Fake Face Generator: Generating Fake Human Faces using GAN

Md. Mahiuddin[1], Md. Khaliluzzaman[2], Md. Shahnur Azad Chowdhury[3], Muhammed Nazmul Arefin[4]
Dept. of Computer Science and Engineering, International Islamic University Chittagong (IIUC) [1, 2, 4]
Chattogram- 4318, Bangladesh[1, 2, 4]
Dept. of Computer Science and Engineering, Chittagong University of Engineering and Technology[1]
Chattogram- 4349, Bangladesh[1]
Dept. of Business Administration, International Islamic University Chittagong (IIUC), Chattogram- 4318, Bangladesh[3]

***Abstract*—As machine learning is growing rapidly, creating art and images by machine is one of the most trending topics in current time. It has enormous applications in our current day to day life. Various researchers have researched this topic and they try to implement various ideas and most of them are based on CNN or other tools. The aim of our work is to generate comparatively better real-life fake human faces with low computational power and without any external image classifier, rather than removing all the noise and maximizing the stabilization which was the main challenge of the previous related works. For that, in this paper, we tried to implement our generative adversarial network with two fully connected sequential models, one as a generator and another as a discriminator. Our generator is trainable which gets random data and tries to create fake human faces. On the other hand, our discriminator gets data from the CelebA dataset and tries to detect that the images generated by the generator are fake or real, and gives feedback to the generator. Based on the feedback the generator improves its model and tries to generate more realistic images.**

***Keywords*—Generative adversarial network; fake human faces; generator; discriminator; CelebA dataset**

I. Introduction

Fake images can be used by police to track down people who are involved in adultery. As we know nowadays child and underage pornography is alarming. To catch such people we can use computer generated fake faces, that will be more ethical than using real images [11]. As we can see nowadays creating fake or edited images is very easy. To identify these kinds of images, we can build a model. For this model to be trained we can use the computer generated images to identify the real and fake images. We use the fake computer generated images in visual art and the advertising industries. These images have a huge role in computer games. In computer and mobile games along with play station games, these images can be used. As augmented reality and virtual reality are growing rapidly, we can use these computer generated fake images.

II. Literature Review

There are many fake image generation models created or updated by the researchers which generate pretty good images. Zhang et al. [1] in 2020 discussed an algorithm that can improve the quality of generated images as the quality of the fake images generated by the conventional GAN is limited. Wang et al. [2] in 2016 tried to simplify the overall process of generative models which gave them more realistic high resolution images as well as highly stable and robust learning procedures. Ghatas et al. [3] in 2020 proposed a method of building a complex modular pipeline using previously trained models to generate the kin image. They tried to build such a model where it changes the way of approaching GAN problems.

Hamdi et al. [4] in 2019 proposed a new GAN that uses the K-nearest neighbor for selective feature matching in a high level space and they named their work as K-GAN. Tolosana et al. [5] in 2020 tried to recognize fake news, fake images, fake media, and face manipulation using their method. They provided a thorough review of digital manipulation techniques which were applied to facial content due to the huge number of possible detrimental applications.

Karras et al. [6] in 2019 proposed a method that re-designs the generator architecture so that we can understand the various aspects of the image synthesis process. Though GAN has seen rapid improvement in recent years, we still have a very poor understanding of the properties of the latent spaces. They tried to understand the inner workings of the latent spaces. Their work starts learning from constant input and tries to adjust the “Style” of the image on each layer. Their method was tested on Flickr-Faces-HQ and FFHQ datasets.

Zhao et al. [7] in 2019 proposed an image translation network by exploiting attributes with the generative adversarial network. It can remarkably contribute to the seven authenticity of the generated face by supplementing the sketch image with the additional facial attribute feature. The generator and discriminator both use skip-connection to reduce the number of layers without affecting network performance. In the underlying feature extraction phase this network is different from most attribute-embedded networks. They divided their networks into two parts as sub-branch A and sub-branch B, which takes a sketch image and attribute vector to extract low level profile information and high level semantic features.

In recent years, Generative Adversarial Networks have achieved extraordinary results for various applications in many fields especially in image generation because of their ability to create sharp and realistic images. In this paper, Shirin Nasr Esfahani et al. [8] discussed five areas of image synthesis

based on different techniques of GAN. They are: Text to Image Synthesis, Image to Image translation, Face Manipulation, 3D Image Synthesis and Deep Master Prints. In this paper they have actually tried to focus on the applications of the above techniques and discussed their merits and demerits and future applications. An Introduction to Image Synthesis with Generative Adversarial Nets is proposed in [10]. This model is working well for the simple dataset, for the complex dataset the model does not work well.

Though there are many fake image generation models, what we have tried to do in our work is that we have tried to generate the images using low computational power and without any external image classifier. We have used two neural networks in our model. One is Generator and the other is Discriminator. Generator creates fake human faces and Discriminator detects the faces generated by the generator if this is fake or real. If the face generated by the generator is detected as fake then the discriminator gives feedback to the generator. And based on the feedback of the discriminator, the generator improves itself and tries to generate better quality faces. This process goes on until the generator creates better quality fake human faces. From detecting fake news, fake images to data misclassification, fake human face generation helps us to be used in all of these areas. And generative adversarial network model is one of the best one to create better real life fake human faces.

Rest of the paper is organized as follows. The proposed method is explained in Section III. In Section IV, the experimental results are presented. The paper is concluded in Section V.

III. Proposed Method

A. Methodology

In our model we have tried to implement generative adversarial networks in a very classic way with new and latest tools and technologies. We have tried to implement a generative adversarial network to generate fake human faces from random noise with low computational power and without any external image classifier, rather than removing all the noise and maximizing the stabilization which was the main challenge of the previous related works. We know that for implementing generative adversarial network we have to set a generator whose work will be to generate fake faces from the noise and update its own model after receiving the feedback from the discriminator and we also have to implement a proper discriminator to identify the real and fake image, so that it give the right feedback, means it says true to real image and false to fake image and based on its feedback the generator will change its model in such a way that eventually it will create almost real like images.

That's how we are trying to implement our generative adversarial network to generate fake human faces. Our generator and discriminator will use the keras's classic sequential modeling.

We proposed to use two sequential models, one as a generator and another one as a discriminator to implement our generative adversarial model. For implementing the generator, as we can see from the workflow diagram, we have used random noise as input data for the generator. We have also used the Batch Normalization technique and the Reshape module from the keras to resize the data and normalize to train the generator properly. In our generator we have used a total of five layers, one is an input layer with 128 nodes, starting with three more dense layers 512, 1024 and 2048 nodes respectively with each layer and lastly the output layer. We have used the Leaky ReLU activation function for all the layers except the output layer; in the output layer we have used the tanh activation function for the output layer. We have also used the loss function as the binary cross-entropy for our generator model. Our generator will receive feedback data from the discriminator and based on the received feedback our generator will be updated.

On the other hand, our discriminator is also implemented with the sequential model. However, the main difference between our generators is that after each epoch it does not learn, meaning it will not update or train after each epoch. Our discriminator's main purpose is to identify the real and fake image and give the feedback to the generator, so that we will use the sequential model. In our discriminator's input data we have used both the generator produced data and the real image from the CelebA dataset and it takes both images and tries to identify the real and the fake image. Here, the model uses a total of four layers. As an input layer we have taken the generated data and the real image data and there are also two hidden dense layers along with an output layer which produce binary values. Other than the output layer we have used the Leaky ReLU activation function in our model and as our output will be classification type, so in the output layer we have used the sigmoid function. For the loss function in our model we have used the binary cross-entropy function.

B. Overall Workflow

In Fig. 1 the diagram we have shown the overall workflow of our proposed model, and later in this paper we have also explained how our generator, discriminator and the fully connected sequential model work [12] [13].

C. Generator

We have proposed two sequential models, one as a generator and another one as a discriminator to implement the generative adversarial model. We have used random noise as input data in the generator model. The input data is actually a layer which has 100 nodes. Then we have used four hidden layers where we have used Leaky Relu as our activation function. The hidden layers have 128, 256, 512, 1024 nodes respectively. We have used the Tanh activation function in our last layer. We have also used Batch Normalization to normalize the data. At last, we have reshaped the nodes into (48, 48, 1) size. The Adam optimizer has been used as our optimizer. During the compilation of the model, binary cross-entropy has been used as our loss function. The generator will produce some fake human faces and it will improve itself by the feedback from the discriminator. The flow diagram of the generator is presented at Fig. 2.

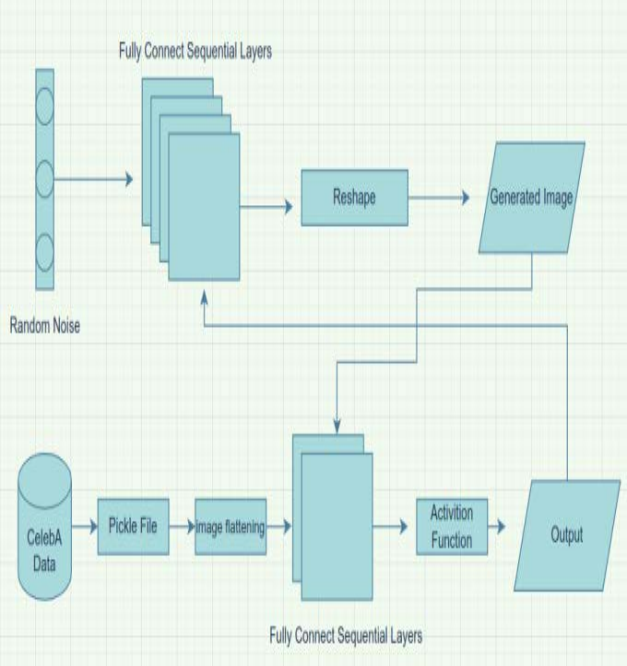

Fig. 1. Overall Workflow Diagram.

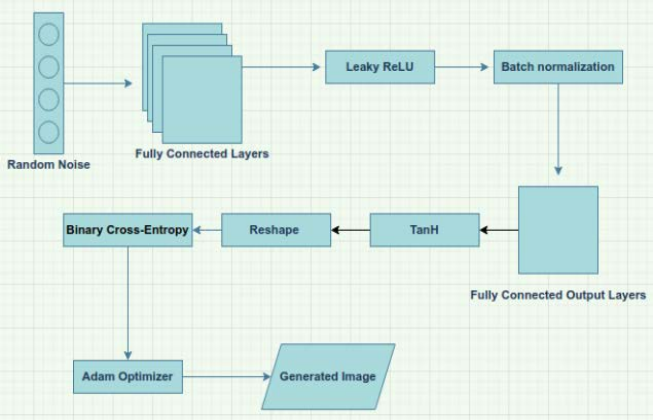

Fig. 2. Generator.

D. Discriminator

Discriminator is our second sequential model which we have used in our generative adversarial model. At first we used the flatten operation to make the data one dimensional. After that there are two dense layers. The first hidden layer has 48*48 nodes and the second hidden layer has half of the first hidden layer, which means 1152 nodes. Leaky ReLU has been used as the activation function in those two dense layers having the learning rate as 0.2. In the output layer the sigmoid activation function has been used as the activation function. We have used binary cross-entropy as the loss function and Adam as the optimizer during the compilation period. The discriminator will receive images from both the actual data from the celebA dataset and fake data from the generator and it will give feedback to the generator if the images generated by the generator are fake or real. The generator will receive feedback from the discriminator and improve itself. The flow diagram of the discriminator is presented in Fig. 3.

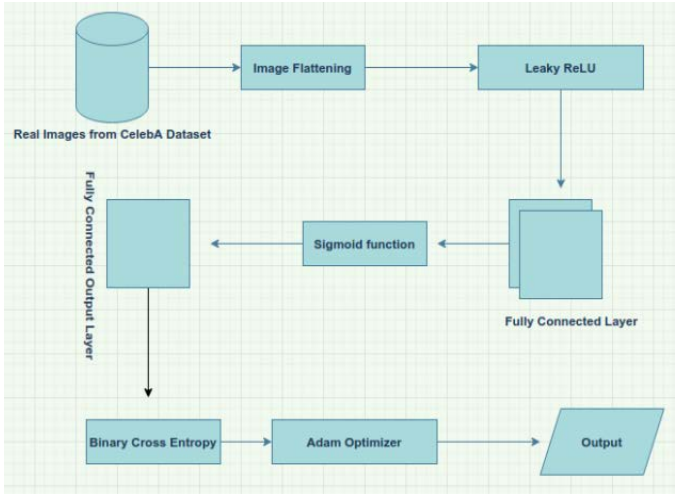

Fig. 3. Discriminator.

In our proposed model we have used two fully connected sequential models, one as a generator and another one as a discriminator. The generator receives a random noise and tries to train it as a human face and then our discriminator receives data from the CelebA dataset and tries to match the faces with the generator's fake faces, and give feedback to the generator, and based on the feedback our generator learns and tries to create comparatively better fake human faces after each training.

IV. Experimental Results

In this section, we have presented a detailed overview of the experiments of our presented model based on the CelebA dataset. We have shown how our model worked during different scenarios of the dataset. We have changed the size of the data multiple times to see how the model works each time. And we have got different results each time. We have described how the results differ and improve over the time.

A. Dataset and Experimental Settings

We have used the CelebA dataset for our model. In our CelebA dataset there are more than 200k data. At first, we ran our model on this whole dataset but we used the size of images as 48*48. We ran 100000 epochs on the model to get a good result. Normally in generative adversarial models we have to run a lot of epochs to get a good result. Generative adversarial models generally generate new images according to the task given. So, it requires a lot of time to generate the new images. That's why we ran the model 100k epochs [14].

B. Experimental Tools and Environment

We have used different tools and environments for preprocessing the data and running the model so that we can have good results from our model.

C. Programming Language

Tensorflow and Keras, which are the two libraries of the Python programming language, have been used in our model. Nowadays Python is the most popular programming language for implementing Machine Learning and Deep Learning models as it has many collections of packages

which are very helpful to implement the different Machine Learning and Deep Learning models. Python is also the most preferred language because it is very easy to learn and implement. Python has many free and open source libraries for Machine Learning and Deep Learning models. Tensorflow is one of them. Keras is an open source software library which supports a Python interface for artificial neural networks. Keras plays the role as an interface for the Tensorflow library.

D. Result

The four images shown in the Fig. 4 were generated by the same model after 52000, 65000, 81000 and 97000 epochs respectively. In the first image two faces could not be generated properly. Rest of the faces of the first image were generated but still there were a lot of noises which actually prevented us from understanding the faces properly.

Compared to the first one, the second image is slightly a little bit better but there are also one or two faces which could not be generated properly and contained a lot of noise. Almost all of the images were generated properly in the third image containing some noise, though it was comparatively better than the first two images. In the fourth image, this image performed quite well. In the fourth image all of the faces were generated and had a small noise. After receiving the images from the CelebA database we first flatten the images then we run the sequential model with the Leaky ReLU. Then in the output layer we use the sigmoid function to return the true or false type result value. Then in the discriminator we use the binary cross entropy as the loss function along with the Adam optimizer. This output from the discriminator then goes to the generator as a feedback and based on the feedback of the discriminator, the generator tries to create new images. So, after the above discussion we can say that there are still noises in the generated faces at the end of the model too, though the results of the latter part of the model performed comparatively better.

The four images shown in Fig. 5 were generated after 52000, 63500, 81500 and 99500 epochs respectively. After the 52000 epochs, two faces could not be generated properly, which we can see from the first image above. Some faces are not perfect though some faces are generated properly with little noise. In the second image which was generated after 63500 epochs, all of the faces were almost properly generated with some noise. After 81500 epochs and 99500 epochs, the faces which were generated, that we can see from the above third and fourth images, are much better. The interesting fact is that if we observe the faces properly we can see that almost all of the faces are different from one another.

After the above discussion what we can observe is that the faces became better as the time went on. And at the last phase of the model the faces became more and more clear and understandable. Another interesting fact is that as the time went on all of the faces were generated by which we can understand that the model was improving over time. And we can also see the faces were different from one another which is also a very positive output of this model.

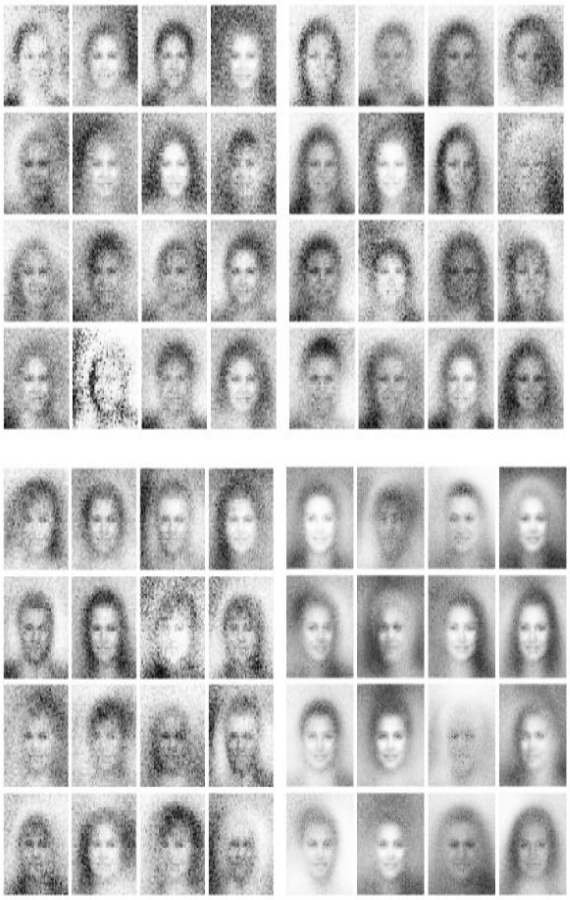

Fig. 4. Generated Data of 48*48 Size (CelebA Dataset).

Fig. 5. Generated Data of 100*100 Size (CelebA Dataset).

E. Loss Function

Here, in the Fig. 6, we can see the loss function of the discriminator. We ran our models at about 100000 epochs. If we see the graph we can see most of the loss value between 0 to 1, which is pretty good value for the discriminator. And as the discriminator is an un-trainable model we see the loss value maintains the same ranges between the whole 100000 epochs. Now we can see some spike in the loss value of our graph. If we try to separate the high spike by trying again and again to visualize the diagram without the high spike of the loss value in our (48 x 48) size images, we can then easily see our loss value is somewhere between 0 and 1.4. And if we calculate the high spike, we can see there are 771 high spikes between 100000 epochs which is around 0.77% high spike and other 99.23% are the average, means 0 to 1.4 ranges. This is pretty good value for our model.

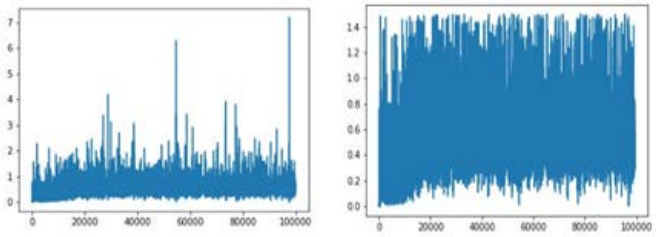

Fig. 6. Loss Values.

F. Comparative Analysis

Zeliha Dogan [9] et al. proposed a model titled "Baby Face Generation with Generative Neural Networks", where they tried to generate fake baby faces using comparatively low computational power. We have chosen this paper for comparison because both this paper and our work follow almost the same objectives. Both this paper and our model mainly tried to generate fake faces using comparatively low computation power. Below in this section we have tried to compare our work with this paper, and show where our proposed method stands out and why. We have also tried to show our drawbacks and possible ways to overcome this drawback.

Fig. 7. Sample Output Images of the Proposed Model.

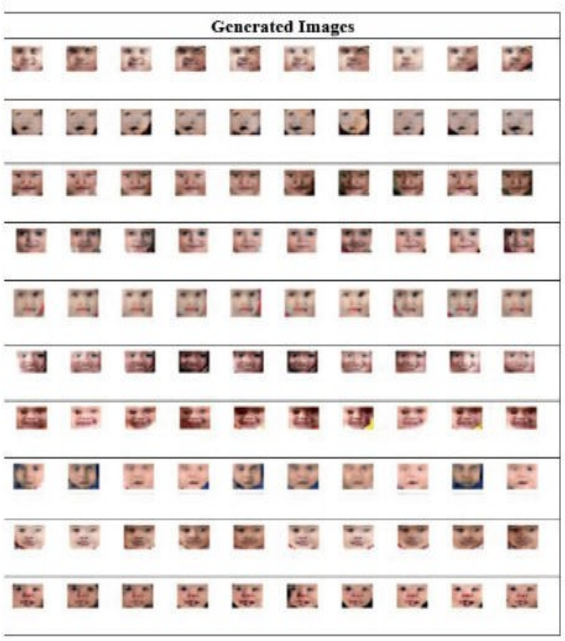

Fig. 8. Sample Output Images of the Baby Face Generation Model.

For sample output we have shown two figures. One is the Fig. 7 which is the generated output of our proposed model and another is the Fig. 8 which is the generated output of the baby face generation paper. At first glance we may think that both of these papers are very different. However, if we think clearly that both of those images are generated by a computer. And the objectives of those image creations are the same, which means creating fake human faces. If we see the baby face generation model images, they clearly lack the uniqueness of their faces. These images look like they were created with some other faces. It's more like editing rather than creating or generating fake faces. On the other hand if we see our generated images we can clearly see that these images are not like those. And surely we can't figure out if these images are created by mixing some other faces. It's more like generating from the ground. Other than that we can clearly see that our image has more sharpness in eyes, nose and faces. But we have drawbacks in color and hair.

In the Table I, we can see that the different aspects of two models. In our model we have used black and white data with three different sizes of images 48*48, 100*100 and lastly 150*150. On the other hand, Baby face generation paper uses RGB images with 200*200 sizes. They ran only 200 to 300 epochs whereas our model ran about 100000 epochs. In our dataset we use 10k to 203k images on the other hand baby faces use only 623 images.

Now we can clearly see that though the sizes of images are smaller, we have used a lot of images with more epochs without the color of the image which can be trained in comparatively low computational power. Though we need more computer power than the baby face generator paper, using this extra computational power we can with proper scaling of the images create better images with sharper faces.

TABLE I. COMPARISON TABLE

Model	Image Type	Image Size	Number of Epochs	Da-taset	Size of Dataset used	Generator Loss	Output
Our Model	Black & white	48*48, 100*100, 150*150	100000	CelebA	230K, 50K, 10K	0.0-10.0	Overall better quality image, sharp Face
Baby Face Generator Model	RGB	200*200	200-300	UTKFace	623	1.0-2.0	Low quality image, RGB image

So after comparing with the baby face generation paper, we can say that in our paper using some extra computation power with proper image scaling we generated much better human faces which are more realistic and sharper. But we also have some drawbacks and our main drawback is that we have used comparatively more computational power than the baby face generation paper and our resulting images are black and white and our model needs comparatively more training data.

G. Discussion

In this paper, we try to create fake human faces. Our main focus was to create as much as a good image with low computational powers without any external image classifier. As we discussed above we used two sequential layers and a CelebA dataset to create fake human faces.

From our result and the loss function, we can see that, we successfully created human faces from random noises using limited computational power. However, the images are not as good as the real life images, so there is lots of scope in our paper to improve. However, we can see from the result that we have successfully created fake human faces from the random noise data. This is one of our main focuses.

Overall, using our low computational power we have tried to generate real life fake human faces. For comparing this model's results and to see how it reacts with different sizes of images we have tested it with three different sizes. At first, we tested it using 48 * 48 sizes. At that time, the model did not give that much good result though it generated fake human faces having noises. And then, to improve the quality we used 100*100 sizes. This time, the model performed quite well, where it gave quite a good result. When we used 100*100 sizes of the generated images were very clear and understandable. To get a better result, we then used 150*150 images. This time the model gave comparatively better results than the previous two models. Each time we ran our model at 100000 epochs.

V. CONCLUSION

In this work, a generative adversarial network model is proposed to generate fake human faces with low computational cost. The proposed generative adversarial network model generates comparatively better real life fake human faces with respect to the present state-of-the-art. We have also shown how we can only use two fully connected sequential models, one as a generator and another one as a discriminator to generate fake human faces. Though there are a lot of improvement areas in our proposed method, we can say that using our proposed model we have successfully produced pretty good fake human faces comparatively using very low computation power. In future, we can develop model to generate the images that can be used by police to track down people who are involved in adultery. We can also develop models for generating fake computer generated images in visual art, for the advertising industries and computer games.

REFERENCES

[1] Z. Zhang, X. Pan, X., S. Jiang, & P. Zhao, "High-quality face image generation based on generative adversarial networks," Journal of Visual Communication and Image Representation, Vol. 71, 2020.

[2] X. Wang, & A. Gupta, "Generative image modeling us- ing style and structure adversarial networks," In the European conference on computer vision, pp. 318-335, Springer, Cham, 2016.

[3] F. S.Ghatas, & , E. E. Hemayed, "Gankin: generating kin faces using disentangled gan," SN Applied Sciences, Vol. 2, No. 2, pp. 1-10, 2020.

[4] A.Hamdi, & B.Ghanem, " IAN: Combining Generative Ad- versarial Networks for Imaginative Face Generation," arXiv preprint arXiv:1904.07916, 2019.

[5] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, & J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," Information Fusion, Vol. 64, pp. 131-148, 2020.

[6] T. Karras, S. Laine, & T. Aila, "A style-based generator architecture for generative adversarial networks," In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4401-4410, 2019.

[7] J. Zhao, X. Xie, L. Wang, M. Cao, & M. Zhang, "Generating photographic faces from the sketch guided by attribute using GAN," IEEE Access, Vol. 7, pp. 23844-23851, 2019.

[8] S. N. Esfahani, & S. Latifi, "Image generation with gans- based techniques: a survey," AIRCC's International Journal of Computer Science and Information Technology, Vol. 11, No. 5, pp. 33-50, 2019.

[9] G. Ortaç, Z. Doğan, Z. Orman, & R. ŞAMLI, " Baby Face Generation with Generative Adversarial Neural Networks: A Case Study," Acta Infologica, Vol. 4, No.1, pp. 1-9, 2020.

[10] H. Huang, P. S. Yu, & C. Wang, "An introduction to image syn- thesis with generative adversarial nets," arXiv preprint arXiv:1803.04469, 2018.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, ... & Y. Bengio, "Generative adversarial nets," Advances in neural information processing systems, Vol. 27, 2014.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, ... & Y. Bengio, "Generative adversarial networks," Communications of the ACM, Vol. 63, No. 11, pp. 139-144, 2020.

[13] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, & A. A. Bharath, "Generative adversarial networks: An overview," IEEE Signal Processing Magazine, Vol. 35, No. 1, pp. 53-65, 2018.

[14] X. Liu, B. V. Kumar, Y. Ge, C. Yang, J. You, & P. Jia, "Normalised face image generation with perceptron generative adversarial networks," In 2018 IEEE 4th International Conference on Identity, Security, and Behaviour Analysis (ISBA), pp. 1-8, IEEE, 2018.