International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-6, August, 2019

Generation of Facial Drawings Using Generative Adversarial Networks

Debabrata Datta, Abhradeep Dey, Adityam Ghosh, Rishabh Tiwari

Abstract*: *Facial sketches are widely used in judicial and legal proceedings. Law enforcers use facial sketches to help them with the visual aspects of the case, using witness descriptions and video footage. However, drawing forensic sketches by hand is a time-consuming procedure and a situation may arise where the authorities have less time in hand to solve a case. The present research work aims to create a basic model which can generate facial images from a given set of input features; similar to what a forensic artist does, thus, enabling a faster and efficient sketching procedure. In this work, a category of generative algorithms, called Generative Adversarial Networks has been used to build this model. To train this model, a dataset of anime girls has been used and thus it can only generate the same, making sure that the generated image contains the input features.

Index Terms*: *Artificial Intelligence, Auxiliary Classifier Generative Adversarial Network, Deep Learning, Face Generation, Forensic Art, Generative Adversarial Network, Machine Learning, Neural Networks.

I. INTRODUCTION

The recent times have seen an increase in criminal activities such as assault, robbery, kidnapping etc. throughout the world, as shown by the Crime Data of the United Nations Office on Drugs and Crime (UNODC) [3]. In order to assist the law enforcement agencies by making the procedure of processing the evidence, extracting clues, and producing them in court fast and accurate, a large number of research projects have been undertaken. For example, biometric scanners, and face recognition algorithms to match faces from a database are widely used in investigation and prosecution procedures. Sketching faces based on given facial features is one of the applications where technology has been helping the law enforcement agencies.

Traditionally, forensic sketches are hand drawn based on the details given by various eyewitnesses. However, when it comes to the field of forensic arts, plenty of people are not willing to work here and thereby leaving a handful of skilled artists available for the investigation in the crime branch. Most of the artists working in this area are either freelancers hired by departments or active-duty officers or other agency employees who are called in when needed. Moreover, traditional methods are very time-consuming. When creating a face model, the forensic sartist looks at whether the person is masculine or feminine, as well as their skin tone, age, wrinkles, freckles, the shadow of the beard, and attractiveness. But most importantly, they look for the descriptions of the eyes and hair of the suspect. In case of eyes, they look for the direction of the eyeballs, their colour, position and shape of eyebrows, and the distance between the eye sockets. In the case of hair, they look for their length, colour, and other features. With this view in mind, the present research work has been designed on the generation of face images using the application of Generative Adversarial Networks or GANs [1] to analyse the features (viz. hair colour and eye colour) given as input by the user and then generate an image of that person. A variation of GAN, known as Auxiliary Classifier GAN or ACGAN [4] has been used here and this enables the work to take feature inputs and generate images based on the inputs. For training and testing, a dataset of anime girls from Getchu [17] has been used. Thus, the present work can only generate faces of anime girls with this model. This is a basic model which can be improved later by using a dataset of real people and including more facial features.

II. BACKGROUND STUDY

Researchers have developed tools to create composite sketches based on the description of an eyewitness, in the last few years. These sketch generation tools, for example, Identi-Kit 7 HD [2], are fast and the law enforcement agencies now often utilize them for creating the composite sketches. In contrast to hand-drawn sketches, composite sketches enable quick feedback from the eye-witness during the process of sketch generation. But tools like these have a learning curve, and a person who wants to use the software have to learn using it at first.In the past few years, various researches have been done related to the generation of face images based on user inputs. An anime girl face generator was made using DRAGAN [6] which can generate random face drawings based on various facial features. The research work described in this paper has aimed to develop a similar model which can be used as a stepping stone to automate the process of facial image generation with the help of Auxiliary Classifier GAN or ACGAN, thus, eliminating the learning curve and enabling people,

Revised Manuscript Received on October 30, 2019.
* Correspondence Author
Debabrata Datta*, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata, India.
Abhradeep Dey, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata, India.
Adityam Ghosh, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata, India.
Rishabh Tiwari, Department of Computer Science, St. Xavier's College(Autonomous), Kolkata, India.

irrespective of technical expertise, to get facial sketches drawn based on a number of given features.

Figure 1 shows the flowchart of Generative Adversarial Networks which are composed of two neural networks.

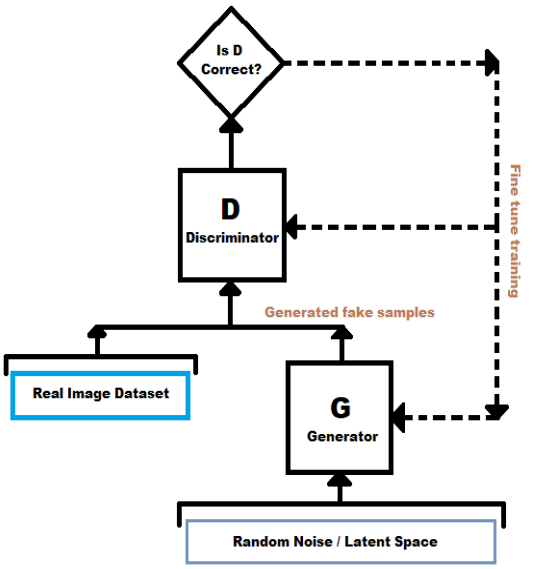

Figure 1: The Model of a Generative Adversarial Neural Network

The neural network $G(z;\ \theta_g)$ is used to model the generator [1]. Its role is mapping input noise variables $\boldsymbol{z}$ to the desired data space $\boldsymbol{x}$. Here θ_g are the parameters of the generator. Conversely, a second neural network $D(x;\ \theta_d)$ models the discriminator[1] and outputs the probability that the data came from the real dataset, in the range (0,1). Here, θ_d are the parameters of the discriminator. As a result, the discriminator is trained to correctly classify the input data as either real or fake. For this, an equation is used and this also termed as the loss/error function which is given as follows [1]:

$$\min_{(G)}\ \max_{(D)} V(D,\ G) = E_{x\sim pdata}[\log(D(x))] + E_{z\sim pz}[\log(1-D(G(z)))]. \quad (1)$$

During training, both the discriminator and generator are trying to optimize opposite loss functions, they can be thought of two agents playing a minimax game with value function $V(D, G)$.

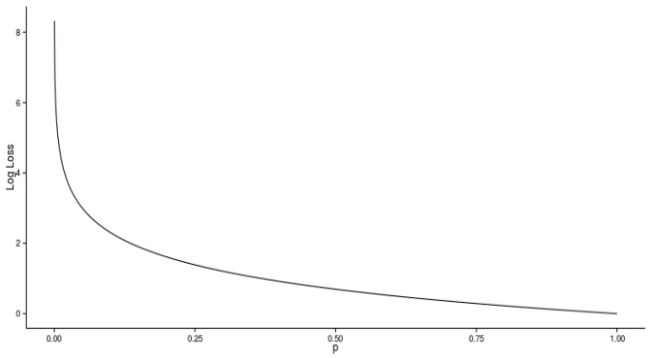

Figure 2: The Log Loss Graph; low probability values are highly penalized [9]

In this minimax game, the generator is trying to maximize its probability of having its outputs recognized as real, while the discriminator is trying to minimize this same value. In practice, the logarithm of the probability is used in the loss functions instead of the raw probabilities, since using a log loss heavily penalizes classifiers that are confident about an incorrect classification, as shown in Figure 2.

III. PROBLEM DESCRIPTION

The basic concept of this research work is to take as input the textual description of a person, which in this case is an anime image and to generate five samples of random images from that random noise which will be similar to the textual description provided by the user.

Generative Adversarial Networks have been the state of the art for random generation of images. It is a modified functional model of a Variational Auto-Encoders (VAEs) [15]. With a less amount of training and a bit of tuning, the GAN can generate better images than a VAE. Thereby for this research work, the Auxiliary Classifier Generative Adversarial Networks (ACGANs) have been implemented to generate images for labelled generation.

ACGAN was introduced by AgustusOdena*et al* [4]. In the ACGAN, every generated sample has a corresponding class label, $c \sim p_c$ in addition to the noise z, both of which are used by the generator G to generate images $X_{fake} = G(c,z)$.The discriminator D gives both a probability distribution over sources and a probability distribution over the class labels, $P(S \mid X)$, $P(C \mid X) = D(X)$. The objective function has two parts: the log-likelihood of the correct source, L_S, and the log-likelihood of the correct class, L_C. The following two equations express L_S and L_C respectively [4]. They are given by:

$$L_S = E[\log P(S = real \mid X_{real})] + E[\log P(S = fake \mid X_{fake})] \quad (2)$$

$$L_C = E[\log P(C = c \mid X_{real})] + E[\log P(C = c \mid X_{fake})] \quad (3)$$

D is trained to maximize (LS + LC) while G is trained to maximize (LC − LS). ACGANs learn a representation for z that is independent of class label [15].

Thus, the pseudo code of the proposed methodology is as follows:

begin:

X ← data for training

Y ← labelled description of the data

generator ← Stacked model of Dense and Transposed Convolution layers (up to 4 layers with stride value of 2 and kernel size of 3) and Batch Normalization

discriminator ← Stacked model of Dense and Convolution Layers (up to 4 layers with stride value of 2 and kernel size of 3) and Maxpooling and Batch Normalization

combined ←

Stacked model of Generator and Discriminator

Training:

Generate random noise and sampled labels for that noise.

Train the discriminator to maximize $L_S + L_C$.

After certain epochs, stop discriminator training and train the generator to maximize $L_C - L_S$.

Stop the training when a convergence position is reached.

end training end

IV. IMPLEMENTATION

The proposed algorithm for the research work was implemented using Python 3.6.810 as the programming tool and for the machine learning libraries, TensorFlow 1.11 (GPU version)11 was used. Keras12 was also used for its high-level API with TensorFlow as its backend. Other dependencies that were included were NumPy13 and Matplotlib14. In order to decrease the time taken for training the model, by taking the full advantage of the parallel processing capabilities of the GPU, the CUDA® parallel computing platform7 was used. For GPU-accelerated library of primitives for deep neural networks and highly tuned implementations for standard routines such as forward and backward convolutions, pooling, normalization, and activation layers, the NVIDIA CUDA® Deep Neural Network (cuDNN)8 library was used.

The application was trained and tested in a computer with an Intel i5 8th generation processor, 8 Gigabytes of RAM, 1 Terabyte of hard drive space, and a Nvidia GTX 1050Ti graphical processing unit which had a computation capability of 5.2.

The very first phase of the implementation of the proposed method was to collect the dataset which was downloaded from Brian Chao's Github repository [16]. The dataset consists of 21,551 anime faces scraped from Getchu [17], which are then cropped using the anime face detection algorithm lbpcascade_animeface [18] has been used for this research work. All images are resized to 64 × 64 pixels.

The different features like hair color and eye color were defined based on which feature would be considered to differentiate the pictures in the dataset. In order to convert the features of different pictures into matrices, the pictures were tagged based on different features. This technique was implemented by taking a finite set of different variants of hair and eyes as mentioned in table 1 and table 2.

Table 1: The First Set Of Different Variants Of Hair And Eye Colours Used To Differentiate Between The Pictures In The Dataset

Hair	Orange	White	Aqua	Gray	Green	Red
Eyes	Gray	Black	Orange	Pink	Yellow	Aqua

Table 2: The Second Set Of Different Variants Of Hair And Eye Colours Used To Differentiate Between The Pictures In The Dataset

Hair	Purple	Pink	Blue	Black	Brown	Blonde
Eyes	Purple	Green	Brown	Red	Blue	

Then every image in the dataset has been iterated through comparing their features with the finite set already defined. When a successful match was found, the index of that feature was put in the finite dataset and was used for tagging the array for the respective image.The next phase of the implementation was training the model. The model consisted of two neural networks, the generator and the discriminator. The generator was designed with four transposed convolution layers and a dense layer whereas the discriminator was designed with convolution layers and maxpooling layers. Both the neural network models had used the Adam Optimizer as the optimization function [22].

Figure 3: Face Images Generated At The End Of 10,000 Epochs

During training, when the generator was trained, the discriminator was not trained and vice versa. The generator's work was to use random noise to generate images and then it was mapped with the sampled labels which were then fed into the discriminator. The discriminator and the generator were trained to tally against each other and then come down to a convergence point. Once the convergence point was reached, the training was stopped.

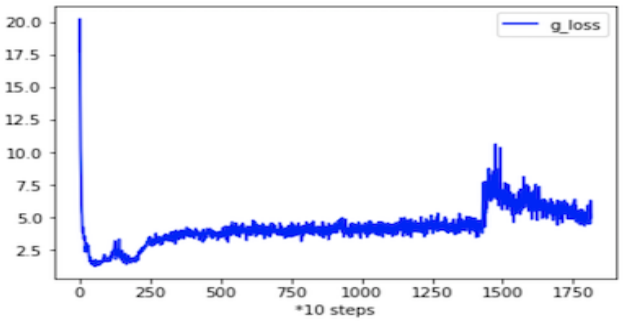

Figure 4: The loss function graph of the Generator Model

The proposed model was trained for 25,000 epochs to generate images each of resolution 64 × 64 pixels. Figure 3 shows the images generated at the end of 10,000 epochs. Figure 4 depicts the loss function graph of the generator model. It is observed from the graph that the loss function is minimized because during compilation of the application, the negative value of the generator loss function $(L_C - L_S)$ was taken into account and thereby generating a minimization graph after the complete epoch.

On the other hand, figure 5 depicts the loss function graph of the discriminator model. It is observed again that the loss function is minimized because during compilation of the application, the negative value of the loss function $(L_S + L_C)$ was chosen and thereby generating a minimization graph after the complete epoch.

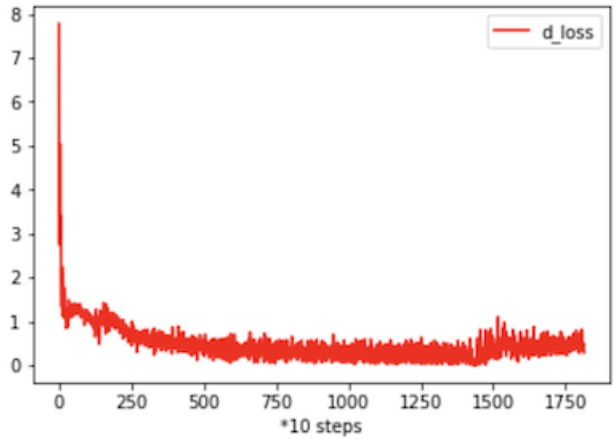

Figure 5: The Loss Function Graph Of The Discriminator Model

The model was then tested by taking random inputs from the users and then five samples of images based on the given hair color and eye color were generated for every test input. Figure 6 (a, b, c, d) shows different generated images based on different inputs.

(a)

Figure 6: Samples Of ((d) Images Generated Under User-Defined Conditions.

(a) black hair and blue eyes
(b) red hair and green eyes
(c) white hair and different colours of eyes
(d) pink hair and different colours of eyes

V. FUTURE IMPROVEMENTS AND CONCLUSION

In the present research work, the automatic generation of

the anime character has been explored. By combining a clean dataset with GAN training strategies, a model was successfully built and the model can generate facial images of anime characters. However, this model has some constraints which are mentioned below:

The primary constraint is that only anime girls could be generated as the dataset used here was only of anime girls. Though with minor modifications in the algorithm and using a dataset of real face images would help the model generate images of real people based on their features.

The next constraint is related to the features which were considered here. For simplification, only the hair color and eye color were chosen as the inputs to generate an image. However, this model can be improved to take more features, like, the position of the mouth, spectacles, skin tone, hair length, etc., as various inputs. Incorporating more features would help the model to generate images which would have been more accurate to the description.

Finally, it was observed that the output images were of much lower resolution. This was because of the absence of SRGAN. To improve the quality of the image, this model can be trained along with the SRGAN [19] or the SRCNN [20] system which was optimally designed for high-quality resolution image generation. Moreover, as the quality or the resolution of the image increases, its distinguishable features have also increased. Thus, the discriminator can differentiate between a fake image and a real image with a higher resolution facility. DiscoGAN [21] can also be applied for different image perspective.Thus, if all these improvements can be made to this model, this model can be even better than what it has been put forward here and can even be used in real-life situations, and help in the judiciary and legal proceedings.

REFERENCES

1. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, SherjilOzair, Aaron Courville, YoshuaBengio, "Generative Adversarial Networks", In Proceedings of the 27th International Conference on Neural Information Processing Systems – Volume 2, pp. 2672 – 2680, December, 2014.
2. Identi-Kit Solutions | Advanced Facial Composite Software. http://identikit.net/, last accessed on 09.06.19, 8:30 pm.
3. Crime Data | Statistics and Data.https://dataunodc.un.org/crime, last accessed on 25.06.19, 8:00 pm.

4. Augustus Odena, Christopher Olah, Jonathon Shlens,"Conditional Image Synthesis with Auxiliary Classifier GANs", In Proceedings of the 34th International Conference on Machine Learning, pp. 2642 – 2651, 2017.
5. Naveen Kodali, Jacob Abernethy, James Hays, Zsolt Kira. On Convergence and Stability of GANs. arXiv:1705.07215v5.
6. YanghuaJin, Jiakai Zhang, Minjun Li, Yingtao Tian, Huachun Zhu, Zhihao Fang. Towards the Automatic Anime Characters Creation with Generative Adversarial Networks.arXiv:1708.05509v1.
7. CudaParallel Computing Platform for Developers | NVIDIA, https://www.nvidia.in/object/cuda-parallel-computing-in.html, last accessed on 26.06.19, 8:40 pm.
8. NVIDIA cuDNN | NVIDIA Developer, https://developer.nvidia.com/cudnn, last accessed on 26.06.19, 8:40 pm.
9. Diego Gomez Mosquera. GANs from Scratch 1: A deep introduction. With code in PyTorch and TensorFlow,https://medium.com/ai-society/gans-from-scratch-1-a-deep-introduction-with-code-in-pytorch-and-tensorflow-cb03cdcdba0f, last accessed on 26.06.19, 9:00 pm.
10. Python Release Python 3.6.8 | Python.org, https://www.python.org/downloads/release/python-368/, last accessed on 27.06.19, 7:00 pm.
11. TensorFlow,https://www.tensorflow.org/, last accessed on 27.06.19, 7:00 pm.
12. Home – Keras Documentation,https://keras.io/, last accessed on 27.06.19, 7:15 pm.
13. NumPy – NumPy,https://www.numpy.org/, last accessed on 27.06.19, 7:15 pm.
14. Matplotlib: Python plotting — Matplotlib 3.1.0 documentation, https://matplotlib.org/, last accessed on 27.06.19, 8:15 pm.
15. Diederik P Kingma, Max Welling,"Auto-Encoding Variational Bayes", In Proceedings of the 2nd International Conference on Learning Representations, 2014.
16. Brian Chao (Mckinsey666), "Anime-Face-Dataset",https://github.com/Mckinsey666/Anime-Face-Dataset, last accessed on 28.06.19, 6:15 pm.
17. Getchu.com,http://www.getchu.com/, last accessed on 28.06.19, 6:30 pm.
18. nagadomi.lbpcascade_animeface, https://github.com/nagadomi/lbpcascade_animeface, last accessed on 28.06.19, 6:40 pm.
19. Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, AlykhanTejani, Johannes Totz, Zehan Wang, Wenzhe Shi,"Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, July, 2017.
20. Chao Dong, Chen Change Loy, Kaiming He, Xiaoou Tang,"Image Super-Resolution Using Deep Convolutional Networks",IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 38, Issue 2, pp. 295 – 307, June, 2015.
21. Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, Jiwon Kim,"Learning to Discover Cross-Domain Relations with Generative Adversarial Network", In Proceedings of the International Conference on Machine Learning, Volume 70, pp. 1857 – 1865, August, 2017.
22. Diederik P. Kingma, Jimmy Ba. Adam: A Method for Stochastic Optimization, In Proceedings of the 3rd International Conference on Learning Representations, May, 2015.

AUTHORS PROFILE

Mr. Debabrata Datta pursued Master of Technology from University of Calcutta, India and he is currently pursuing his Ph.D. in Technology from the same university. He is an Assistant Professor in the department of Computer Science, St. Xavier's College (Autonomous), Kolkata, India He is a life member of IETE. He has published more than 25 research papers in different reputed international journals and conferences. His main research interest is in the field of Data Analysis. He has more than 11 years of teaching experience and has more than 5 years of research experience.

Mr. Abhradeep Dey has completed his B.Sc. with honours in Computer Science from St. Xavier's College (Autonomous), Kolkata, India.

Mr. Adityam Ghosh is a graduate in Computer Science from St. Xavier's College (Autonomous), Kolkata, India. He has published a research paper on recommender system in a reputed international journal.

Mr. Rishabh Tiwari has completed his B.Sc. with honours in Computer Science from St. Xavier's College (Autonomous), Kolkata, India.