Text-to-Image generation in Generative Adversarial Networks : A Review

Yash Mathur
Computer Science Dept.
CMR Institute of Technology
Bengaluru, Karnataka
Email: y.mathur1498@gmail.com

Dipnarayan Roy
Computer Science Dept.
CMR Institute of Technology
Bengaluru, Karnataka
Email: dipnarayan.roy@gmail.com

Vishal Kundar
Information Science Dept.
CMR Institute of Technology
Bengaluru, Karnataka
Email: vkundar@gmail.com

Abstract—**The advancement of deep learning in recent years has spread out through various domains. From automation to forecasting, it is now found everywhere. One of the recent developments in the field has been Generative Adversarial Networks (GAN). The model was proposed based on zero-sum game theory, where two neural nets compete against each other and has become a new research hotspot. GANs have been widely studied due to the enormous application prospects, including image and vision computing and video and language processing. One of these applications is text-to-image generation, which is being reviewed in this paper. A simple caption can render a high quality image that can deceive the human eye. Although GANs have shown remarkable success in various tasks, they still face challenges in generating high quality images. In this paper we will review text-to-image generation using StackGAN or Stacked Generative Adversarial networks.**

I. Introduction To Text-to-Image Generation

In this paper work, we are interested in translating text in the form of single-sentence human-written descriptions directly into photo-realistic image pixels. For example, “this particular bird has blue wings and a red belly”. The problem of generating images from visual descriptions gained interest in the research community, but it is far from being solved.

Most recently proposed text-to-image synthesis are based on generative adversarial networks (GANs)- an emerging generative model proposed by Ian Goodfellow of Google Brain scientists in 2014. As a new method of learning and generative model, GAN can avoid some deficiency in the practical application of some traditional generation models, and can subtly optimize some loss functions that are hardly to deal with through adversarial learning, and realize the semi supervised and unsupervised learning technology by implicitly modeling the high dimensional distribution of data. GANs excel in various challenging tasks, such as realistic image generation, video frame generation, artistic style migration, etc.

However, it is very difficult to train GAN to generate high-resolution photo-realistic images from text descriptions. Simply adding more upsampling layers in state-of-the-art GAN models for generating high-resolution (e.g.256×256) images generally 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 which may result in the model collapse. Here, to annihilate the difficulties, we are going to review the StackGAN [1] method which decomposes the problem into two stages- one that deals with the low resolution images and the other that deals with the higher resolution images, therefore reducing the chances of model collapse and efficient realistic image generation.

In this paper, we are providing GANs proposal background and development of text-to-image generation, including basic theory and related work of GAN in Section II. We see the theory of StackGAN in Section III. In Section IV, architecture of StackGAN is discussed, Implementation and Results are seen in Stage V and we conclude in Stage VI..

II. Theory and Related Works

A. Generative Adversarial Networks

Generative image modeling has been a fundamental problem for a long time and there has been exceptional progress with the emergence of deep learning. A significant development in this sector came with Variational Autoencoders (VAE) with probabilistic graphical models whose goal was to maximize the lower bound of data likelihood. Autoregressive models used neural networks model the distribution of pixels, thus generating synthetic images.

Recently, Generative Adversarial Networks have shown promising results in this field and are then used in attaining various challenges such as in the realistic image generation, character building, video generation, data augmentation etc. GANs consist of two neural networks, a Generator **G(x)**, and a Discriminator **D(x)**. They both play an adversarial game where the generator tries to fool the discriminator by generating data similar to those in the training set. The Discriminator tries not to be fooled by identifying fake data from real data. They both work simultaneously to learn and train complex data like audio, video or image files.

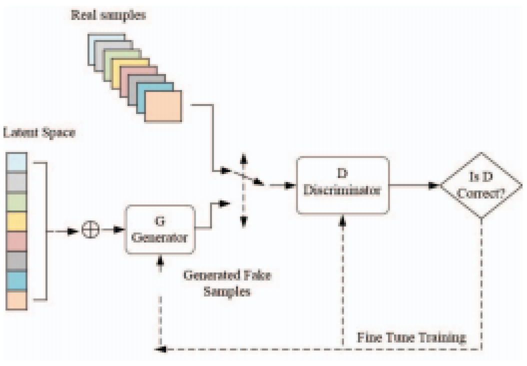


Figure 1. Structure of GANs

The generator model generates images from random noise(z) and then learns how to generate realistic images. Random noise which is input is sampled using uniform or normal distribution and then it is fed into the generator which generates an image. The generator output, which are fake images and the real images from the training set is fed into the discriminator that learns how to differentiate fake images from real images. The output D(x) is the probability that the input is real. If the input is real, D(x) would be 1 and if it is generated, D(x) should be 0.

B. *Text-to-Image Generation*

Recently, Generative Adversarial Networks have shown promising performance for generating sharper images. But the training instability makes it hard for GANs to generate high-resolution images. Several methods have been recently developed to generate images from unstructured text. Text-to-Image generation has been in the research hotspot for quite some time now. Mansimov et al. [2] built an AlignDRAW model by estimating alignment between text and the generating canvas. Reed et al. [3] used conditional PixelCNN to generate images using text descriptions and object location constraints. Nguyen et al. [4] used an approximate Langevin sampling approach to generate images conditioned on text. However, their sampling approach requires an inefficient iterative optimization process. With conditional GANs, Reed et al. [5] successfully generated plausible 64×64 images for birds and flowers based on text descriptions. Their follow-up work [6] was able to generate 128×128 images by utilizing additional annotations on object part locations. Because of the difficulty in generating high quality images there have been several approaches that involve multiple GANs to improve the quality of the image. Wang et al. [7] utilized a structure GAN and a style GAN to synthesize images of indoor scenes. Durugkar et al.[8] used multiple discriminators along with one generator to increase the chance of the generator receiving effective feedback. However all of these methods had their discriminator trained to approximate the image distribution at a single scale. The approach taken is definitely on the right track to break the high resolution image generation task into several easier sub tasks to be accomplished in multiple stages. The method we review here, StackGAN, directly generates high resolution images that are conditioned on their low-resolution inputs. StackGANs can also correct incoherent artifacts or defects in low resolution results by utilizing an encoder decoder network before the upsampling layers.

III. Stack GAN Theory

A. *Basic GAN Training*

GANs are composed of two models the Generator and the Discriminator that compete with each other to produce results. The Generator(G) tries to reproduce the true data distribution that is difficult for the discriminator to differentiate from the real images. The Discriminator D tries to distinguish between real images and synthetic images. The training is a minimax game between the two with the following objective function:

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

Here x is a real image and z is a noise vector sampled from prior distribution p (uniform or Gaussian distribution). In practice, the generator G is modified to maximize log(D(G(z))) instead of minimizing log(1 − D(G(z))) to mitigate the problem of gradient vanishing.

Conditional GANs [9], [10] are extensions of GANs 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.

B. *Conditional Augmentation*

With reference to Fig. 1, text description t is first encoded by an encoder, yielding a text embedding ϕt. we introduce a Conditioning Augmentation technique to produce additional conditioning variables cˆ. In contrast to the fixed conditioning text variable c, we randomly sample the latent variables cˆ from an independent Gaussian distribution N (μ(ϕt), Σ(ϕt)), where the mean μ(ϕt) and diagonal covariance matrix Σ(ϕt) are functions of the text embedding ϕt. The proposed Conditioning Augmentation yields more training pairs given a small number of imagetext pairs, and thus encourages robustness to small perturbations along the conditioning manifold.

To further enforce the smoothness over the conditioning manifold and avoid overfitting [11], [12], we add the following regularization term to the objective of the generator during training

$$D_{KL}\left(\mathcal{N}\left(\mu(\varphi_t), \Sigma(\varphi_t)\right) \parallel \mathcal{N}(0, I)\right)$$

which is the Kullback-Leibler divergence (KL divergence) between the standard Gaussian distribution and the conditioning Gaussian distribution

C. *Stack GAN - Stage I*

StackGAN involves two stages to generate the high resolution image. It decomposes the complex problem of text-to-image generation into two stages. Stage I 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, producing a low resolution image.

Let ϕt be the text embedding of the given description. The Gaussian conditioning variables cˆ0 for text embedding are sampled from N (μ0(ϕt), Σ0(ϕt)) to capture the meaning of ϕt with variations. Conditioned on cˆ0 and random

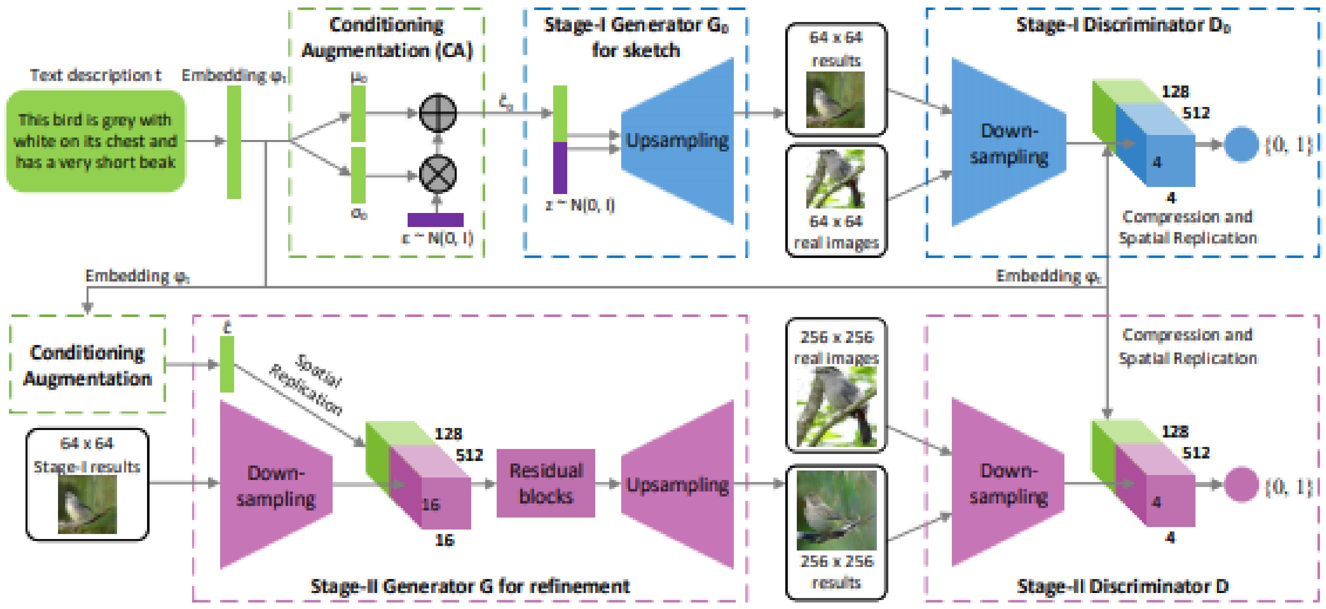


Figure 2. The architecture of the proposed StackGAN

variable z, Stage-I GAN trains the discriminator D0 and the generator G0 by alternatively maximizing LD0 in and minimizing as given below.

$$\mathcal{L}_{D_0} = \mathbb{E}_{(I_0,t)\sim p_{\text{data}}}[\log D_0(I_0, \varphi_t)] + \mathbb{E}_{z\sim p_z, t\sim p_{\text{data}}}[\log(1 - D_0(G_0(z, \hat{c}_0), \varphi_t))]$$

$$\mathcal{L}_{G_0} = \mathbb{E}_{z\sim p_z, t\sim p_{\text{data}}}[\log(1 - D_0(G_0(z, \hat{c}_0), \varphi_t))] + \lambda D_{KL}(\mathcal{N}(\mu_0(\varphi_t), \Sigma_0(\varphi_t)) \parallel \mathcal{N}(0, I))$$

where the real image I0 and the text description t are from the true data distribution pdata. z is a noise vector taken randomly.

D. Stack GAN - Stage II

Stage-II corrects defects in the low-resolution image from Stage-I and completes details of the object by reading the text description again, producing a high-resolution photo-realistic image. 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 s0 = G0(z, c^0) and Gaussian latent variables cˆ, the discriminator D and generator G in Stage-II GAN are trained by alternatively maximizing LD and minimizing LG as given below.

$$\mathcal{L}_D = \mathbb{E}_{(I,t)\sim p_{data}}[\log D(I, \varphi_t)] + \mathbb{E}_{s_0\sim p_{G_0}, t\sim p_{data}}[\log(1 - D(G(s_0, \hat{c}), \varphi_t))]$$

$$\mathcal{L}_G = \mathbb{E}_{s_0\sim p_{G_0}, t\sim p_{\text{data}}}[\log(1 - D(G(s_0, \hat{c}), \varphi_t))] + \lambda D_{KL}(\mathcal{N}(\mu(\varphi_t), \Sigma(\varphi_t)) \parallel \mathcal{N}(0, I))$$

Different from the original formulation of GANs, the random noise z is not used in this stage with the assumption that the randomness has already been preserved by s0. Gaussian conditioning variables cˆ used in this stage and cˆ0 used in Stage-I GAN share the same pre-trained text encoder, generating the same text embedding ϕt. However, Stage-I 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.

IV. Model Architecture

As seen in Fig. 2 the text description t is first encoded by an encoder resulting in a text embedding ϕt. Then conditional augmentation is carried out on the embedded text to produce additional conditioning variables c.

For the Stage-I generator G0, to obtain text conditioning variable c, the text embedding ϕt is first fed into a fully connected layer to generate μ and σ for the Gaussian distribution N(μ(ϕt), Σ(ϕt)) from which c is then sampled. The Ng dimensional conditioning vector cˆ0 is computed by cˆ0 = μ0 ⊖ σ0 (where ⊖ is the element wise multiplication, ~ N (0, I))

Then c^0 is concatenated with a Nz dimensional noise vector to generate a W × H image by a series of up-sampling blocks.

For the Stage-I discriminator D0, the text embedding ϕt is first compressed to Nd dimensions using a fully-connected layer and then spatially replicated to form a Md × Md × Nd tensor. Meanwhile, the image is fed through a series of downsampling blocks until it has Md × Md 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

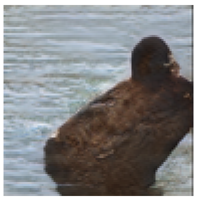
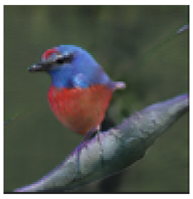
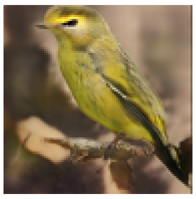
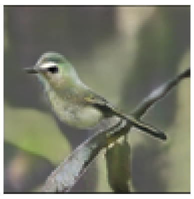
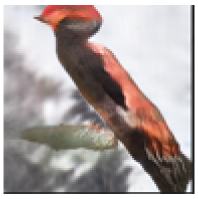
Text Description	This bird has a dark brown overall body color, with a small white patch around the base of the bill.	a small bird with blue wings and a red belly	this bird is short and stubby with yellow on its body	small white green and black beak with long black tarsus and medium beak	A large bird with head and body proportional with red wings
Stack GAN Stage 2					

Figure 3. Stack GAN stage 2 example result of images generate with the corresponding textual description given by the user.

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.

Stage-II generator is designed as an encoder-decoder network with residual blocks[13]. Similar to the previous stage, the text embedding ϕt is used to generate the Ng dimensional text conditioning vector cˆ, which is spatially replicated to form a Mg × Mg × Ng tensor. Meanwhile, the Stage-I result generated by Stage-I GAN is fed into several down-sampling blocks until it has a spatial size of Mg × Mg. The image features and the text features are concatenated along the channel dimension. 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. Finally, a series of up-sampling layers are used to generate a W × H high-resolution image.

For the Stage II discriminator, its structure is similar to that of StageI discriminator with only extra down-sampling blocks since the image size is larger in this stage. 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 embeddings, while the second is synthetic images with their corresponding text embeddings.

V. Implementation And Results

A. Implementation

Hyperparameters are set as: Ng = 128, Nz = 100, Mg = 16, Md = 4, Nd = 128, W0 = H0 = 64 and W = H = 256. For training we first train Stage-I Generator and Discriminator iteratively for 600 epochs by fixing Stage-II.

Then we train Stage-II Generator and Discriminator iteratively for 600 epochs by fixing Stage-I. We used ADAM [20] solver with default beta1=0.5. The batch size is 128, and the learning rate is initialized to be 0.0002 and decayed to 1/2 of its previous value every 100 epochs.

B. Results

As seen in Fig. 3 the results are astounding. Some are good enough to fool even the human eye. However there are some pictures that look bizarre. Further training and tuning can lead to better results. It is difficult to evaluate the performance of generative models, even with evaluation metrics only by looking at the results can we decide if the outcome has been successful or not. However we choose inception score (IS) [14] for quantitative evaluation.

Inception score is the first well-known metric for evaluating GANs. IS = exp (ExDKL (p (y|x) || p (y))), where x denotes one generated sample, and y is the label predicted by the inception model [15]. The intuition behind this metric is that good models should generate diverse but meaningful images. Therefore, the KL divergence between the marginal distribution p(y) and the conditional distribution p(y|x) should be large.For Stack Gan we obtained an inception score of: 3.45 ± .02

VI. CONCLUSION

In this paper, we review Stacked Generative Adversarial Network, named StackGAN, for better resolution text-to-image synthesis. First, we build a stage 1 generative network for the Stack GAN to generate low quality images through a multi-stage process. Second, we propose a deep multimodal similarity model to convert the low resolution image to a higher resolution image. Our Stack GAN significantly outperforms previous deep convolution GAN models using the birds dataset available on the Reed et al paper. Extensive experimental results clearly demonstrate the effectiveness of the proposed attention mechanism in the AttnGAN, which is especially critical for text-to-image generation for complex scenes.

Acknowledgment

This work is supported under CMR Institute Of Technology, Bangalore..

References

[1] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, Dimitris N. Metaxas "StackGAN++: Realistic

Image Synthesis with Stacked Generative Adversarial Networks," , IEEE, June 2018.

[2] E. Mansimov, E. Parisotto, L. J. Ba, and R. Salakhutdinov. Generating images from captions with attention. In ICLR, 2016.

[3] S. Reed, A. van den Oord, N. Kalchbrenner, V. Bapst, M. Botvinick, and N. de Freitas. Generating interpretable images with controllable structure. Technical report, 2016.

[4] A. Nguyen, J. Yosinski, Y. Bengio, A. Dosovitskiy, and J. Clune. Plug & play generative networks: Conditional iterative generation of images in latent space. In CVPR, 2017.

[5] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text-to-image synthesis. In ICML, 2016.

[6] S. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee. Learning what and where to draw. In NIPS, 2016.

[7] X. Wang and A. Gupta. Generative image modeling using style and structure adversarial networks. In ECCV, 2016.

[8] I. P. Durugkar, I. Gemp, and S. Mahadevan. Generative multi-adversarial networks. In ICLR, 2017.

[9] J. Gauthier. Conditional generative adversarial networks for convolutional face generation. Technical report, 2015.

[10] M. Mirza and S. Osindero. Conditional generative adversarial nets. arXiv:1411.1784, 2014.

[11] C. Doersch. Tutorial on variational autoencoders. arXiv:1606.05908, 2016.

[12] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In ICML, 2016.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In CVPR, 2016.

[14] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. In NIPS, 2016.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In CVPR, 2016.