Text to Image Generation

Vibhu Dubey, Sohum Sikdar, Abhik S Basu
Indraprastha Institute of Information Technology Delhi
vibhu20150, sohum20339, abhik20165 @iiitd.ac.in

Abstract

Text to Image generation is a deep learning task that involves training a deep neural network on text and images, such that it learns to convert the textual description into visual representations, and finally, it can generate unseen images based on the texts we provide. This report talks about the Datasets used, the commonly used deep learning architectures, and finally, we talk about some baseline models that have performed this task well.

1 Problem Statement

Text to Image generation deals with generating a relevant image or a set of relevant images with a given textual description input describing the scene of the image. Therefore the task is a generation task. This generation task can be limited to a particular or wide-ranging domain of images.

2 Dataset Description

CUB Dataset: There are a total of 11,788 images of 200 different species of birds. The source for these images were various sources from the internet and then they were cropped and annotated to put more focus on the species of the bird. The train set contains 5,994 images of 150 bird species, and the test set contains 5,794 images of the remaining 50 bird species.
Oxford Flowers Dataset: This data consists of a total of 8,189 images of 102 different categories of flowers. Images have been collected from various sources on the internet and then annotated with the correct flower category. There is no predefined train-test split for this dataset. However, each flower has images from 40 to 258.
For the model presented in the final evaluation the dataset being used is the Oxford Flowers Dataset.

3 Related Works

Primarily the two Architectures that are used for Text to Image generation tasks are Generative Adversarial Networks (GANs) and Variational AutoEncoders (VAEs).
GANs use a two-network system, one being the generator network while the other being the discriminator network, the task of the generator is to synthesize images, whereas the discriminator network distinguishes real data from synthesized data, the training continues till the discriminator network is unable to differentiate synthesized data from real-world data. VAEs, on the other hand, use a single network and use a probabilistic approach to generate synthetic data by learning and sampling from the underlying data distribution of the real-world data provided to it. For tasks where more realistic images are to be generated, GANs are the better choice, but for more imaginative and creative images VAEs give better results.
We have considered ControlGAN and RAT-Text2Img as our two baselines, both of which are Generative Adversarial Networks.
As for our model, we have made use of a conditional DCGAN. A DCGAN is a direct extension of the GAN but it explicitly uses convolutional and convolutional-transpose layers in the discriminator and generator, respectively. It was first described by Radford et. al. in the paper Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks. The discriminator is made up of strided convolution layers, batch norm layers, and LeakyReLU activations.
In a conditional DC GAN, additional information (label embedding, in our case) is fed into both the generator and discriminator networks as an extra input, in addition to the random noise that is used as input to the generator network. This additional input enables the generator to produce images that are specific to the given input condition.

4 Methodology

We first give the basic model setup and a short overview of the ControlGAN and the RAT-

Text2Img models. Then in section 3.3, we delve deeper into the architecture presented by our group for the final evaluation.

4.1 ControlGAN

Control Generative Adversarial Networks are fine-tuned GANs trained for controllable text-to-image synthesis. ControlGAN has two parts in its architecture, first, a text encoder and then an image decoder, both of which are trained adversarially. The text encoder works to encode the given text into a fixed dimensional latent code. To this latent code, random noise is concatenated, which is finally fed to the image decoder, which is a Convolutional Neural Network. ControlGAN further has control variables which allow the generated image to be modified on specific attributes. These variables are learnt through a network known as the control predictor, which essentially takes the text as input and predicts the control variables. These control variables are then used to modify the latent code. ControlGAN uses a multi-component loss function that incorporates various objectives to ensure the quality and controllability of the images.

$$L = \lambda_1 L_{GAN} + \lambda_2 L_{CLS} + \lambda_3 L_{rec} + \lambda_4 L_{per}$$

The above equation gives the overall loss, where λ_1, λ_2, λ_3, and λ_4 are hyperparameters.L_{GAN} is the adversarial loss used by GANs, L_{CLS} is a classification loss that makes the generated image match the given set of attributes, L_{rec}, is the reconstruction loss that encourages the generator to produce images which match the given training images and finally L_{per} is a perceptual loss that makes the generated image match the label images with respect to their perceptual features like texture, structure, etc.

4.2 RAT-Text2Img

Recurrent Affine Transformation for Text-to-Image Synthesis is another deep learning architecture used for our task. It mainly consists of three components, a text encoder, a recurrent affine transformer and finally, an image decoder. The text encoder works like the encoder in ControlGAN, i.e. gives the fixed-length vector representation of the text provided. The recurrent affine transformer then takes the encoded text vector as input and generates a series of affine transformation parameters, which are used to transform a set of initial feature maps. Finally, the Image encoder, which is a CNN generates the images. Similar to ControlGAN, the loss here is also a combination of multiple losses, sich as adversarial loss, perception loss and then a L_1 or L_2 loss.

4.3 Conditional DCGAN

Conditional Deep Convolutional Generative Adversarial Networks are a type of GANs that not only generate from a sample of a learnt distribution but can also condition the output to get certain features that are needed. For example, if the GAN is trained to generate images of flowers, the network can be trained to multiple classes/types of flowers, that when, during inference, a certain class is passed to the network, not only does the generator outputs an image of a flower, but also a flower similar to the passed class/type of the flower.
The architecture of the GAN can be broken down into two primary networks, the generator and the discriminator, like traditional GANs, and another support network that is used to generate the embedding of the conditional parameter.
The classes - valued from 1 to 102, are passed to the support network generating the embedding of the class. For the Generator Network, the class embeds are passed through a linear layer which is concatenated with a random noise $\sim N(0, 1)$. This when forwarded through the Generator Network outputs fake images. The Discriminator Network gets the text embeds, which are repeated to match the dimensions of the images, the images and the repeated embeds are then concatenated to be passed through the Discriminator Network which predicts the probability of the image passed to it being a real image or a fake/generated image. The training for all networks are done simultaneously and is explained later. The overall architecture is highlighted in Figure 1.

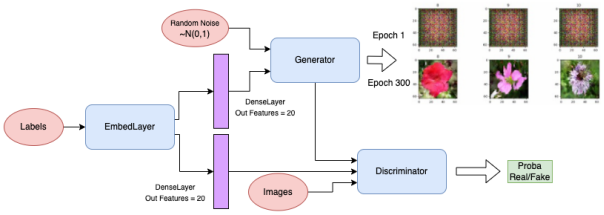

Figure 1: Architecture for the Control DCGAN

5 Experimental Setup

We used ControlGAN and RAT-Text2Img as out baseline models, and their respective codes were taken from their official Github repositories [RAT-

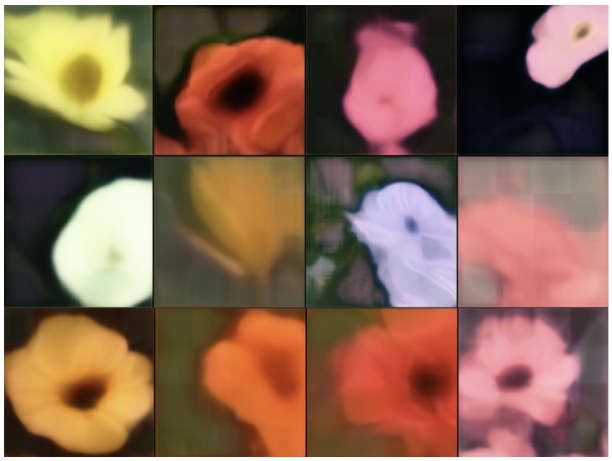

Figure 2: Flowers generated using RAT Text-2-Img

Text2Img], [ControlGAN]. We used the standard Google Colab as our training and testing environment. We used the standard training and testing parameters as described in the respective papers. We had to configure their code to a considerable extent to make it work in our environment and also introduced training checkpointing in RAT-Text2Img.

Due to our limited compute resources, we had to decrease the batch size of RAT-Text2Img from 24 to 12. Since, the standard GANs have a very high training time which might include pretraining also; for eg. RAT-Text2Img was trained for more than 3 days on an RTX3090ti amounting to around 600 epochs, therefore to establish our baseline we decided to finetune the available pretrained models available for 1-2 epochs and then evaluating them.

For Control-GAN we used the pretrained model on the CUB_200_2011 dataset and for the RAT-Text2Img we used the pretrained model on the Oxford Flower Dataset. Both of these models we then finetuned for 1 epoch with a 10^{-6} learning rate for the generator, on their respective datasets. Then we used the test split of the respective datasets to generate the images from the finetuned generators.

The Conditional DCGAN is trained on the Oxford Flower dataset. The architectures used for the underlying Generator and Discriminator Networks are given below in Figure 3, which can be modified to increase the robustness of our model. The loss used for the Discriminator is a standard Binary Cross Entropy loss, which is also derived through the connection graph when calculating the loss for the Generator Network. The model was trained for 300 epochs on a GTX2080. The limitations on the complexity of the networks and the epochs were put in place due to the limitation of compute power available. For the epochs trained, the trained images over time are shown in Figure 4. The saved model can be found at [link]

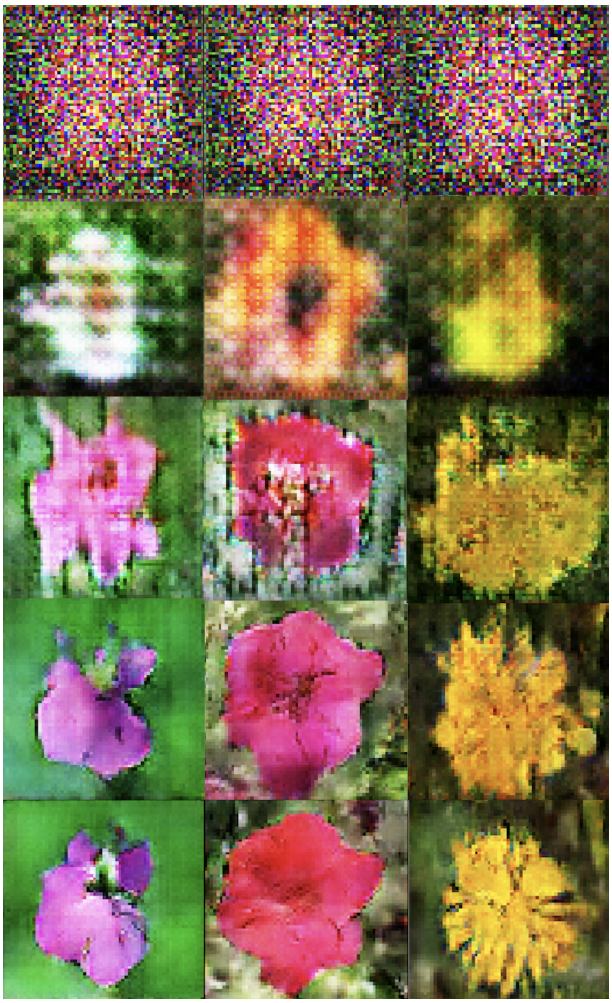

Figure 3: Examples generated over epochs

6 Result

Some observations that could be seen with the RAT-GAN model was that while it did return the required results, the images were blurry and slightly low resolution, further the colors seemed to be diffusing from the flowers. The different components of an image seemed to be combined together.
Whereas if we see the model formed by the Conditional DC GAN, we see that the features that it was forming were much more finer, and it was able to form different types of flower categories after training. The images looked much more like flowers compared to the RATGAN model.
Quantitavely we checked the performance of our model on the basis of 3 different metrics. They were the Frechet Inception Distance, Kernel Inception Distance and the Inception Score. We took

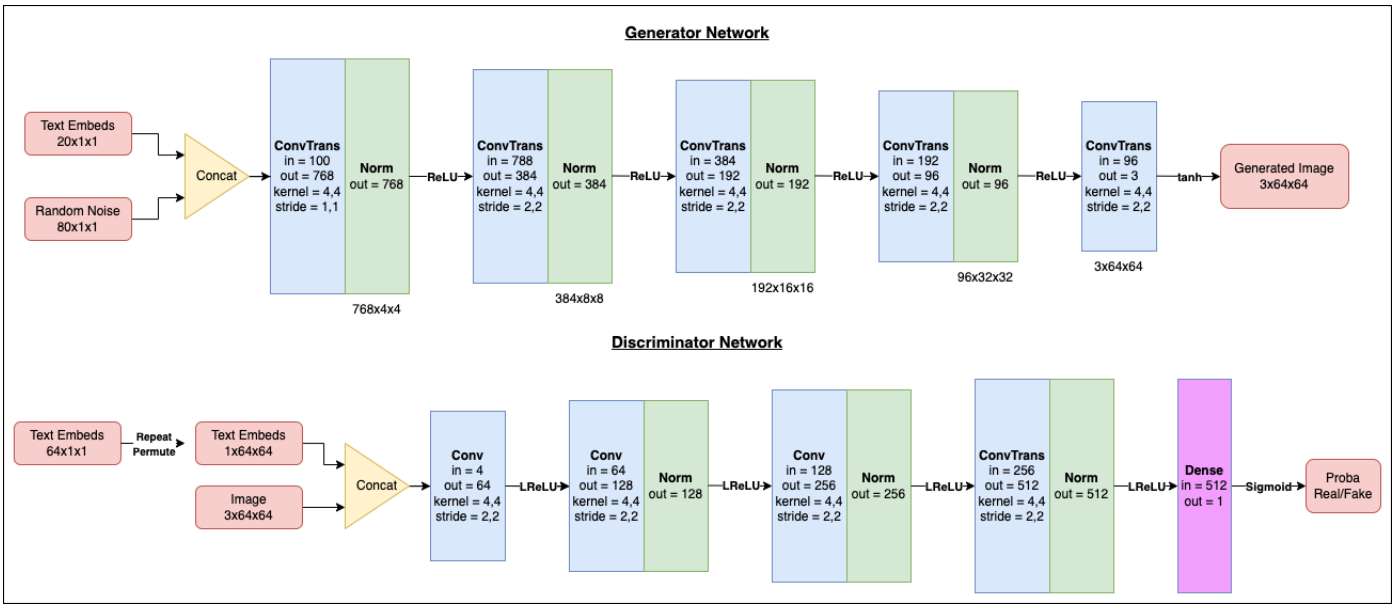

Figure 4: Generator and Discriminator Networks

1000 images from the output of each model/dataset and then carried out our comparisons amongst these in batches of 100 at a time. We see that the inception. While the RATGAN performs marginally better than out model (0.3 more) in terms of inception score.

However, our model has a marginally lesser Frechet Inception Distance and it is much closer to the Dataset Frechet Inception Distance. This is better and shows our model has performed better since the Frechet Inception Distance metric was an improvement over the original Inception Score. This is because Frechet Inception Distance performs well in terms of discriminability, robustness and computational efficiency. Further, it is consistent with human judgments and is more robust to noise than the inception score.

The Kernel Inception Distance difference between both models is very less (0.02) and further the Kernel Inception Distance has its shortcomings as well since it is based off of likelihoods, so it is possible that there is high likelihood and low quality which can perform better, however the difference is marginal and Frechet Distance is considered a better metric over the other two and there we achieve a much better result hence showing that our model works well.

Qualitatively, we released a form with the images for all 3 sources (dataset, RATGAN output and Conditional DCGAN output). The results have been attached. We asked people to evaluate the images (12 for each group) on the basis of their sharpness, smoothness and clarity. Our model performed better across all 3 parameters in comparison to the RATGAN model. The green bar (our model) performs better than the RATGAN (orange bar) in all 3 metrics by a magnitude of almost 1 or more in all cases. The dataset by itself performs better since they are real images and they perform best across all three metrics.

Figure 5: Flowers generated using conditional DCGAN

7 Error Analysis

$$l(x, y) = L = \{l_1, ..., l_N\},$$

$$l_n = -[y_n \cdot \log x_n + (1 - y_n) \cdot \log(1 - x_n)] \quad (1)$$

Performance across Metrics				
S. No.	**Output Source**	**Frechet Inception Distance**	**Kernel Inception Distance (Mean, Std. Dev.)**	**Inception Score (Mean)**
1	Dataset	1.1471	0.0212, 0.0085	2.8224
2	RAT Text2Img	30.2377	0.2753, 0.0148	2.8529
3	Conditional DCGAN	5.1238	0.2932, 0.0154	2.5468

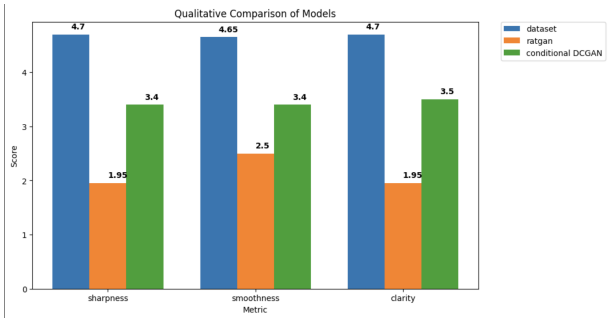

Figure 6: Qualitative Metrics Histogram

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)]$$

$$+\mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \quad (2)$$

We use the standard loss and updation rule as introduced in the GAN paper by Goodfellow. In order to avoid the early vanishing gradients while training the generator(G) we $\max_G \log(D(G(z)))$ where $z \sim p(z)$ instead of $\min_G \log(1 - D(G(z)))$.

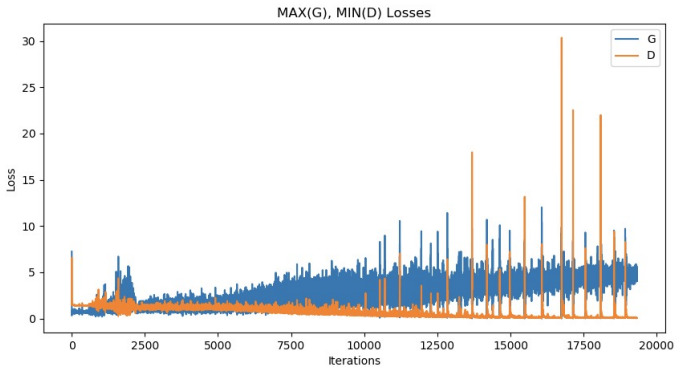

Figure 7: Loss Across Iterations

From Figure 7 of the loss we can observe that as the the model is trained the loss of the generator is maximised. Also, we can observe that as the generator learns the underlying data distribution and therefore starts generating better images, the job of discriminator becomes increasingly tough. Therefore, we can see the spikes in the loss of the discriminator towards the end since it is difficult for the discriminator to distinguish between the generated and actual images. Ideally at the end the discriminator can only do random guesses.

8 Team Contribution

Abhik: Ideation, Model, Testing, PPT
Sohum: Ideation, Model, Report, PPT
Vibhu: Ideation, Model, Testing, Report

9 Future Work

Possible avenues for working in the future could focus towards more robust text to image generation models. This means the model could be made such that the images are of higher quality. Currently the generated images suffer from low resolution, blurriness and lack of details. Furthermore, contextual information should be more heavily incorporated into the training of the models. This would help in making the images much more detailed and finer in its description.
Additionally, mulitmodal learning can also be incorporated wherein audio and video signals help in the generation of much more diverse and informative images. It will also add more coherence between the generated images and the input.
Lastly, the models should be such that they can be trained on a large scale dataset, it would allow the models to get a much better description of the real world. Future work should be focussed on training on large scale datasets for better and more diverse image generation.

10 References

[1] Li, B., Qi, X., Lukasiewicz, T. and Torr, P., 2019. Controllable text-to-image generation. Advances in Neural Information Processing Systems, 32.

[2] Ye, S., Liu, F. and Tan, M., 2022. Recurrent Affine Transformation for Text-to-image Synthesis. arXiv preprint arXiv:2204.10482.

[3] https://github.com/senmaoy/RAT-GAN

[4] https://github.com/mrlibw/ControlGAN

[5] https://github.com/hanzhanggit/StackGAN-inception-model

[6] https://github.com/Yutong-Zhou-cv/Awesome-Text-to-Image

[7] https://pytorch.org/tutorials/beginner/dcgan$_{f}aces_{t}utorial.html$

[8] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2020. Generative adversarial networks. Communications of the ACM, 63(11), pp.139-144.

[9] https://github.com/soumith/ganhacks

[10] Radford, A., Metz, L. and Chintala, S., 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

[11] Borji, A., 2022. Pros and cons of GAN evaluation measures: New developments. Computer Vision and Image Understanding, 215, p.103329.