Art Creation Project Proposal

Ho Ko*[1] and Meng Yuan†[1]

[1]ECE, UWaterloo

August 11, 2021

Abstract

In unsupervised learning tasks, generative models have become a major research hot spot, of which GAN (Generative Adversarial Network) is the one of most representative, which has been widely used in image processing and computer vision, natural language processing, music, speech and audio, medicine, and Data science. In our project, we mainly focus on the application of GAN in image generation. As we know, GAN has made remarkable achievements in generating photographic images such as human faces, animals, plants, etc. But what attracts us more are the applications in the field of artistic creation, such as the generation of hand-drawn drawings and a variety of anime characters. In recent years, GANs are committed to generating high-quality anime avatars with the premise of saving computing power, which are initially inspired by an animation character generation project called MakeGirlsMoe. We will learn some algorithms based on GAN proposed in the past three years, such as Multi-Scale Gradients-GAN and StyleGAN. The emergence of StyleGAN is an important milestone in image generation technology based on style transformation, we use it as our baseline, and aiming to solve some existing concept-wise problems of the StyleGAN model, which we proposed 4 variants to the mapping model.

1 Introduction

In the first section, we make a brief introduction of generative models and three representative models among them, Auto-Encoder, Variational Auto-Encoder and GAN respectively, including differences and connections between them, as well as advantages of each of them. In addition, we review the Multi-Scale Gradients-GAN, which has been proposed in recent years.Next, we learned the StyleGAN model in depth and dissected its internal structure and the main functions of each component to prepare for our next experiments. In the third section, we introduce two datasets consisting of anime faces and abstract art images respectively. The former one is from Kaggle, and the later is collected by ourselves, only a preliminary screening has been carried out, so there may be some noises, which may have a certain impact on the results. In the next part, we use the StyleGAN model as our baseline, and based on our current different datasets, we make some modifications, mainly focusing on the mapping model, to do some experiments with 4 variants. Results and conclusions of our experiments are shown in the last two sections.

2 Related Research

In this section, we will briefly cover the related researches and concepts for image generation.

2.1 Autoencoder

AE (autoencoder) [1] is a type of neural network model that aims to find an encoding scheme for the target data automatically. Different layers within the model represent different depths of abstract features being extracted from the original data. A typical AE model usually have a layer in the middle with the lowest number of neurons, which the features extracted in that layer will be equivalent to the compressed representation of the input data.

*h25ko@uwaterloo.ca

†m45yuan@uwaterloo.ca

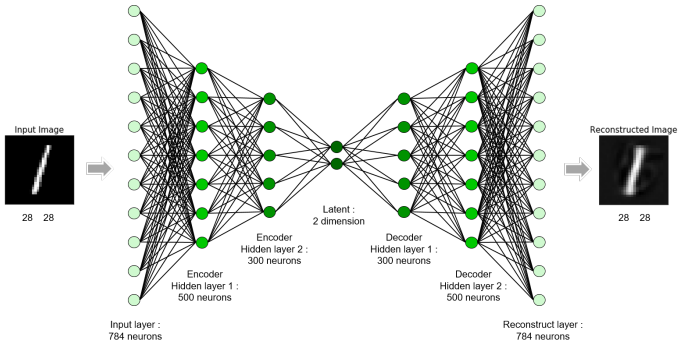

Figure 1: Autoencoder example[2]

This type of model can be achieved by setting the ideal output of the model to the input data itself. When the performance of a trained model is good enough, the output will be identical to the corresponding input, which also means the output calculated by a partial of the model can reconstruct into the original data when feeding through the remaining part of the trained model.

An AE model mainly consists two parts, an encoder and decoder. The encoder is responsible for the compression process of the model, usually via reducing the dimension of the input data layers by layers until it reached the narrowest part of the model. Output value of that layer will then be the encoded input data, representing the loaded data with a lower dimension.

The decoder on the other hand is in charge of reconstructing the data from its encoded form. It generates details based on the limited number of extracted critical features or reference information, which opens an opportunity to be utilized as a generator for convincing data outside the training data set with similar type.

2.2 Variational Autoencoder

A VAE (Variational Autoencoder)[3] is an variation of AE, in particular it is designed to generalized the generation process. Both model generates new data by inputting a new and reasonable point in the encoded space to the decoder, but VAE improve the AE model especially for generation purposes.

The goal of a standard AE is to automatically extract the critical features that represents the data, and decode it back to its original form with little lose, the optimized encoding space will usually end up separating different input data as distance as possible for the decoder to reconstruct accordingly. This results in a distinct distribution of the data in the encoded space which the space between different input data are far apart. This limited the capability of generating new unseen data, as the encoded space are usually sparse with a lot of large gaps between data points, resulting in the points in that space are not corresponding to a reasonable and appropriate data.

VAE solves this problem by using means, μ and standard deviations, σ to represent the data in a probabilistic manner instead of encoding it into precise values. This allows the space in between different data be more interpretable and generated data from random points in the encoded space will have a much higher probability of making sense and being reasonable.

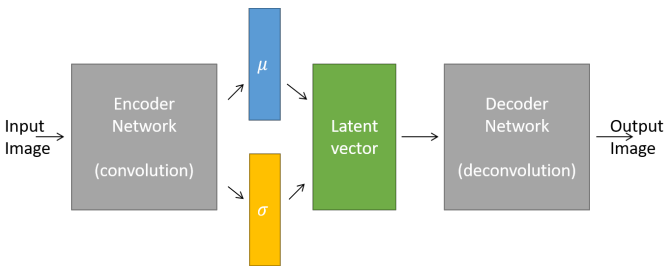

Figure 2: VAE model for image data

2.3 Generative Adversarial Network

GAN (Generative Adversarial Network)[4] is a model designed mainly for generating data. Its approach is totally different to the traditional AE methods, which it utilizes the confrontation between the generator and the discriminator instead of directly look for the hidden representing features via training the model to extract and reconstruct the data.

The generator is trained based on the discriminator, aiming to trick the discriminator into thinking the data it produced is the real data. Theoretically if the discriminator is well-trained, and since the functions that constructed the discriminator are all derivable, the generator will be able to find a better solution to confuse the discriminator, result in capable of generating more realistic data.

The same concept is applied to the discriminator, as its goal is to classify the generated data from the real data. After the discriminator is trained, it becomes more proficient in distinguishing the difference, and giving the generator a new goal to achieve, a better classifier to trick.

By repeating these adversarial processes, both the generator and discriminator improves every round, and a satisfying result will be reached when the accuracy of discriminator is close to 50%, meaning the ML classifying model can no longer tell the difference between the real and generated data.

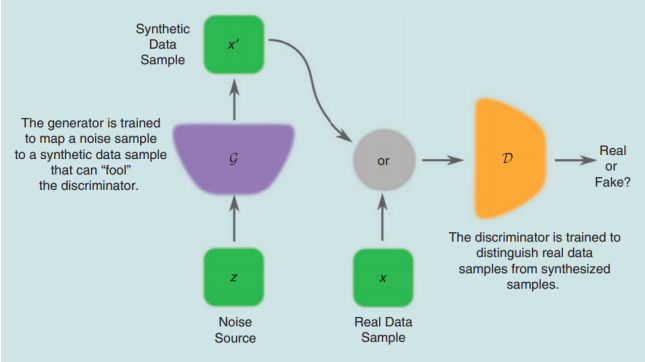

Figure 3: An overview of the GAN model, the Generator(G) and Disciminator (D) are trained back and forth [5]

2.4 Multi-Scale Gradients-GAN

The MSG-GAN (Multi-Scale Gradients-GAN) [6] is a GAN model improved from the ProGAN (Progressively Growing GAN,) [7] variation. The ProGAN is suggesting that it will be far too tough for a model to aim generating a realistic high resolution image directly from the beginning. So instead, a model should first target to generate the lowest resolution first, then progressively increase the up-scaling layers and aim for a higher resolution step by step. Each time the ProGAN model increases the resolution, the concept of residual block were used to skip the higher resolution part and allowing the result of previous low resolution from the generator being able to be utilized by the discriminator similar to before the transition of increasing resolution.

The MSG-GAN improved the ProGAN model by directly feeding the previous low resolution result from the generator to the corresponding layer in the discriminator. The concept of this improvement is similar to improving from Res-net[8] to UNet[9], not only lowered the training time but also preserved most of the critical information with little effort.

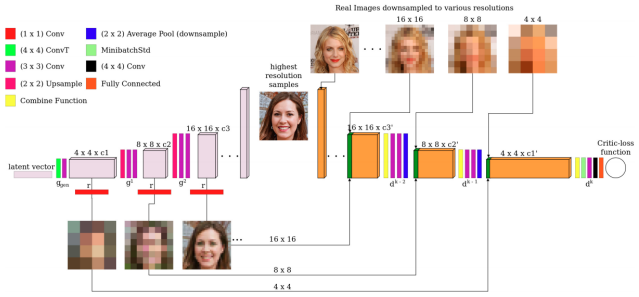

Figure 4: An overview of MSG-GAN model[6]

3 StyleGAN – Our Baseline

StyleGAN[10] was introduced by Nvidia in 2018, revolutionizing the approach of image generation. StyleGAN consists of mapping network and synthesis network. The former one is composed of eight fully-connected layers, which map the input latent $z \in Z$ into an intermediate latent space W through a series of affine transformations. And $w \in W$ is then fed into multiple convolution layer of synthesis network through AdaIN(Adaptive Instance normalization) to control the style of generated image in different levels of resolutions. Besides, after each convolution layer, noise is added, which is used to add variations of details to the generated image. This model architecture allowed the output image to be adjusted in different levels of details and able to guarantee outputting the same image with different variation. The mechanism of style adding also add the possibility for image style combinations to generate image with specific features and properties, which is what the traditional GAN is totally incapable of, while it only receives the latent code at the beginning. (Fig. 6.)

Since we use the StyleGAN as our baseline, we will go deep to its internal architecture as well as functions of each component in the model.

3.1 Mapping network

Previous to learning the mapping network, let us firstly understand the latent space. The latent space is a representation of compressed data, saving the most important features only, where distances between similar data points are closer. Thus, the latent space is important to help us learn key features of data, further finding a simpler way to representing data. Similarly, we need to map the highly coupled and interrelated features in Z to the intermediate latent space W to obtain representative hidden features after decomposition. The role of the Mapping Network is to complete the decoupling of the input latent Z and it consists of 8 Fully-connected layers.

3.2 Style Mixing

The style mixing can be regarded as a highlight in the StyleGAN, which is mainly finished in the synthesis network, where there are 18 layers, evenly assigned to 9 different resolutions from 4×4 to 1024×1024, and there is an AdaIN[11] (Adaptive Instance Normalization) unit between each pair of them. The AdaIN firstly normalize all channels of output of the last convolution layer to adapt the mean and variance of the style produced by the latent W, so that the feature of this style can be visualized. The original intention of Style mixing is to find the location of the latent code that controls the different styles. The specific method is to enter two different latent codes z_1 and z_2 into the mapping network to obtain w_1 and w_2 respectively, which represent two different styles respectively, and then randomly select a middle intersection in the synthesis network. The part before the intersection uses w_1, and the part after the intersection uses w_2. The generated image should have both features controlled by w_1 and w_2. which can be called as style mixing.

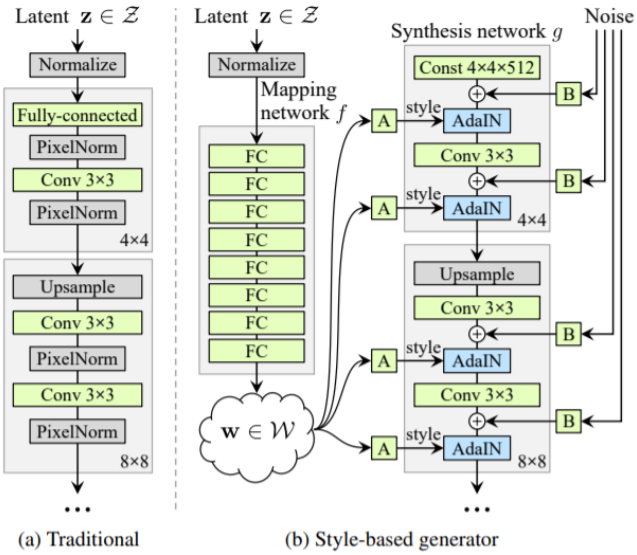

Figure 5: Comparison of traditional generator and StyleGAN generator [10]

4 Proposed models

The architecture of the current project is reference based upon the idea of the StyleGAN model. We believe that by applying latent variable layer by layer to the generative model is a much better way not only to control the properties of the generated images, but also an overall easier convergence structure for art creational purposes comparing to the traditional up-sampling the random noise approach.

But StyleGAN did not actually fully utilize the idea of applying dedicated styles to different resolution layers. In the original paper[10], the model first transform the latent variable to an intermediate space W thorugh 8 fully connect layers, then use the same variable to apply throughout the whole model(different resolution layers). But we think styles in different resolution should not always be the same, especially in art creational purposes, as artists usually comprehending different color theme in different layers, and combined together for a more vivid work.

Also, it has a huge downside of remapping the style when higher resolution modules are being added to the model since all levels are using the same matrix $A \in W$ to control the styles. The 8 layers of hard coded structure for latent space transformation is also doubtful as the magic number is just set without the adaptability to apply to other cases and the total of 8 fully connected dense layers also produces the vanishing gradient problem, making the mapping structure overly complicated without reasonable usage.

Here we proposed 4 variations of the mapping models for solutions to these problems.

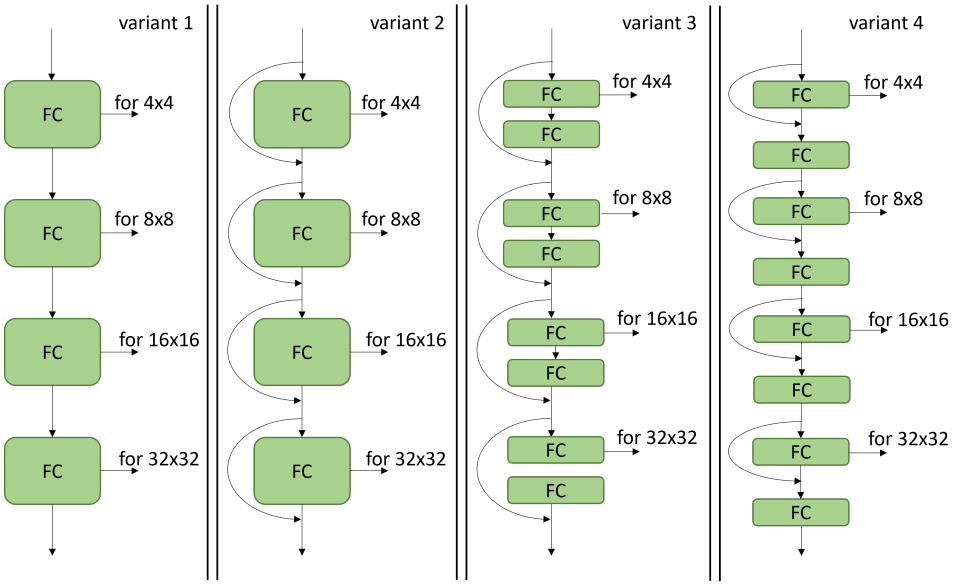

Figure 6: Illustrations of 4 variations of the mapping models

The first variant is by directly controlling the style of different resolution levels by different layers of the mapping model. Via this structure, the vanishing gradient and different control matrix for different resolution details are solved. But all the styles for higher resolution are only considering the style from the previous layer directly, this limits the usage for images with highly patterned structure where details of different scales are correlated. For example, images in the abstract art dataset we collect.

The second variant is by applying resnet-links throughout different layers. This will allow the higher resolution styles be not only considering the lower style output, having the capability to generate much more variation of images. But looking closly to this solution, we noticed that only linear combination of the styles from the previous resolution are being considered(leakyRuLU is applied, so theoretically it is not linear, but not nonlinear enough), which is reasonable to believe it is highly not complicated enough. So we add another layer right before concatenating the styles to the next layer for much more ability to learn complex transformation.

The only problem left is that the width of the mapping model will increase linearly for higher resolution, resulting in a quadratic growth of parameters to train. To solve this problem, the final variant is purposed. By moving the dense layer introduced in the third variant to right after the concatenation layer, the model is able to maintain a linear growth number of parameter along the increase of resolution. The vanishing gradient problem will appear again when aiming to produce

higher resolution images due to the lack of direct concatenation of the previous layers, but we believe it is appropriate since we do not want the mapping model to adjust earlier layers intensely while training for higher resolution layers.

5 Dataset

Actually, we collect three datasets, anime faces[12], abstract art as well as landscape images, but due to the limitations of time and computation resources, we focused on the first two for experiments.

The anime faces dataset is found on Kaggle, and it consists of 21551 anime faces scraped from the website "www.getchu.com", which is a comparably clean dataset.

The other one is collected by ourselves, we firstly scrap images of the abstract style from Wikiart by using the python script file[13], and then filter out noise pictures, producing our abstract art dataset eventually, which is a small dataset consisting only 306 images.

6 Results

The model used for experiments in this project is mainly applying the forth mapping variant model with the original StyleGAN generator and discriminator structure. We applied it through two different kinds of art datasets, anime portraits and abstract art for comparison between our mapping variant against the original version.

All the following experiments are conducted with 10000 epochs for each resolution layers when training it as MSG-GAN manner.

6.1 Anime Portraits

	Parameter	Generator loss	Discriminator loss
FC(512) x 8	2097152	12.6688	16.3742
FC(256) x 8	524288	9.2945	12.2617
FC(256) x 4	262144	11.5217	15.5377
Variant_v3(128)x5	655360	9.6903	13.5434
Variant_v4(128)x5	409600	9.5073	12.1308
Variant_v4(128)x6	491520	7.7205	9.2092

Figure 7: Wasserstein loss comparison between different mapping models for anime portraits

According to our experiments(Fig. 7), the fully connected structure for style mapping transformation is can be fine-tuned to beat the performance of the third mapping variant, but the fine tuned version of the forth mapping variant performance much better in terms of number of parameters being less than the FC(256)x8 case, the generator and discriminator loss is also decreased around 25% of the best case by the StyleGAN mapping structure, not to mention the original FC(512)x8 is far behind, the loss is around twice as much as our best settings.

As you can see in the example results(Fig. 8), the original fully connected approach(FC(512)x8) tends to have more blobbing effect, different area of the image being connected together resulting in an obvious artifact noise. This problem is actually being mentioned in the paper of StyleGAN2[14], and Nvidia's solution is to adjust and change the AdaIN's concatenate method. To our understanding, we think it might be happening due to the same style applying through out the whole model, resulting in similar color theme being combined in all resolution layer and generates these blobbing effects. That is why our approach also did manage to remove this problem but from another perspective by redesigning the mapping structure for more detailed control-ability for the model.

Figure 8: Results on anime faces dataset

6.2 Abstract Art

We only picked the best model result from both mapping structures for comparison, and once again not only the number of parameters of our approach is less than half the original one, the loss of generator and discriminator are also around 20% better.

	Parameter	Generator loss	Discriminator loss
FC(512) x 4	1048576	218.2272	251.8703
Variant_v4(128)x6	491520	177.3948	208.9415

Figure 9: Wasserstein loss comparison between different mapping models for abstract art

We can see that there are clearly some obvious artifact in the FC(512)x4 approach(Fig. 10). This original approach failed to learn the weird color themes combining in different levels of details, resulting in always mixing all the colors together, where as our approach (the forth variant(128)x6) seems to master in this particular task.

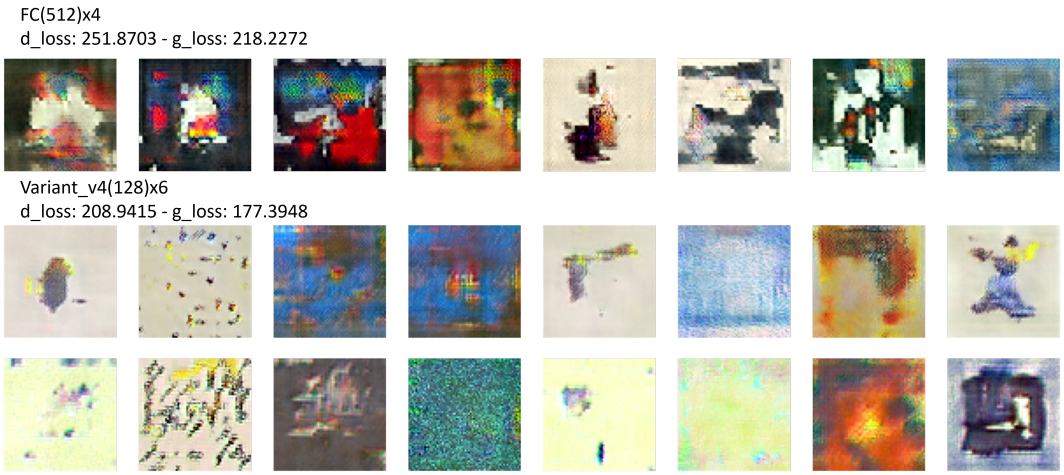

Figure 10: Results on abstract art dataset

7 Conclusion

Image generation has always been a difficult task and the development of model designs are significantly improving in recent years. Most of the application are for real-world image generation, which is undoubtedly useful, but we believe by generating art and realistic cartoon characters can not only help generate ideas for artists, but also with a probability of being combined and become part of the art industry in the future.

The StyleGAN model based on style transformation technology once attracted the attention of the public. Based on our understandings of it and our current two datasets, we try 4 different mapping variants, primarily focused on the best, last variation.

Our results show that our mapping structure not only provides the flexibility to adjust the vector size to control style in different resolution layers, extendable for different resolution goals, the total number of parameters is way lower and the performance improves significantly.

This allows the model to mimic the art creation process of typical artist by having different approaches in different layers, allowing this model to be more possible for generating more realistic art.

Due to large amount of time and computing resource required by this project, extensive work for example to generate a full realistic cartoon or high-quality image will be not possible, we generate images of relatively low resolution, 64×64 instead. But we believe if the easier target can be reached, such as generating convincing character images or satisfying abstract art, the more sophisticated goal will be reachable in the future.

References

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations by Error Propagation*, p. 318–362, MIT Press, Cambridge, MA, USA, 1986.

[2] Gabriel Achour, Woong Je Sung, Olivia J Pinon-Fischer, and Dimitri N Mavris, "Development of a conditional generative adversarial network for airfoil shape optimization," in *AIAA Scitech 2020 Forum*, 2020, p. 2261.

[3] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[4] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[5] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath, "Generative adversarial networks: An overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, 2018.

[6] Animesh Karnewar and Oliver Wang, "Msg-gan: multi-scale gradient gan for stable image synthesis," *arXiv preprint arXiv:1903.06048*, 2019.

[7] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[8] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, vol. 31.

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[10] Tero Karras, Samuli Laine, and Timo Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[11] Xun Huang and Serge Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1501–1510.

[12] "Anime dataset," https://www.kaggle.com/soumikrakshit/anime-faces.

[13] "Wikiart scraper," https://github.com/robbiebarrat/art-DCGAN/blob/master/genre-scraper.py.

[14] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8110–8119.