image synthesis to a novel sketch-refinement process. Stage-I GAN sketches the object following basic color and shape constraints from given text descriptions. Stage-II GAN corrects the defects in Stage-I results and adds more details, yielding higher resolution images with better image quality. From the perspective of the output generated this method generated best result than the others. Compared to existing text-to-image generative models, StackGAN generates higher resolution images with more photo-realistic details and diversity.

In this work, we examined the training and evaluation of a Stack-GAN for highly realistic synthesis of images from text phrases. In future work we'd like to try and scale to larger image-caption datasets like MSCOCO [13]. We'd also like to try a sequential dual-training method, where we train do text-to-image synthesis in tandem with image-to-text synthesis. For multi-category datasets like MSCOCO these might perform better. On further research and acquiring better accuracy, we'd like to enhance this project for the purpose of crime investigation such as face sketching of the criminal from the witness description.

REFERENCES

[1] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in Proc. IEEE Conf. Comput. Vision and Pattern Recognition, 2016, pp. 2414–2423.

[2] Alec Radford , Luke Metz, “Unsupervised representation learning with deep convolutional generative adversarial networks” in Proc. ICLR Conference paper , 2016.

[3] Carl Doersch, “Variational Autoencoders” [online]. Available:https://arxiv.org/abs/1606.05908 [Accessed 06 June 2018]

[4] ZhitingHu, ZichaoYang, XiaodanLiang, RuslanSalakhutdinov, EricP.Xing, “Controllable Generation” [online]. Available: https://arxiv.org/pdf/1703.00955 [Accessed 03 June 2018]

[5] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris Metaxas, “Stacked Generative Adversarial Networks” in Proc. ICCV Oral Presentation Comput. Vision and Pattern Recognition, 2017.

[6] Variational Autoencoders [online] http://kvfrans.com/variational-autoencoders-explained/ [Accessed 06 June 2018]

[7] Behnam Neyshabur, Srinadh Bhojanapalli, Ayan Chakrabarti “Stabilize the Training Process” [online]. Available: https://openreview.net/forum?id=SJahqJZAW [Accessed 09 June 2018]

[8] Junbo Zhao, Michael Mathieu, Yann LeCun “Energy Based GAN”, Cornell University Library [online]. Available: https://arxiv.org/abs/1609.03126?context=cs [Accessed 10 June 2018]

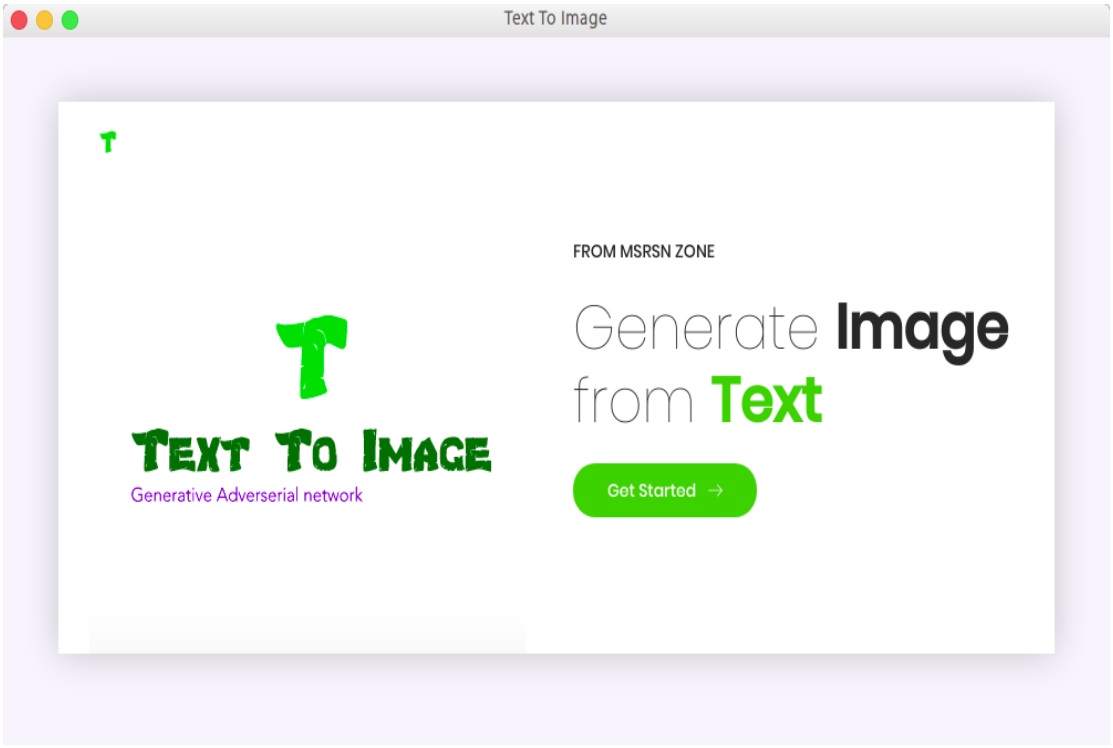
[9] Firdaouss Doukkali “Batch normalization” [online]. Available: https://towardsdatascience.com/batch-normalization-in-neural-networks-1ac91516821c [Accessed 15 novemberv 2018]

[10] S.Sharma “ReLU”. Available: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6 [Accessed 15 novemberv 2018]

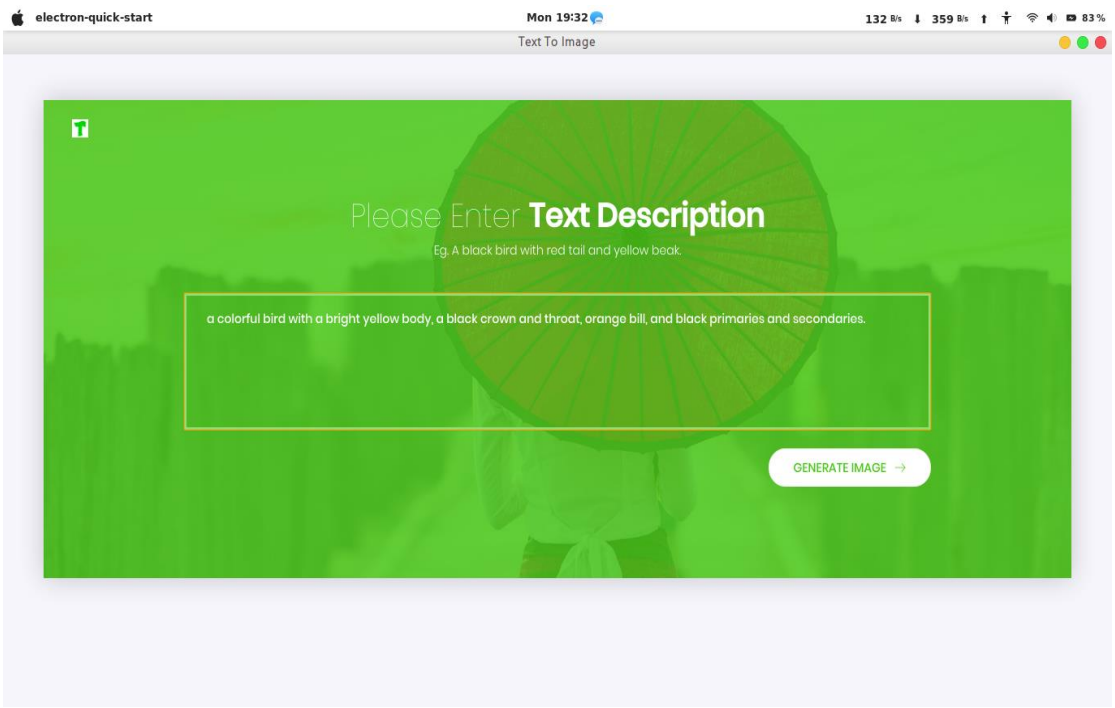
[11] CUB [online] https://github.com/visipedia/tf_classification/wiki/CUB-200-Image-Classification [Accessed 16 novemberv 2018]

[12] MSCOCO [online] https://github.com/cocodataset/cocoapi [Accessed 16 novemberv 2018]

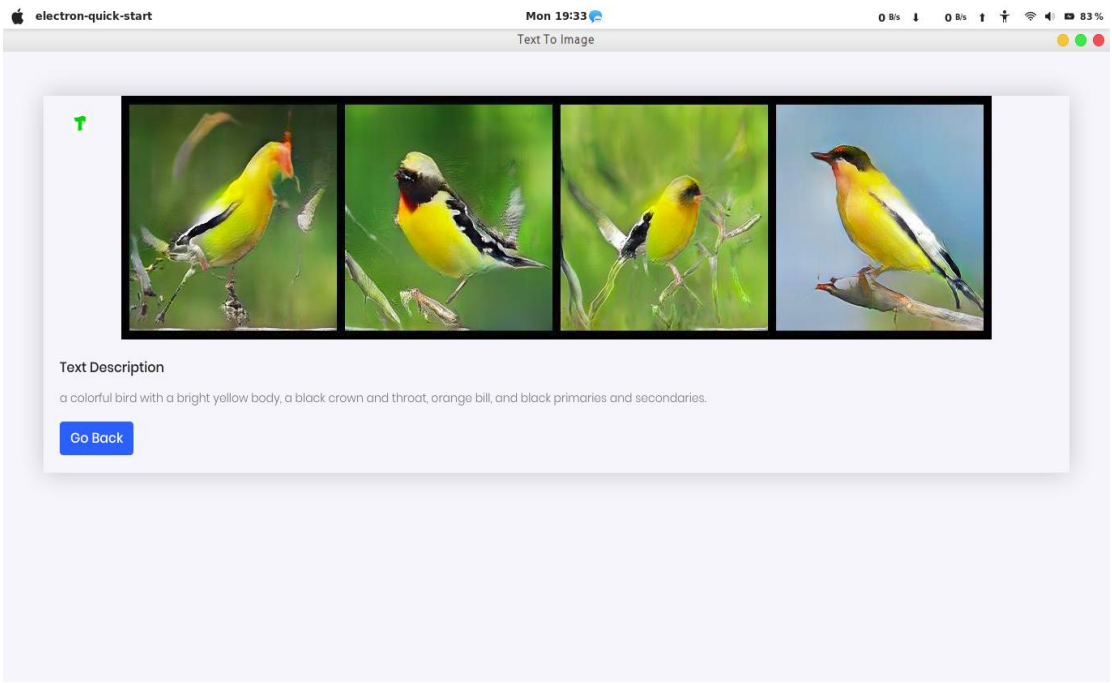
APPENDIX



Initial Graphics User Interface For our Project



Graphics User Interface for taking text from user to generate a photo realistic image in our Project



Graphics User Interface for output according to the text from user to generate a photo realistic image in our Project