iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 7 **Issue:** V **Month of publication:** May 2019

DOI: https://doi.org/10.22214/ijraset.2019.5534

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

Image Synthesis using Textual Descriptions

Husain Shaikh[1] Omkar Shelar[2] Omkar Waghmare[3] and Shubham Shende[4]
[1, 2, 3, 4]Vishwakarma Institute of Information Technology, Pune

Abstract: ***The synthesis of images using just detailed textual descriptions would be immensely useful.We propose a model wherein the entire synthesis would be achievable through a series of Neural Networks.To achieve this we would use a novel new method ,the Generative Adversarial Net[1], In this work, we develop a novel deep architecture and GAN formulation to effectively bridge these advances in text and image modeling, translating visual concepts from characters to pixels.[2] We demonstrate the capability of our model to generate lifelike images of natural phenomena such as sunsets,mountains using their detailed textual descriptions.A Generative Adversarial Net consists of two neural networks, a generator and a discriminator, where the generator tries to produce realistic samples that fool the discriminator, while the discriminator tries to distinguish real samples from generated ones.We further use the concept of STACK-GANs[2], to further enhance the resolution of the images.The STACK-GAN uses a two stage GAN to generate image . StackGAN-v1, to generate images from text descriptions through a sketch-refinement process. Low-resolution images are first generated by our Stage-I GAN . On top of the Stage-I GAN, we stack Stage-II GAN to generate high resolution (e.g., 256256) images. We use the same concept, to further enhance our generated images.[3] The main goal of the paper is to integrate the latest advancements, within this specific field and to generate undistinguishable images through the computer, given an appropriate input.***
Keywords: GANS, text to image , neural network, generative network, deep learning

I. INTRODUCTION

Giving computers the capability of generating images would open up new research and commercial opportunities in various domain that would require creativity. It would also help advance unsupervised learning.
Through the analysis of many viable techniques, as well as the implementation of existing techniques in modern environments, we hope to help, in the creation of visual stimuli, that is completely independent from normal human intervention.

II. RELATED WORKS

A. Generative Adversarial Network

Mathematically the whole adversarial game can be described using a single formula[1] :

$$\min G \max D V(D,G) = E_x P_{data}(x)[\log D(x)] + E_z P_{zz}[\log(1-D(G(z)))]$$

The aim is to minimize G(Generator) and Maximize D (Discriminator).The formula can be explained using two scenarios :

1) When the discriminator wins
2) When Generator wins

It is important to know that the discriminator outputs values between 0 and 1 It is also important to know that while taking the discriminator into consideration the whole formula is taken.whereas while taking generator into consideration only the second term of the formula is taken.
Maximum value : Infinity.
Minimum value : Zero.

B. Stage-1-GAN

The author divides the problem into two sub-problems. First, learn a text feature representation that captures the visual details. Second, use these features to synthesize a compelling image. The distribution of images based on text description is highly multimodal. There can be many possible configurations of pixels that correctly illustrate the description. The major contribution of the paper as stated by the authors is to develop a simple and effective GAN architecture and training strategy that enables compelling text to image synthesis of birds and flowers from human written descriptions. Dataset used : CaltechUCSD Birds dataset[4] and the Oxford-102 Flowers dataset[7] along with five text descriptions per image we collected as our evaluation setting. Output of the model : 64 X 64 images conditioned on text. From a distance the results
are encouraging, but upon close inspection it is clear that the generated scenes are not usually coherent. Successfully developed a model for generating images based on detailed visual description. Also generated images with multiple objects and variable background.

C. Stage-2-A STACK GAN

The author solves the problem of low resolution image generation by proposing new network architectures , introducing heuristic tricks or modifying the learning objectives . In [3] the author use two stage 2 GANS to generate image . StackGAN-v1, to generate images from text descriptions through a sketch refinement process. Low-resolution images are first generated by our Stage-I GAN . On top of the Stage-I GAN, we stack Stage-II GAN to generate high resolution (e.g., 256256) images. Dataset used : CUB , Oxford-102 and COCO datasets which contains 200 bird species with 11788 images Paper concludes by saying that generated images from interpolated embeddings can accurately reflect colour changes and generate plausible bird shapes. And can generate bird with complex colours with 256x256 resolution images. Also paper show we can implement two stack
GANS to improve the quality of image generated[3].

D. Progressive GAN

Progressive GAN[5] propose to start with training a generator and discriminator of 4 X 4 pixels after which it incrementally adds extra layers that doubles resolution up to 1024 X 1024. The paper concludes by saying that for textto-image synthesis, current methods work well on datasets where each image contains single object but performance on complex datasets is much worse. For image-to-image translation, the paper reviews some general methods from supervised to unsupervised settings, such as pixel-wise loss, cyclic loss and selfdistance loss.
In theory the usage of such a neural network would allow for the creation of completely high resolution imagery, indiscernible by the human eye.But no such research has been done on the integration of this the STACK GAN and the Progressive GAN.
All implementations are halted by the fact that the Progressive GAN cannot handle text embeddings while maintaining the quality of the picture.

E. Attentional GAN

The Attentional GAN[6] proposes an alternate method to image generation.Instead of focusing on encoding sentences as whole vectors the algorithm, takes each word and encodes it separately. Using this to fin-grained information of the individual words. and the concept of hidden features, which it uses to draw out sub-sections of the image corresponding to their respective words.
This focus on hidden features and extremely specific word vectors enable the Attentional GAN model to create iages which have highly pronounced colours and certain sub-sections. The main advantage being the extra high-detail for the same resolution of the image

III. DATASET AND FEATURES

To examine the Stack-GAN architecture, we ran experiments on the CaltechUCSD Bird (CUB) dataset[3] and Oxford-12 flowers dataset[7] . The CUB dataset conists of 200 different bird species and a toal of 11,788 images. Following the pre-processing step in [8], we cropped the images of all the birds so that they covered at least 75% of the total image size. The Oxford-102 dataset consists of 102 categories of flower species and a total of 8,189 images. In this case, the flowers make up a majority of the image area, and we therefore did not crop the images in any position. Each image in the datasets has with every image a collection of 10 captions.

IV. IMPLEMENTATION AND TRAINING

A. Pre-Processing Data

The GAN algorithm requires all inputs to be of a homogeneous numerical data format, we thus convert our inputs(images and captions) into numpy arrays. Two essential inputs are required for the algorithm to work:

a) The image as a numpy array

b) The caption with respect to every image encoded into a numpy array

Creating the Dictionary We first need to create key-value pairs of all distinct word, making sure that no word is left un-encoded, this is done by parsing through the entire caption database. Care must be taken to ensure that for any input caption, the system is robust enough to convert it, even if it contains words that were never encountered beforehand or special characters. A special keyword ”¡PAD¿” is also added to the dictionary after the entire process of key-value mapping is completed.This is done in order to maintain the final intended structure of our ”captions.npy” input file. The dictionary thus created can convert word to ids, reversing this dictionary another dictionary is created that can convert id to words,the key becomes the value and the value becomes the key.
Creating the Specialized Structure refer to the image above,we can see that the overall structure of the required numpy array is very specific. A numpy array of 3-Dimensional list is created with dimensions(7370 X 10 X 20). Where ”7370” refers to the total number

of images, ”10” refers to the number of captions per image and ”20” refers to the length of each caption. Each caption(of size 20) has an element of data-type ’numpy.str’ Using the dictionary that we have created in the previous step, we reference each word of a caption (key) and replace it with it’s respective number (value), and store it in a list. It is important to mention here that we need to keep the length of every caption the same (in our case 20). If the caption exceeds the limit of ’20’ words, we completely discard the caption, and if it does not exceed the size limit then we append the list (of caption of size 20) with the number (value) corresponding to the (key) ’PAD’. This final structure is then converted into a numpy array using ’numpy.save’ function available in the numpy module, and fed to our GAN.

B. Processing and Training

1) *Loading npy Files:* In the pre-training section, we created four numpy array files namely ”train image.npy”, ”train captions. npy”, ”Id2Word.npy”, ”word2Id.npy”. We load these files into the code.
2) *Flattening the Captions:* We know that our ”train caption.npy” file has a dimension of 7370 X 10 X 20. We need to convert the dimension to 36850 X 1 (7370*5=36850) because we do not need 10 captions per image as it will increase our training time significantly. Instead we found that 5 captions per image gave us optimal results. Hence we flattened out our initial numpy array
3) *Initializing Hyper parameters:* The following hyper parameters must be taken into account for the working of the GAN.

a) Learning Rate: Learning Rate[3] determines the rate at which we change the weights in our neural network, its dependent usually on the gradient descent algorithm.The new weight associated with our nodes,is determined by both gradient and learning rate. $New_weight = old_weight - LR * gradient$ It’s often very hard to get this right, normally the learning rate is set based on past experiences.The learning is responsible for the convergence of our neural network.

b) *LR Decay:* Suppose you’re implementing mini-batch gradient descent, with a reasonably small mini-batch.Your algorithm may just end up wandering around, and never really converge, because you’re using some fixed value for learning rate.To overcome this we will use the concept of LR decay, wherein as the number of iterations(epochs)increase, the learning rate will get smaller, such that the convergence is localized to a minima.And thus whatever inference is generated, will be higher in accuracy. Here we use exponential LR decay,such that our neural network, attains a very low value of LR in later epochs for a much more accurate result.

c) *Z-Noise:* Here the z-noise as in [2] is the random noise sent to the generator as the basis for the creation of the image.This is done through the use of a n-dimensional numpy array, wherein this random noise is combined with the encoded captions, in the same array and this whole n-dimensional array is upsampled and sent to the discriminator. This n-dimensional array containing the noise is then modified to fit the model better, through these repeated modifications, we are left with noise that has been converted into a viable output.

4) *Discriminator Training:* This is one half of our neural network, the discriminator has the sole objective of classifying whether the image generated belongs to our dataset or not(Outputs 1 if image is viable; 0 otherwise).The discriminator contains what is commonly referred to as a Convolutional Neural Network(CNN).This CNN is trained on both actual images from our dataset(Oxford-Flowers)[7] and the generated images created from scratch by the generator.The input provided to this CNN[8] is of three combinations.

a) Real Image + Real Caption
b) Fake Image + Real Caption
c) Fake Image + Fake(Wrong) Caption

This allows the discriminator to be well versed with both the actual solution, but also be flexible enough that, the generator can fool it if a good enough image is generated.Under no circumstances do we want a Discriminator, that is too powerful, because in that scenario,the generator will never win, so no images will be created.

This is done to ensure that the Nash Equilibrium is kept.That is the Generator and Discriminator, are competing in a game where they are both equally powerful.

5) *Generator Training:* As discussed above the generator uses the Z-Noise to generate images through an n-dimensional array.This array is sent through multiple up sampling blocks. Finally resulting in an image(Here 64x64 in size).This image is sent to the discriminator which classifies it as 0 or 1. If the output gained is 1, then the generator has been successful, the image is viable.But if the output is 0, then discriminator will send a feedback through a built-in back propagation algorithm to change the weights in such a way so as to improve the generated image. Through thousands of such iteration wherein, both the generator and discriminator will get better through each epoch, we train the GAN as in [3] as a whole.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

V. EXPERIMENTS AND RESULTS

The GAN was run across an AWS server wherein the spec sheet of the system was as follows:
This allowed us to train the neural network across a period of approximately 14 hours, on the GPU,ending with the creation of a collection of 64 images for each epoch.The dataset used was the Oxford-Flowers dataset.Initally the main bottleneck had been the processing time, which was overcome through the use of GPU processing.
The following images are representative of the output across multiple epochs.

VI. CONCLUSIONS

Generative Adversarial Networks next frontier in artificial intelligence and we are moving towards it. Giving computers the capability of generating images would open up new research and commercial opportunities in various domain that would require creativity. It would also help advance an amalgamation of supervised and unsupervised learning.

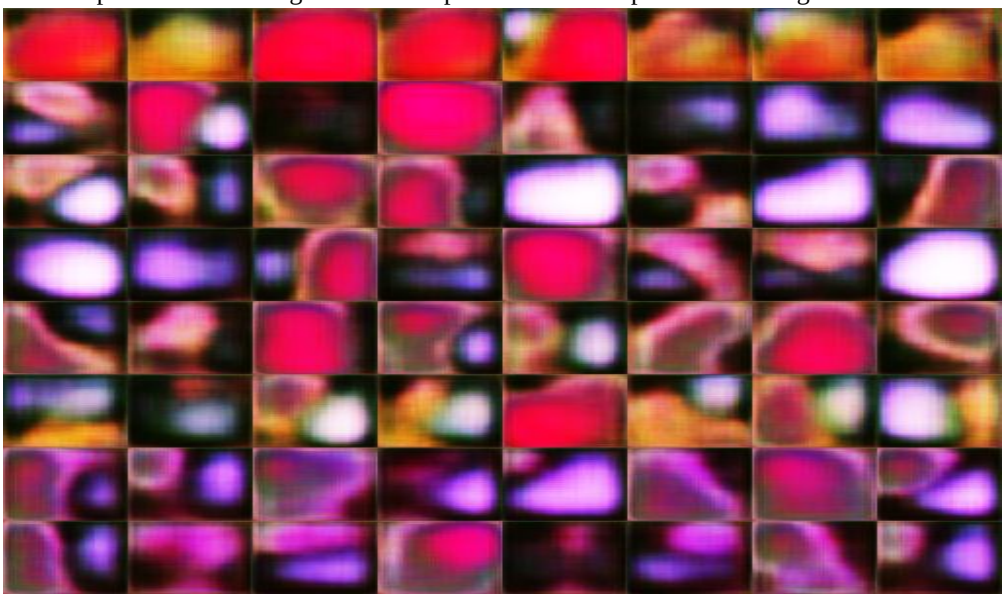


Figure 1: Epoch-1-Flowers

Figure 2: Epoch-50-Flowers

Figure 3: Epoch-200-Flowers

Figure 4: Epoch-599-Flowers

VII. FUTURE WORK

We have been trying to extend this algorithm to become much more diverse, our current aim is to generalize the technique, to work on a large variety of images.

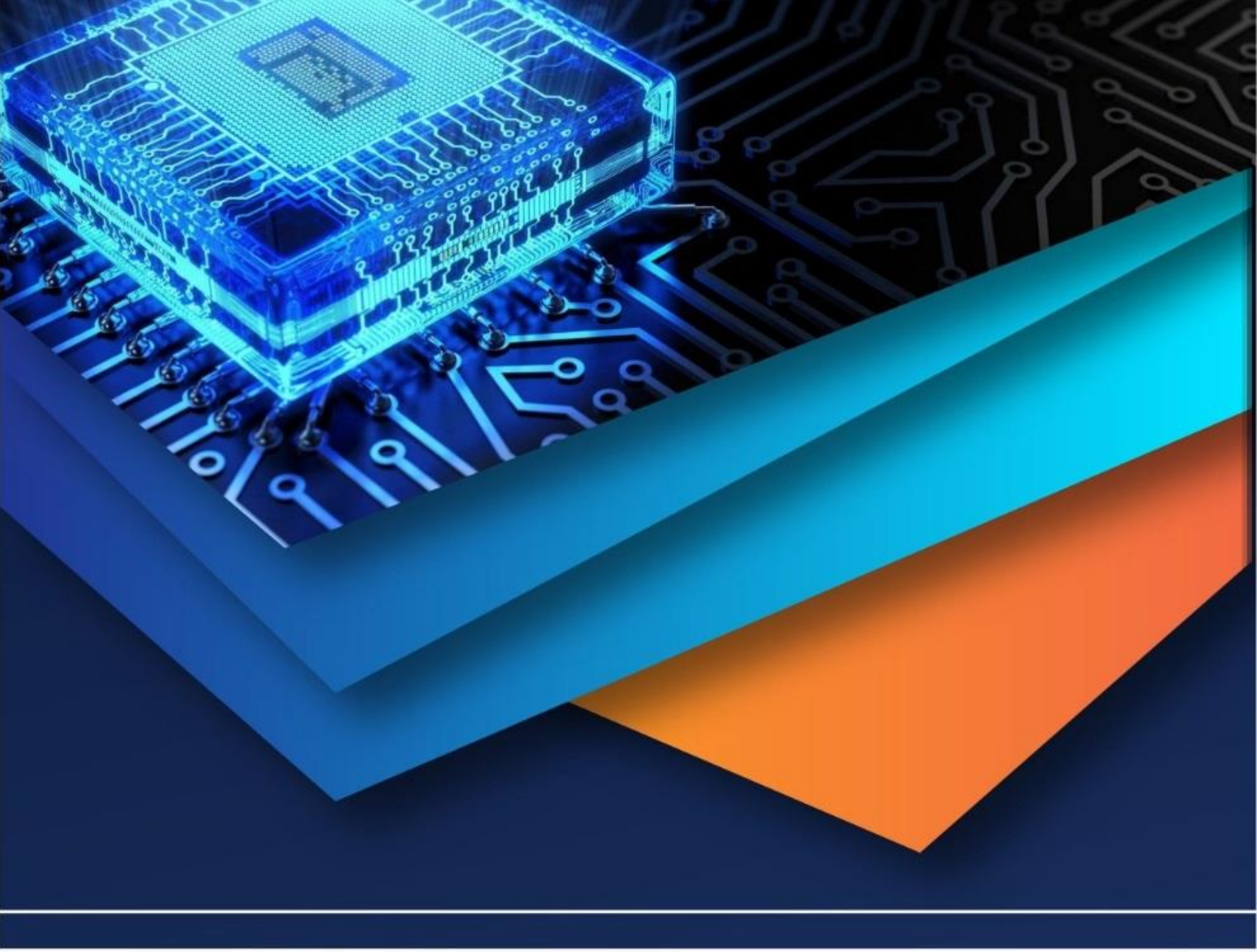
Through the use of Google Concptual Captions[9], we hope to create a modified algorithm that would be able to generate, images of a variety of objects seamlessly.Our current work has been not as successful herewith wherein we have encountered a whole slew of problems trying to train through this extended dataset.

VIII. ACKNOWLEDGMENT

It is matter of great pleasure for us to submit this report on ”IMAGE SYNTHESIS USING TEXTUAL DESCRIPTION”, as a part of curriculum for Bachelor in Engineering (Computer Engineering) of University of Pune, We are thankful to my guide Assistant Prof. S.A.Tiwaskar in Computer Engg Department for his constant encouragement and able guidance. We are also thankful to Dr.B.S.Karkare, Director of VIIT Pune, Dr. S.R. Sakhare Head of Computer Department for their valuable support. We take this opportunity to express our deep sense of gratitude towards those, who have helped us in various ways, for preparing our project.

REFERENCES

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In NIPS, 2014.
[2] Generative Adversarial Text to Image Synthesis Scott sReed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran,arXiv:1406.2661v1 [stat.ML] 10 Jun 2014
[3] StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Senior Member, IEEE, Xiaogang Wang, Member, IEEE, Xiaolei Huang, Member, IEEE, Dimitris N. Metaxas , Fellow, IEEE
[4] Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The caltech-ucsd birds-200-2011 dataset. 2011.
[5] Progressive Growing of GANs for Improved Quality, Stability, and Variation Tero Karras, Timo Aila, Samuli Laine, Jaakko Lehtinen
[6] AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, Xiaodong He
[7] Oxford Flowers Dataset Visual Geometry Group Department of Engineering Science, University of Oxford
[8] Understanding of a convolutional neural network 3 Author(s) Saad Albawi ; Tareq Abed Mohammed ; Saad Al-Zaw,IEEE,08 March 2018
[9] Conceptual Captions: A Cleaned, Hypernymed, Image Alt-text Dataset For Automatic Image Captioning,Sharma, Piyush and Ding, Nan and Goodman, Sebastian and Soricut, Radu,Proceedings of ACL,2018



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY