SATGAN: Image Generation from Text Descriptions using Self Attention Generative Adversarial Network

Imranul Ashrafi, Arani Shawkat, Muntasir Mohammad
Supervisor: Dr. Nabeel Mohammed
North South University

21/05/2019

1 Abstract

Generation of fine grained image with details from text descriptions is a highly challenging task in Computer Vision. Various successful attempts have been taken so far but most of the tasks lack details and do not match the text descriptions properly. In this paper, we propose SATGAN, a two stage Generative Adversarial Network with Self Attention performed on text descriptions. The first stage draws the primary 64x64 image and the second stage applies the attention (on text) then generates high resolution 256x256 image. We experimented our model on CUB-200 (2011) Birds dataset. Our extensive experiment and comparison with the state-of-the-art models shows that using Self Attention on text descriptions and Spectral normalization improves the quality of the generated image while reducing the computational cost. The Inception score was found to be 5.04 ± 0.37, which is a boost of 15.6% on the CUB dataset. Also the model scored an FID score of 42.87.

2 Introduction

Human beings have been given the power of imagination. They can quickly imagine a scenario based on text descriptions. For example, if a person is to read a novel, they can actually imagine the meadows, the plot unraveling. The recent hype of the world of Computer Vision is to give that same power or something close, to a machine. This is where "Text to Image Generation" comes in. This is very much needed in various applications like, photo-editing, computer-aided design etc. Text to Image Generation using Generative Adversarial Network (GAN)[1] have shown the most promise. GANs' using deep convolutional networks have been especially successful[2][3][4]. Using this, various state of the art architectures have been made which are discussed in the Related Work section. But the main problem of those architectures is that, they are very much computationally costly. In this paper, we tried to solve 2 problems:

- Reducing the computational cost for training GAN architecture
- While reducing computational cost, still being able to produce high quality image

At present the state of the art architecture Attention Generative Adversarial Network (AttnGAN)[5] shows the most accuracy in terms of producing close to real images. But the problem here remains that of computational cost. Attention is given to not only images but also to word contexts which increases

computational cost. Furthermore, 2 Generators out of 3, using attention architecture adds up even more to the computational cost. In contrast to that, in our paper, we experimented with Vanilla GAN to produce images, GAN with Self Attention [6] on images, and GAN with Self Attention on Text Descriptions. We also used Spectral normalization [7] for normalizing image. At the same time, we decreased the number of Generators to 2, 1^{st} Generator produces 64x64 images and the 2^{nd} upsamples the image to 256x256 based on the attention maps from Self Attention architecture. Detailed explanation on results are described in section 5.

3 Related Work

Mansimov et. al. used bidirectional RNN attention model along with the conditional DRAW network in order to generate images from text descriptions[8]. AttnGAN[5] uses attention driven multi stage refinement for text in order to generate images. At first a low resolution image is generated using the sentence vector(vector form of text descriptions). After that, each sub-region of image is refined using word vector of the sentence based on context. In addition, a deep multi-modal similarity model is introduced for calculating GAN loss. Image Generation using PixelCNN[9] used the conditioning GAN based on modified PixelCNN decoders. These conditions can be vectors, labels, tags and latent embedding. The difference between PixelCNN and any other architecture is that, along with generating excellent samples, it returns explicitly probability densities. These densities help to generate excellent samples and use them as transfer learning in other categories. Based on the condition, the model can generate various outputs. Scott Reed used Deep Conditional GAN (DC-GAN)[10] to produce finer images. They used deep convolutional and recurrent text encoders for obtaining vector representations of text descriptions. Matching-aware discriminator (GAN-CLS) was also used in order to discriminate between real and fake images as well as real image and mismatch text. They were also the first to use Inception score as a metric for determining accuracy of GAN. Additional condition of real image and mismatched text are added to the GAN. StackGAN-v2[11] uses the architecture from StackGAN-v1 which uses two stages of GAN, one for generating low resolution primary image from text and other one for generating high resolution image using the low resolution primary image and the text description as inputs. Furthermore, StackGAN-v2 also uses multiple generators and discriminators in order to generate images in multiple scales. Zizhao Zhang introduced hierarchical-nested adversarial objectives inside networks to produce high resolution images[12]. They also introduced a new

visual semantic similarity measure. LSTM (Long Short Term Memory) is basically a Recurrent Neural Network (RNN). The basic difference between LSTM and any other RNNs' is that it is able to remember the relationships among vectors over a long distance which other RNNs' find very hard to do. It is practically a nature of LSTM. Xu Ouyang represented an architecture that uses LSTM network to find semantic meaning from the input text. They used the real image as the goal for multiple similar sentences and showed that it produced better results. These all were done based on generating image on a single category. Multi-Instance StackGAN[13] on the other hand produced multiple instances from a broader variety of categories. The model showed promise in generating complex scene composition consisting of multiple objects based on input text description. Vashisht Madhavan et. al.[14] came up with a Dual loss DCCGAN. They used encoded captions and used them for generating images in DCCGAN (Deep Convolutional Conditional GAN).

4 Methodology

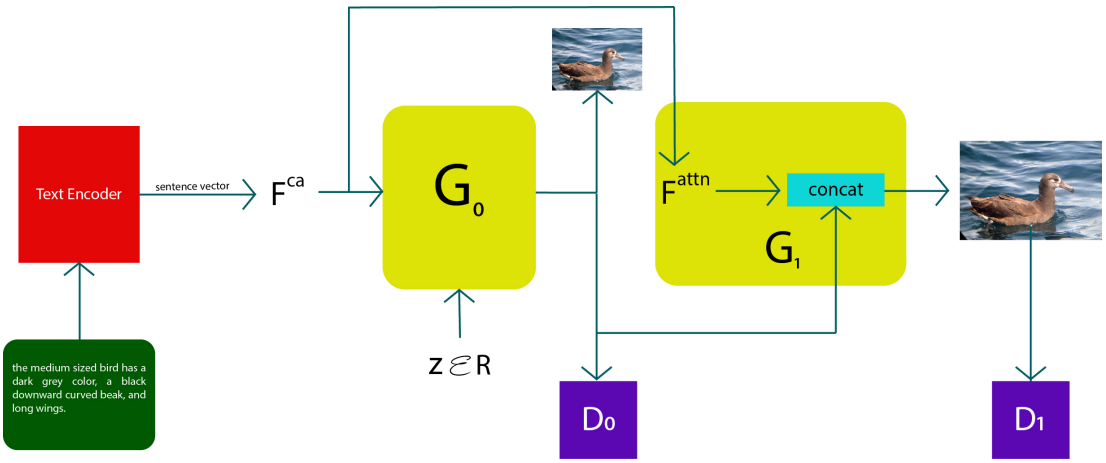


Figure 1: GAN architecture with Self Attention on Text

4.1 Conditioning Augmentation

The text embedding is generally non-linearly transformed into conditioning latent variables which are usually of high dimension (>100 dimensions). With limited amount of data, a type of discontinuity is created in the latent data matrix, which is not desirable for training purposes. This is where Conditioning Augmentation comes in. The trick here is that, latent random variables $\hat{c}$ are sampled from a Gaussian distribution $N(\mu(\varphi_t), \sum(\varphi_t))$ where

the mean $\mu(\varphi_t)$ and diagonal covariance matrix $\sum(\varphi_t)$ are the functions of the text embedding φ_t. The Kullback-Leibler divergence (KL divergence) is also used as additional regularization term for smoothing the learning curve for generator :-

$$D_{KL}(N(\mu(\varphi_t), \sum(\varphi_t))||N(0,1))$$

4.2 Generative Adversarial Network

GAN or Generative Adversarial Networks' have generally 2 networks - a Generator Network and a Discriminator Network. The Generator Network generates images and Discriminator Network tries to deduce whether the image is a generated one or an actual one. On the basis of feedback from Discriminator, Generator again generates images. A sort of min-max game takes place between the two.

In our experimental architecture, there are 2 Generators (G_0, G_1) which have hidden states (h_0, h_1). The output images are (x_0, x_1). z is our noise vector and e is our sentence embedding vector. Our architecture becomes thus -

$$x_0 = G_0(z, F^{ca}(e))$$
$$C_1 = F^{attn}(F^{ca}(e))$$
$$C_1 = concat(C_1, x_0)$$
$$x_1 = G_1(C_1)$$

We also experimented with applying Self Attention on image. Then the equation changes to -

$$C_1 = F^{attn}(x_0))$$
$$C_1 = concat(C_1, F^{ca}(e))$$

Here, x_1 and x_2 are of shape (batch_size, 3, 64, 64) and (batch_size, 3, 256, 256) respectively. Here F^{ca} represents the Conditional Augmentation[11] and F^{attn} represents the Self Attention module.

4.2.1 Stage-I Generator

At first, a noise vector and sentence embedding vector is sent to our Generator G_0. The sentence embedding vector is passed through the Conditional Augmentation[11] and produces a sentence conditional vector. The sentence conditional vector and the noise vector are then concatenated to produce conditioning vector. The conditioning vector is then passed through the

linearity layer, normalization layer, non-linearity (ReLU) layer. The vector is then upsampled 4 times to produce a (batch_size, 32, 64, 64) vector. This vector is then convoluted over to produce a (batch_size, 3, 64, 64) size images. This is the 64x64 image produced by the first Generator G_0.

4.2.2 Stage-II Generator

In the 2^{nd} Generator G_1, no noise is applied. Instead, the image from the G_0 is taken and reshaped into (batch_size, 8, 64, 64) in order to apply Self Attention[6] mechanism on the image. The image is again reshaped back to (batch_size, 3, 64, 64). For the experiment of applying attention on text only, the attention is applied on the sentence conditional vector. Then the vector and the image from G_0 is concatenated. In both cases, the image is passed through a bunch of hidden layers, batch normalization layer and non-linearity (ReLU) layer in order to finally produce an encoded image. In case of the experiment on applying attention on image only, the image is concatenated with the sentence conditional vector and a 2^{nd} conditioning vector is produced. Again, in both cases, this conditioning vector is then again passed through hidden layers, batch normalization layer and a non-linearity (ReLU) layer. The vector is then sent to a Residual Block[15].

Before going any further, a little discussion on Residual Blocks is necessary. Neural Networks are universal function approximators which theoretically can increase their accuracy based on the number of layers used. But practically speaking, there are the problems of vanishing gradients and curse of dimensionality, which makes learning both simple and complex functions for the network becomes quite difficult. At one stage, the accuracy starts to saturate and eventually degrade. This is where residual blocks[15] come in. Residual blocks keep the residue gradients and use that to learn for the present layer and the layer about 2-3 hops away. In this way, simple functions like identity functions can also be trained inside the network. If $\mathbf{x}$ is the image, W_i is the weight applied on the image in the present layer and W_s is the weight applied on the image 2-3 hops back, then the equation for residual blocks[15] become

$$y = F(x, W_i) + W_s x$$

The above equation is the case when the input and output dimensions are not of the same dimension. In our case, the weights are generated from applying Spectral Normalization[7] on the layers.

After going through Residual Block, 4 times upsampling and a final Spectral Normalization[7] and non-linearity (Tanh) gives the final (batch_size, 3, 256, 256) image.

4.3 Self Attention

Self Attention solves the problems that most GAN models[16][17][18] have that are built using convolutional layers. In case of convolution that processes information from local neighbourhood, it becomes computationally inefficient for modelling long-range dependencies in images. In our case, Self Attention mechanism is applied because of the fact that, it produces attention maps based on images and also in texts only which we believe that improves the overall rendering of image and shapes.

Self Attention adapts the model of [19] which enables both the generator and the discriminator to model relationships between widely separated spatial regions. The image features of $x \epsilon \mathbb{R}^{C*N}$ are first transformed into 2 feature spaces, $f(x) = W_f x, g(x) = W_g x$

$$\beta_{j,i} = \frac{exp(s_{ij})}{\sum_{i=1}^{N} exp(s_{ij})}, \text{ where } s_{ij} = f(x_i)^T g(x_j)$$

The output of the attention model is -

$$o_j = \sum_{i=1}^{N} \beta_{j,i} h(x_i), \text{ where } (x_i) = W_h(x_i)$$

Here, $W_g, W_f \epsilon \mathbb{R}^{\bar{C}*C}$, $W_h \epsilon \mathbb{R}^{C*C}$ and $\bar{C} = \text{C/s}$. The final output then becomes -

$$y_i = \gamma o_i + x_i, \text{ where } \gamma \text{ is initialized as } 0$$

4.4 Loss Function

In order to generate close to real images, the loss functions are pivotal. Here, Binary Cross Entropy has been used as the loss function for both generator and discriminator.

$$L_{G_i} = -1/2[log(D_i(x_i))] - 1/2[log(D_i(x_i, e))]$$

$$L_{D_i} = \{-1/2[log(D_i(x_i))] - 1/2[log(D_i(x_i, e))]\}^{real\_img} + \{-1/2[log(D_i(x_i))] - 1/2[log(D_i(x_i, e))]\}^{fake\_img} + \{-1/2[log(D_i(x_i))] - 1/2[log(D_i(x_i, e))]\}^{wrong\_img}$$

KL Divergence Loss further made the training process easier for the generators and discriminators.

$$D_{KL}(P||Q) = -\sum_{x \epsilon X} P(x) \frac{Q(x)}{P(x)}$$

Hence the final loss becomes -

$$L_{G_i} = L_{G_i} + D_{KL}$$

$$L = L_{G_i} + L_{D_i}$$

Here, the discriminator is trained for the real image, i.e. the image from the dataset; the fake image, i.e. the image generated from the generator and the wrong image, i.e. changing labels for the images in the dataset in order to train the discriminator to consider these as false images as well.

5 Experiment

For our implementation we used various steps and explored each criteria differently. We used cropped 64x64 images for our training for Stage 1 and 256x256 for Stage 2, with batch size 32. The training Dataset is Birds Dataset of CUB-200 (2011) which contains 200 bird species. The Generator and Discriminator networks are trained using the ADAM optimizer. The learning rate was set to 0.0002 initially and decayed by half after every 100 epochs. Several attempts for training were taken and modifications were made based on demand.

5.1 Evaluation Metric

It is very difficult to evaluate generative models result because the generated images has to be scored. Previous papers have applied human ranking and Inception Score. Human ranking is difficult to do and it is really not possible for every single image. So, we chose Inception Score and FID Score as our main evaluating method. Although Inception Score is not really a good metric and has some issues [20], the metric is still used because most of the Generative Models use this one for evaluating.

On the other hand, Frchet Inception Distance (FID score) is a relatively new metric in this area. It is actually a better measure because it calculates the distance between the generated image and the real image. Thus, combin-

ing both scores we evaluate our model.

5.2 Analysis on Image

The training is done separately on two stages. The first stage is configured to accept embedded text descriptions and noise. This stage produces 64x64 low resolution image. The purpose of this generator is to generate the correct color and shape representation with respect to the original image. This is a tough task since the GAN model is hard to converge. Our results on the first generator draws the images quite good, though there are images where the shapes are not correctly drawn due to the high variance in real image colors and shapes. The second generator is responsible for drawing high quality 256x256 images from the generator 1 image.

5.2.1 One Stage (3x3 Convolution)

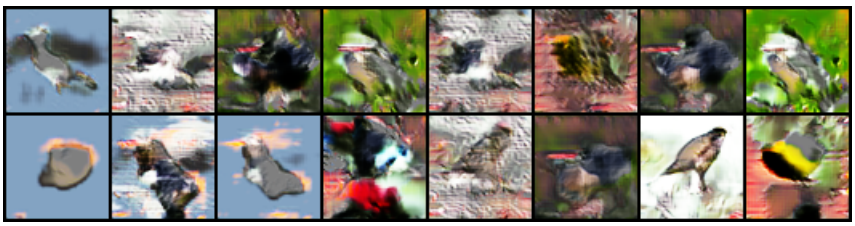


Figure 2: Sample Test Image (3x3 Convolution - 1 Stage Generator)

First, we train our one stage GAN with 3x3 convolution for the layers in generator along with the default loss of generator. It was evident that the discriminator was reaching 0 loss within few epochs and the generator was not learning anything. This is a problem of GAN which was mentioned in the original paper[1]. There is no defined way to solve this problem. To solve the problem, the discriminator was frozen for initial epochs and then unfrozen so that the generator can have enough time to learn the contexts. We used this method throughout our experiments for the later experiments. However, after unfreezing the discriminator still went back to zero loss state, which means the discriminator became strong too quickly. The image generated from the generator was not too good. We found Inception Score 4.01 ± 0.26 and FID Score 145.98 for this experiment.

5.2.2 One Stage (5x5 Convolution, KL Loss)

Figure 3: Sample Test Image (5x5 Convolution, KL Loss - 1 Stage Generator)

To address this problem we figured out that the discriminator is acting stronger than the generator. So, instead of doing 3x3 convolution, we increased the filter size to 5 to do 5x5 convolution on each upsampling of the features. Upsampling was done by the nearest neighbour interpolation. This change made the model stable a bit till 150 epochs. However, the results were improving. To further address the issue, we added the KL Divergence loss to the generator. This change made the GAN training to converge better. We further noticed that reducing the learning rate after some defined epochs is a good idea and thus we reduced learning rate by 1/2 after every 100 epochs. These changes were consistent for the later experiments of our model. We found Inception Score 4.05 ± 0.20 and FID Score 74.44 for this experiment.

5.2.3 Two Stage (No Attention)

Figure 4: Sample Test Image (No Attention - 2 Stage Generator)

For generating 256x256 images we added a second stage generator. First we trained for sufficient epochs only with residual blocks and convolution layers. The result was satisfactory as it matched the earlier paper results of this particular area of text to image generating. The second stage itself does a pretty good job on rendering high quality images. But there are images also

which are not in the correct shape and color. This is because the generator 1 fails to generate the correct shape and color and thus the generator 2 cannot produce the correct image as it is conditioned on the first generator. We found Inception Score 4.96 ± 0.24 and FID Score 49.33 for this experiment.

5.2.4 Two Stage(Attention On Image)

Figure 5: Sample Test Image (Attention on Image - 2 Stage Generator)

After training the vanilla two stage GAN, we applied Self Attention on the conditioned vector which is produced from the addition of the generator 1 image and the text embedding. After sufficient epochs of training, we noticed that the results images were good but not as good as the vanilla GAN. This is because when applying attention, the whole image is considered. Thus, the operation considers the background also. The attention does good when producing good shape and colors but it fails to distinguish between the bird and the background. In the original implementation of Self Attention in GAN[6] was actually image to image synthesis. Since our model in conditioning text to generate image, the attention mechanism fails to generate correct results. We found Inception Score 3.22 ± 0.10 and FID Score 145.75 for this experiment.

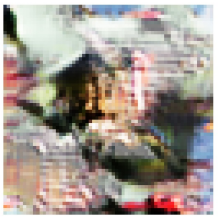
5.2.5 Two Stage (Attention On Text)

Figure 6: Sample Test Image (Attention on Text - 2 Stage Generator)

Since we did not get satisfactory results on applying attention on generated image, we also experimented applying attention on the sentence embedding vector after the conditioning augmentation. After sufficient epochs, we observed that this method produces really good images. We compared these images from the same epoch with the no attention model, we observed that the results improves. The main difference between these two is that, having attention on text separates the backgrounds well while generating quality images with detail. We found Inception Score 5.04 ± 0.37 and FID Score 42.87 for this experiment.

5.3 Quantitative and qualitative results

We tested our model after each experiment. The measured Inception Score and the FID scores are done on the official CUB Dataset test set which generates 2928 images. The FID score is based on Inception v3 model. We also compared our results with the state-of-the-art models.

CAPTIONS	a bird with a short, rounded beak which ends in a point, stark white eyes, and white throat.	this bird has jet-black, slightly lustrous feathers, yellow eyes, and a beak that tapers to a sharp point.	there is an all black bird with dark brown eyes.
STAGE 1 (3x3 Convolution)			
STAGE 1 (5x5 Convolution, KL Loss)			
STAGE 2 (No Attn)			
STAGE 2 (Self Attn + Image)			
STAGE 2 (Self Attn + Text)			

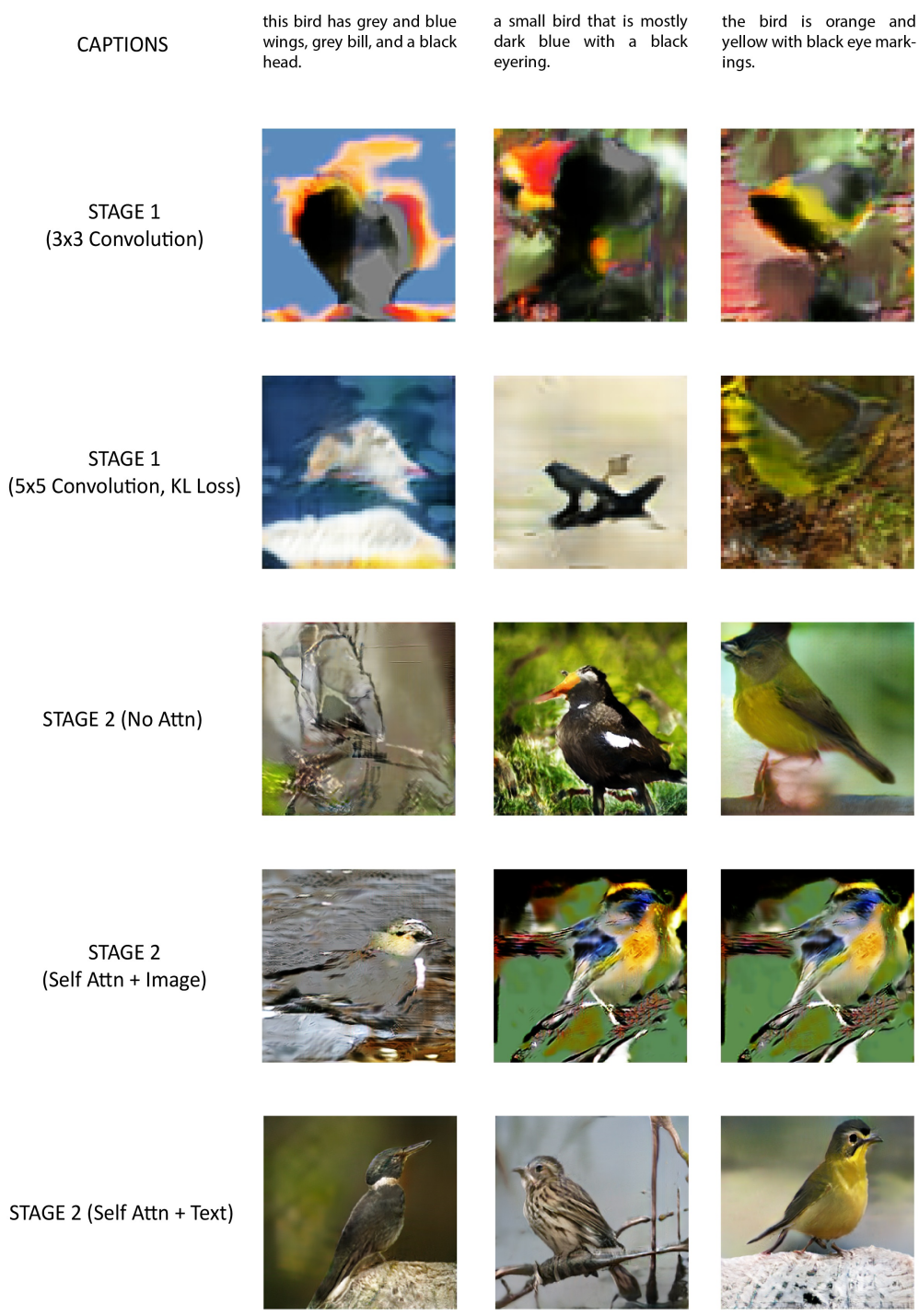


Figure 7: Comparison on models based on visual aspect

Method	Inception Score	FID Score
SATGAN(1 Stage, 3x3 Conv)	4.01 ± 0.26	145.98
SATGAN(1 Stage, 5x5 Conv, KL)	4.05 ± 0.20	74.44
SATGAN(2 Stage, No Attn)	4.96 ± 0.24	49.33
SATGAN(2 Stage, Image Attn)	3.22 ± 0.10	145.75
SATGAN(2 Stage, Text Attn)	$\mathbf{5.04 \pm 0.37}$	**42.87**
StackGAN++	4.04 ± 0.05	15.30
AttnGAN	4.36 ± 0.03	–

SATGAN outperforms the best reported inception score by 0.68 ± 0.34, i.e. 15.6% on CUB dataset. Though Inception score is not a good metric for evaluating generative models as discussed in [20], but it is still used for comparing with the state-of-the-art generative models. On the other hand, the state-of-the art FID score could not be achieved. We believe the reason to be: since our Inception Score is higher, hence the images produced contain much more diversity than the state-of-the-art models. On the other hand, the FID Score is a metric used for evaluating the distance between original images from dataset and the generated images. Hence the reason for the FID Score to come up short is pretty obvious. It might also be the case of lesser training time than the state-of-the-art architectures themselves. Overall, in order to evaluate a generative model, we believe it is necessary to take both Inception Score and FID Score into account.

6 Conclusion

The contribution of our work is a newly proposed architecture SATGAN which reduces the number of Generators and is still being able to generate high quality images from text descriptions. From our experiment we found that attention applied to only text descriptions is good enough and computationally cost effective for generating high quality images. Our SATGAN outperforms the best reported state-of-the-art architectures in terms of generating diversified yet contextually close enough, high quality images. We believe this experiment will usher in a new side of analysis on GAN architectures and also be used as an example for understanding which metrics are better suited for generative model evaluation.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

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

[3] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690, 2017.

[4] E. L. Denton, S. Chintala, R. Fergus, *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, pp. 1486–1494, 2015.

[5] 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.

[6] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," *arXiv preprint arXiv:1805.08318*, 2018.

[7] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference of Legal Regulators*, 2018.

[8] E. Mansimov, E. Parisotto, J. L. Ba, and R. Salakhutdinov, "Generating images from captions with attention," in *International Conference of Legal Regulators*, 2016.

[9] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves, *et al.*, "Conditional image generation with pixelcnn decoders," in *Advances in Neural Information Processing Systems*, pp. 4790–4798, 2016.

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

[11] 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 *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[12] Z. Zhang, Y. Xie, and L. Yang, "Photographic text-to-image synthesis with a hierarchically-nested adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6199–6208, 2018.

[13] A. Fu and Y. Hou, "Text-to-image generation using multi-instance stackgan," *Department of Computer Science Stanford University Stanford, CA*, vol. 94305, p. 26, 2016.

[14] V. Madhavan, P. Cerles, and N. Desai, "Image generation from captions using dual-loss generative adversarial networks,"

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[16] N. Parmar, A. Vaswani, J. Uszkoreit, Ł. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," *arXiv preprint arXiv:1802.05751*, 2018.

[17] 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.

[18] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *International Conference of Legal Regulators*, 2018.

[19] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7794–7803, 2018.

[20] S. Barratt and R. Sharma, "A note on the inception score," in *ICML 2018 Workshop on Theoretical Foundations and Applications of Deep Generative Models*, 2018.