Face Image Generation Through GANs

Tianda Fu
Department of Electrical
and Computer Engineering
Queen's University
21tf5@queensu.ca

Juntao Lin
Department of Electrical
and Computer Engineering
Queen's University
21jl80@queensu.ca

***Abstract*—Generative Adversarial Networks (GANs) are deep neural network structures that contain two networks, pitting one network against the other, hence the name "adversarial". In 2014, GANs were proposed by Goodfellow and other researchers [1] at the University of Montreal, including Yoshua Bengio. The potential of GANs is huge because they can learn to mimic any data distribution, so GANs can be taught to create worlds similar to ours in any domain, such as images, music, speech, and prose. In a sense, they are robotic artists whose output is impressive, even capable of moving people deeply. In this project, we would use several types of unconditional GANs (including DCGAN [2] build from scratch and pretrained StyleGAN2 [3] and StyleGAN3 [4] models) and then test and compare their performance on FFHQ Dataset [5] and Anime Face Dataset [6].**

I. Introduction

Image acquisition systems have been largely deployed in recent years, due to the reducing cost of manufacturing electronics as well as the increasing demand for collecting customer data. Stronger computation power, machine learning architectures with higher accuracy, and the economic advancement in computer vision systems result in the expansion of the global computer vision market. In 2020, the global computer vision market generated 9.45 billion. Followed by a prediction of $41.1 billion by 2030 which implies a compound annual growth rate of 16% [1]. As the computer vision market expands, a large quantity of images is now available, and the demand for synthesizing and processing images increases. Distinct methods have been investigated to perform video and image editing, such as texture synthesis [2], image inpainting [3], and image stylization [4]. Most of the proposed methods are based on pixels, patches, and low-level features [5] that make high-quality image synthesis, such as photorealistic image stylization, remains a challenging task.

Another popular application for GANs would be applications related to medical imaging. The healthcare industry produces a considerable amount of data every day that can benefit the development of different machine learning algorithms. One of the biggest problems with adapting machine learning in medical imaging is the lack of labelled data [6]. The labelling process can be very time-consuming, and it is heavily dependent on the availability of doctors and other medical professionals. Collecting and constructing a well-labelled dataset for clinical research is a complex, labour-intensive task that might be subject to unexpected issues and biases [7].

The training data decides the performance of the supervised machine learning models. These models often require a large amount of labelled data to achieve high accuracy on unseen samples. Data augmentation is commonly used in machine learning and deep learning methods while the number of training samples is limited. Traditional data augmentation methods such as flipping, rotation, and scaling can mislead the generator to learn the distribution of the augmented data [8]. Generative Adversarial Networks (GANs) generate synthetic samples based on a given dataset. In medical imaging, GANs can generate samples that follow the underlying distribution of the original training data [8, 9].

In computer vision, learning and understanding geometric information from source images is a crucial task. The fast advancement of GANs in recent years [10, 11, 12] promotes the development of image synthesis. Through adversarial training, GANs learn the mapping from a latent distribution to the actual training data. Face image synthesis through GANs combines portraits of different people to generate new photorealistic images that exhibit unique identity, expression, and pose [13]. Lately researches on GANs show that in the image synthesizing process, many latent semantics learned by GANs are interpretable [14, 15, 16]. Different units in the generator can represent a specific visual expression or object.

II. Related Work

A. Augmenting Training Data

Bowles et al. [9] proposed to increase the training data size through GANs, therefore improving the performance of a segmentation CNN. Since GANs have an unstable training process with high-resolution images [17], a Progressive Growing of GANs (PGGAN) network was chosen for synthesizing training data. To train the PGGAN, 80k patches were sampled from the available dataset. The impact of adding augmented data to the training set was validated through a Computed Tomography (CT) and Cerebrospinal Fluid (CSF) datasets. Experiment results show a 1-5% improvement in Dice Similarity Coefficient (DSC). Data Augmentation (DA) is recommended as part of the preprocessing steps when the available dataset is relatively small. Adding additional 10-100% augmented data could improve the DSC substantially when the dataset contains $5-50$ labelled images for each class.

To improve the computation efficiency, PNVR et al. [18] proposed a SharinGAN architecture that reduces the domain

gap between real and synthetic images. The SharinGAN model simplifies the process of combining real and generated images. Rather than training GANs to learn the mapping from the latent space to the real images, SharinGAN maps both the real and synthesized image to a shared domain. The primary networks used in the paper were trained to learn the shared domain information while ignoring irrelevant and domain-specific information (both real and synthesized). The primary networks are geometry-aware symmetric domain adaptation framework (GASDA) [19] and SfSNet [20]. These networks were evaluated respectively through the KITTI [21] and Photoface dataset [22]. Further experiments show that pairing with SharinGAN improves the performances of the primary networks.

B. Robotics

Machine learning algorithms can process highly non-linear dataset, but supervised learning requires a well-prepared dataset that requires time and resources to construct. Ren et al. [23] proposed to use GANs to learn the inverse kinematics and inverse dynamics of robot manipulator. Different GAN architectures were explored that include Conditional GANs (CGANs), Least Squares GANs (LSGANs), Bidirectional GANs (BiGANs), and Dual GANs (DualGANs). The performances of the GANs were compared with a multilayer perceptron (MLP) model as a baseline. The experimental setup includes a MICO manipulator (4-DOF) and a Fetch manipulator (8-DOF). The training data for GANs and the baseline model are trajectories generated randomly by these robot manipulators. When solving the inverse kinematics for the MICO manipulator, GANs have the best performance when the size of training data is limited. In other tasks, the baseline model has a better precision in solving the kinematics and dynamics.

C. Face Image generation

Animating different facial expressions naturally on static images is a challenging task in computer vision. Wile et al. [24] proposed a neural network model named X2Face that controls the facial expression and pose of the character in the source video. The network takes two videos as input, the source video, and the reference video. Each video contains only one unique character with different expressions. The output of the model is a synthetic video generated from the source and reference videos. X2Face preserves the facial identity of the character in source frames (modelling in 3D) and extracts the pose and expression from the reference frames. The model uses facial identity in the source video as a model to perform the expressions and poses captured from the reference video. The proposed model was trained in a self-supervised manner with sufficient video data. The network architecture contains an embedding network and a driving network. The embedding network takes the source frame as input and learns the mapping from the source frame to a face representation. The driving network takes the reference frame as input and transforms its pixels to produce a generated frame as the output of the X2Face model. Multi-modality data are included in the model architecture to improve the performance. The pose of the character's face (in source and generated frames) is controlled by a pose code that can vary the roll, pitch, and yaw angles. Audio data from VoxCeleb [25] dataset were used to guide the pose of the character's face by modifying the parameters in the driving network. Since the X2Face model learns the pose, expression, and facial identity of the input videos without making any assumption, the model achieves a decent generalization on unseen faces.

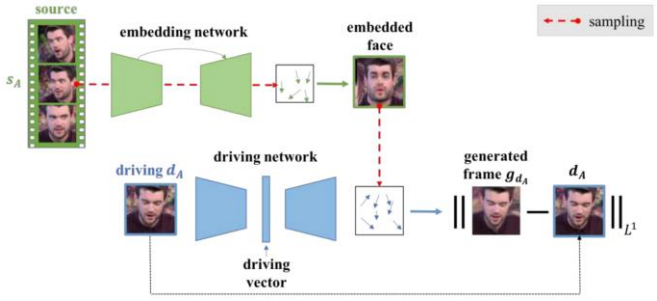


Fig. 1. The architecture of the X2Face model [24]

D. DCGANs

Radford et al. [26] proposed the Deep Convolutional Generative Adversarial Networks (DCGANs) that is a convolution model with architectural constraints on the generator and discriminator. Compared with the original GAN model [27], the following adjustments were made: removing fully connected (FC) layers; adding batch normalization layers; in the generator, replacing pooling layers with fractional-strided convolutions, using Tanh activation in the output layer and ReLU activation in the remaining layers; in the discriminator, replacing pooling layers with strided convolutions, using Leaky ReLU activation for all layers. The DCGANs provide a more stable training process while maintaining a high resolution for the generated images. Radford et al. determined filters learned by GANs are responsible for drawing specific objects. This discovery has been further supported by more recent research in GANs [28, 29, 30, 31].

III. METHODS

In this section, three methods are proposed to solve the face image generation task that are DCGAN, StyleGAN2, and StyleGAN3. The models and parameters used are the same as proposed in [26, 32, 33] with minor adjustments.

A. DCGAN

GAN mainly consists of two parts of the model - the generator and the discriminator.

The job of the generator is to generate "fake" data that closely resembles the training data, and the job of the discriminator is to determine whether a photo is a real training photo or a fake photo generated by the generator. While training, the generator tries to generate better and better fake photos to fool the discriminator, and the discriminator keeps getting better at spotting and correctly distinguishing between real and fake photos.

The balance of the game is when the generator produces a fake that looks like it came directly from the training data,

while the discriminator always guesses with 50% confidence that the generator output is real or a fake.

And here are some parameters:

1) x: The data representation of the image.

2) D(x): The possibility that the output of the discriminator x comes from the real training data (when x comes from the real training set, the value should be relatively large, when x is generated by the generator, the value should be relatively small) which can be seen as a traditional binary classification.

3) z: A latent space vector sampled from the standard normal distribution

4) G(z): Representing the function of the generator mapping from the latent vector z to the data space (the purpose of G is to estimate the distribution from which the training data comes, so that fake samples can be generated from the estimated distribution).

5) D(G(z)): The probability that the output of generator G is a real image.

And after knowing those parameters, the loss function of a generative adversarial network can be presented as:

$$min_G max_D V(G,D) = E_{X\sim Pdata(x)}[logD(x)] + E_{Z\sim PZ(z)}[\log(1 - D(G(z)))] \quad (1)$$

As basically all the experiments have shown that the convolutional neural network (CNN) can be the best model for image processing in deep learning. And the DCGAN is an attempt to combine CNN and GAN.

The principles of DCGAN and GAN are the same, except that D and G are replaced by two convolutional neural networks. But it is not a direct replacement, some changes are made to the convolutional neural network structure to improve the sample quality and convergence speed.

Here are the structures of the generator and the discriminator which were used in this project:

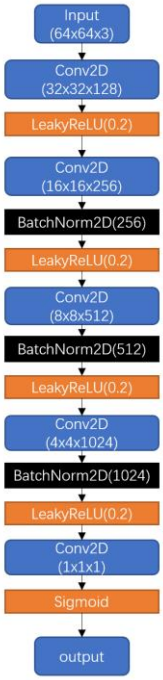


Fig. 2. The structure of the discriminator of DCGAN

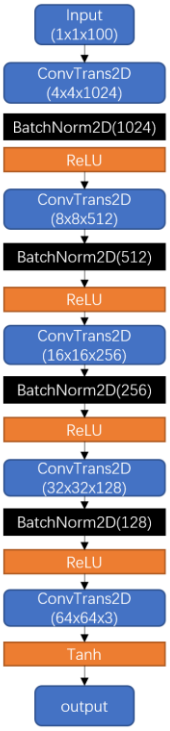


Fig. 3. The structure of the generator of DCGAN

From the original paper of DCGAN [26], the authors have recommended some values for the parameters in this network:

6) All models are trained with mini-batch stochastic gradient descent (SGD).

7) All weights are initialized according to a zero-centered normal distribution with a standard deviation of 0.02.

8) In LeakyReLU, the slopes are all set to 0.2.

9) The Adam optimizer was used and hyperparameters were tuned. The suggested learning rate of 0.001 in the base GAN is too high, so use 0.0002 instead.

10) Leaving the momentum β1 at the suggested value of 0.9 would causes training oscillations and instability, while lowering it to 0.5 helps stabilize training.

Besides those recommendation parameters, this model also needs some other parameters, and they were set as follows:

TABLE I. The list of some other parameters

name	**value**	**description**
num_epochs	100	number of epochs
epochs_D	4	epochs of Discriminator
epochs_G	3	epochs of Generator
batch_size	128	batch size for training
image_size	64	the scale of the input
nc	3	channel of the input
nz	100	the scale of latent vector

B. *StyleGAN2*

The StyleGAN [34] is developed based on the progressive GAN [35] that includes the following changes: removing the connections between latent vector and generator input, addition of mapping network and adaptive instance normalization (AdaIN) layers, additional noise to each convolution block. The StyleGAN is trained with the progressive training method that uses bilinear sampling layers instead of nearest neighbor layers for upsampling.

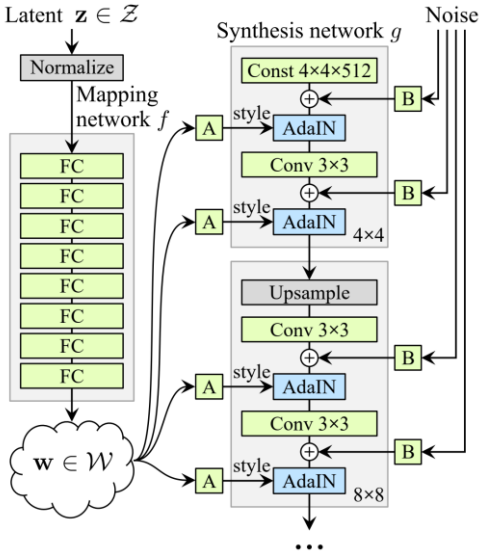


Fig. 4. StyleGAN generator architecture [34]

Compared with DCGAN, StyleGAN added a MLP between the latent space vector z and convolution layers. In the proposed DCGAN architecture, the latent space Z is constrained by the pre-defined gaussian distribution. GANs learn the underlying distribution of the real data through training. While gaussian distribution does not apply to the actual data, the DCGAN might have a poor performance. On the other hand, the intermediate latent space W is not limited by a single distribution. W can have any distribution mapped by the MLP network. Therefore, the GANs architecture with latent space W is more likely to learn the distribution of the real data. To obtain the style vector y = (y_s , y_b), the affine transformation block A will be applied to the intermediate latent vector **w**. The style vector is transformed and incorporated in each block of the generator network through the AdaIN operation described in equation 2. x_i represents the feature map generated by the convolution layers with noise B added.

$$\mathrm{AdaIN}(x_i, \mathrm{y}) = y_{s,i}\frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i} \qquad (2)$$

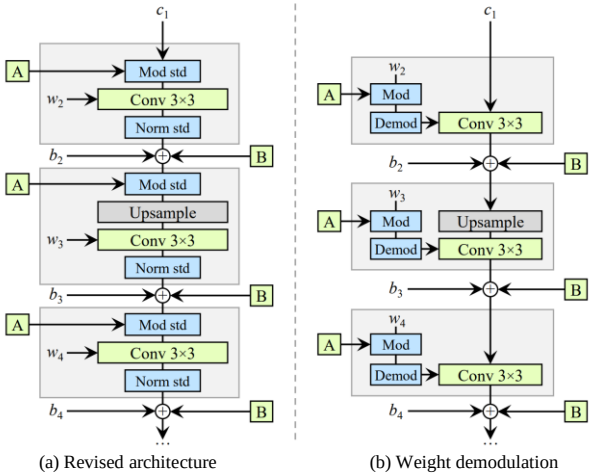
In StyleGAN2 [32] architecture, the AdaIN has been replace with modulation and normalization described respectively in equation 3 and 4. w_i and w_i' are the original and modulated weight with the scaling factor s_i for the corresponding feature map. The normalization is performed through the standard deviation σ_j.

$$w'_{ijk} = s_i w_{ijk} \qquad (3)$$

$$\sigma_j = \sqrt{\sum_{i,k} {w'_{ijk}}^2} \qquad (4)$$

The detailed architecture is shown in Fig. 5. Comparing with the StyleGAN architecture, the additional noise B has been shifted outside the style block. In the revised architecture, the weights are adjusted with the style and normalization, whereas the in the weight demodulation architecture, the instance normalization is replaced by the demodulation operation described in equation 5.

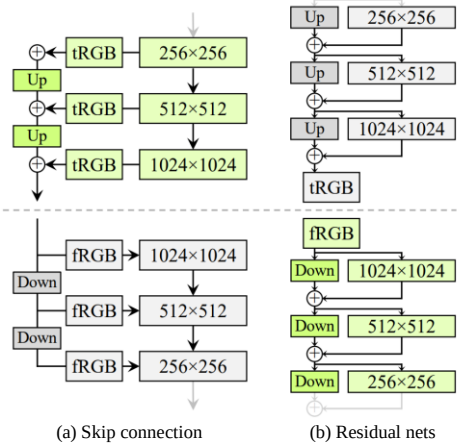
$$w''_{ijk} = w'_{ijk} \Big/ \sqrt{\sum_{i,k} {w'_{ijk}}^2 + \epsilon} \qquad (5)$$



(a) Revised architecture (b) Weight demodulation

Fig. 5. StyelGAN2 generator architecture with weight demodulation [32]

Progressive growing of the network provides high resolution images, but it also comes with the characteristic artifacts, occasionally the facial features would not move along when the person moves. To solve this issue, skip connections are added to the generator, whereas residual nets are integrated to the discriminator.



(a) Skip connection (b) Residual nets

Fig. 6. Networks above the dash line are generators and below are discriminators

TABLE II. Overall structure of FFHQ Dataset

Path	Size	Files	Format	Description
ffhq-dataset	2.56 TB	210,014		Main folder
ffhq-dataset-v2.json	255 MB	1	JSON	Metadata including copyright info, URLs, etc.
images1024x1024	89.1 GB	70,000	PNG	Aligned and cropped images at 1024×1024
thumbnails128x128	1.95 GB	70,000	PNG	Thumbnails at 128×128
in-the-wild-images	955 GB	70,000	PNG	Original images from Flickr
tfrecords	273 GB	9	tfrecords	Multi-resolution data for StyleGAN and StyleGAN2
zips	1.28 TB	4	ZIP	Contents of each folder as a ZIP archive

C. StyleGAN3

StyleGAN3 tackles the texture sticking problem encountered by StyleGAN2 while making transitions such as changing facial expressions, and horizontal and vertical displacements. The generator architecture is shown in Fig. 7, the discriminator remains the same as indicated by Fig. 6. During transition animation, textures such as hair, beard, and fur would stick to the screen rather than moving along with the generated face object. This problem makes the morphing transition looks less natural in StyleGAN2. The texture sticking problem is mainly caused by unintentional positional references from the following sources: image borders, per-pixel noise inputs, positional encodings, and aliasing. These attributes provide the positional reference that results in the pixels generated by the network remaining unexpectedly on the same coordinates. To eliminate all sources of positional reference, the generator architecture must be equivariant that implies operations such as ReLU should not insert any positional reference. Traditional neural networks operate in the discrete domain. Therefore, the feature map generated by convolution and ReLU would be a discrete feature map. This creates the problem of aliasing since the discrete feature maps is considered as the result sampled the underlying continuous feature map.

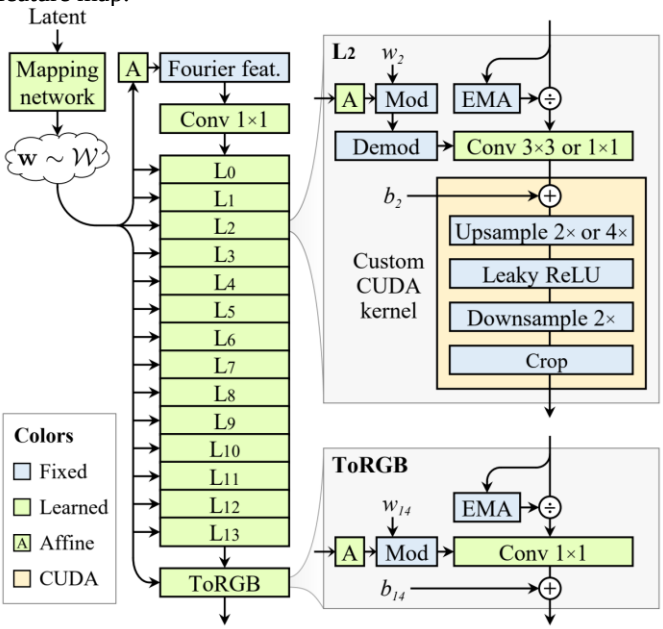


Fig. 7. StyleGAN3 generator architecture

StyleGAN3 replaced the learned 4x4x512 input constant layer in StyleGAN2 with a Fourier feature layer to emphasis and maintain the continuity of the input z [36]. The per-pixel noise inputs B shown in Fig. 4 is also removed since these input signals introduce unintentional positional references. The output skip connections are deleted. To compensate this change, an Exponential Moving Average (EMA) operation is added to normalize the input before every convolution layer. Applying nonlinearity such as Leaky ReLU in the continuous domain can introduce high frequency signals that cannot be captured properly by the output. Therefore, an ideal low pass filter (LPF) is included in the both the ReLU and Leaky ReLU operations. Convolving the LPF with Leaky ReLU must be performed in the continuous domain. This process is approximated in the custom CUDA kernel by upsampling the input signal, applying the Leaky ReLU, and then downsample the signal. A cropping operation is added at the end of the CUDA kernel to remove the image borders. These image border contains absolute image coordinates. Passing feature map that contains the image border information to the next layer tends to leak the absolute coordinate [37], which results in the texture sticking problem.

IV. Experiment

In this project, there are two different datasets have been used for testing the performance of the proposed models. The first one is FFHQ dataset, and the other is called anime face dataset.

A. FFHQ Dataset

Flickr-Faces-HQ (FFHQ) is a high-quality image dataset of human faces, originally created as a benchmark for generative adversarial networks (GAN). And this dataset was firstly been introduced in the paper of StyleGAN [34].

The dataset consists of 70,000 high-quality PNG images at 1024×1024 resolution and contains considerable variation in terms of age, ethnicity and image background. It also has good coverage of accessories such as eyeglasses, sunglasses, hats, etc. The images were crawled from Flickr, thus inheriting all the biases of that website, and automatically aligned and cropped using dlib. Only images under permissive licenses were collected. Various automatic filters were used to prune the set, and finally Amazon Mechanical Turk was used to remove the occasional statues, paintings, or photos of photos.

The whole data path is shown as TABLE II. Considering the computing resources, this project only used a part of all called

thumbnails128x128, which is basically resized version of all the images.

B. *Anime Face Dataset*

This dataset contains 63632 high-quality anime faces in the format of JPG images. And the dataset is clean and often used for varying projects with anime faces [38].

Fig. 8. Example images in Anime Face Dataset

C. *Experiments*

This section shows the simulation results through the FFHQ dataset and anime face dataset using the DCGAN, StyleGAN2 and StyleGAN3-r models. The natural image resolution used in FFHQ dataset is 128x128. Each image has the size of 128x128x3 which is 4 times bigger than the CelebA Dataset used in the original DCGAN publication by Radford et al. [26]. We decide to build a new DCGAN model to solve this dataset.

Compared with the model proposed by Radford et al., an extra convolution layer with batch normalization was allocated to the discriminator of the original DCGAN so that layer can act as a pooling layer and shorter the length and width of the images to half of the values. Meanwhile, a transposed convolution layer with batch normalization was added to the generator to keep the output size of the generator matching the input size of the discriminator.

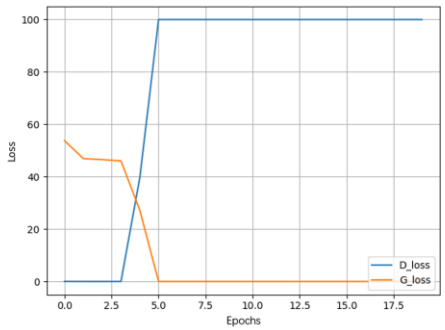


Fig. 9. The loss curve of the model with input_size=128*128

According to the result shown above, Fig. 9 is clear to tell that the loss of the generator is equal to zero after 5 epochs. The generator stopped updating its parameters afterwards even though the discriminator gave a 100% loss. Fig. 10 shows that the image produced by the generator was not able to bring any meaningful information but noise.

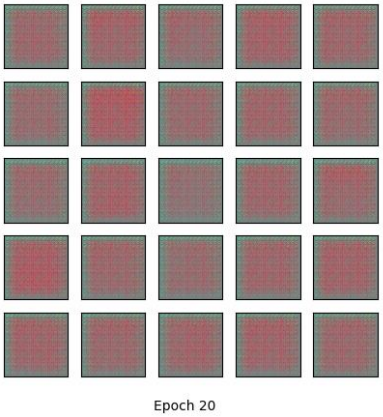


Fig. 10. The final output of the model with input_size=128*128

To solve this problem, the discriminator and the generator configurations were changed back to the original DCGAN model shown previously in section III.A. To match the image size with the discriminator's input size which is 64x64x3, a resize operation was added to the transforms function in the data loader. The images generated by DCGAN look realistic to people as well as to the discriminator. Fig. 11 shows that the generator loss converges to around 5 and the discriminator loss converges to 0 that indicating the discriminator cannot recognize the fake face generated by the generator.

But there is another disadvantage that the output images cannot be real enough in a high precision when compared to StyleGAN2 and StyleGAN3.

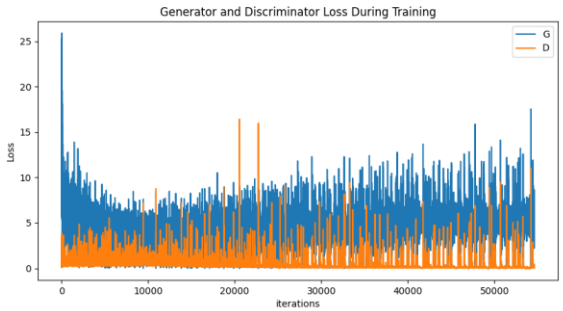


Fig. 11. The loss curve of the DCGAN model

The image generated by StyleGAN2 and StyleGAN3 are shown in Fig. 12(b) and (c), given that both the networks are trained with approximately the same GPU hours. The training images used for StyleGAN networks also have a resolution of 64x64.

To solve this problem, the discriminator and the generator configurations were changed exactly the same as the original DCGAN, and to match the image size with the discriminator's

input size which is 64*64*3, a resize command was added to the transforms part of the dataloader.

(a) DCGAN

(b) StyleGAN2

(c) StyleGAN3-r

Fig. 12. Image generated through the FFHQ Dataset

Although StyleGAN2 can generate high-quality images with detailed facial features, the generator suffers from the rotation and translation of the objects. Fig. 13 shows some images that did not recognize by the discriminator. We suspect characters in the images suffer from vertical translation, lateral translation, and rotations.

Fig. 13. Generated images with translations and rotations problem

To further determine the causes for the warped images, we trace back different versions of the same face through the training process that is shown in Fig. 14. The image on the top left is the final image generated after 3200 epochs of training. Every image moving to the right represents the same character generated ~200 epochs earlier. The two faces on the lower right are the earliest generation of this character that is reasonable and does not have many distortions. During training, we observe the texture sticking problem. Once the generator rotated the image, some facial features rotated along with the face, whereas the other features have the corresponding pixels stayed at the same coordinates. In addition, the quality of the final image produced by the generator network is closely related to the initial distribution of the images. We notice that for the images generated by the network that are photorealistic, these images have relatively accurate facial features as early as going through only 200 epochs of training.

Fig. 14. StyleGAN2 snapshots of the same face during training stages

Experiment results show that StyleGAN3-r does not have any problem with the texture sticking. Among all the images generated during training, none of the images are distorted by either rotation or translation. In the early training stages, StyleGAN3 learned the coordinates of the facial features relative to the face, rather than the absolute coordinates. Further training helps the generator to learn more detailed features with respect to the relative coordinates. The absolute coordinates are eliminated since the sources of undesired positional encoding such as image borders and aliasing are removed.

To validate the generalization of the proposed models, the Anime Face Dataset is included for testing. The images generated by the proposed GANs are shown in Fig. 15. The architectures and parameters used in the GANs remain the same as previously used in the FFHQ dataset.

(a) DCGAN

(b) StyleGAN2

(c) StyleGAN3

Fig. 15. Image generated through the Anime Face Dataset

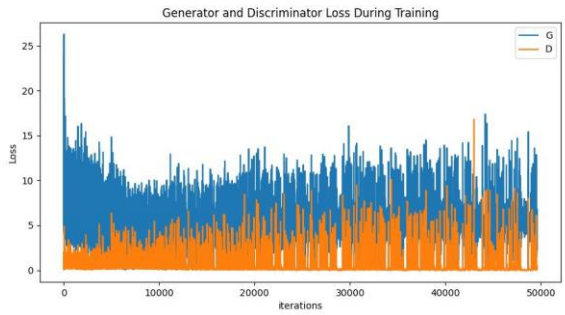


Fig. 16. The loss curve of a DCGAN model with Anime Face Dataset

The results obtained from the Anime Face Dataset are similar to the FFHQ Dataset. DCGAN can handle not only human faces, but also anime faces with totally different features in those images. Unlike FFHQ Dataset, anime faces are not based on real scenarios and textures. So, they always have extremely big and colorful eyes as well as colorful hair which are not commonly seen in real life.

Those obvious features make the anime faces less diverse than real human faces, and in some way, they let the GANs much easier to learn. As the result, the anime faces generated from the generator look more "real".

The training processes on StyleGAN2 and StyleGAN3 are much faster using the Anime Face Dataset. Both networks learned to generate accurate anime figures within 5-6 hours of training, whereas the FFHQ dataset requires 12+ hours for training. The performances of the StyleGAN models are measured in Fréchet inception distance (FID). The results shown in TABLE III. are the performances of the StyleGAN models trained from scratch with approximately the same GPU hours. StyleGAN3 has a higher FID than the StleGAN2 model. Due to hardware constrains, we believe the StyleGAN2 and StyleGAN3 models are not trained to their full potential, especially when StyleGAN3 needs more training since the generator architecture is more complex compared with StleGAN2.

TABLE III. FID scores of the StyleGAN models

	FFHQ FID	**Anima Face FID**
StyleGAN2	12.6	14.5
StyleGAN3-r	29	5.6

V. CONCLUSION

In this paper, DCGAN, StyleGAN2, and StyleGAN3 are proposed for the face image generation task. Two datasets are used to validate the performance and generalization of the proposed models, which are the FFHQ and Anime Face dataset. The FFHQ dataset contains more geometric features that require a long training period using GANs. To compare the performances of the models, the training images are resized to the 64x64 resolution. The generated images though different models also have the same resolution as the training images. The images generated by StyleGANs are more realistic as the generator architectures need more training. StyleGAN3-r has the most robust performance since the network is equivariance to rotation and translation.

VI. REFERENCES

[1] I. Goodfellow et al., "Generative adversarial nets," *Neural information processing systems,* vol. 27, 2014.

[2] A. Radford, L. Metz and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv:1511.06434,* 2015.

[3] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen and T. Aila, "Analyzing and improving the image quality of StyleGAN," arXiv:1912.04958, 2019.

[4] T. Karras, M. Aittala, S. Laine, E. Härk, J. Hellsten, J. Lehtinen and T. Aila, "Alias-free generative adversarial networks," *Advances in Neural Information Processing Systems,* vol. 34, 2021.

[5] T. Karras, S. Laine and T. Aila, "A style-based generator architecture for generative adversarial networks," Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.

[6] B. Chao, "Anime-Face-Dataset," 2019. [Online]. Available: https://github.com/bchao1/Anime-Face-Dataset.

[7] A. Nair, "Computer Vision Market by Component, Product, Application, and Vertical (Industrial and Non-Industrial): Global Opportunity Analysis and Industry Forecast, 2020–2030," *Allied Market Research,* 2022.

[8] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved Texture Networks: Maximizing Quality and Diversity in Feed-Forward Stylization and Texture Synthesis," *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 6924-6932.*

[9] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu and T. S. Huang, "Generative Image Inpainting With Contextual Attention," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 5505-5514.

[10] W. Cho, S. Choi, D. K. Park, I. Shin and J. Choo, "Image-To-Image Translation via Group-Wise Deep Whitening-And-Coloring Transformation," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 10639-10647.

[11] X. Wu, K. Xu and P. Hall, "A Survey of Image Synthesis and Editing with Generative Adversarial," in Tsinghua Science and Technology, vol. 22, no. 6, pp. 660-674, 2017, doi: 10.23919/TST.2017.8195348..

[12] E. H. Weissler, T. Naumann, T. Andersson, et al., "The role of machine learning in clinical research: transforming the future of evidence generation," *Trials 22,* p. 537, 2021, https://doi.org/10.1186/s13063-021-05489-x.

[13] P. Shah, F. Kendall, S. Khozin, et al., "Artificial intelligence and machine learning in clinical development: a translational perspective," *npj Digit. Med.,* vol. 2, p. 69, 2019, https://doi.org/10.1038/s41746-019-0148-3.

[14] N. Tran, V. Tran, N. Nguyen, T. Nguyen and N. Cheung, "On Data Augmentation for GAN Training," *arXiv,* 2020, arXiv:2006.05338.

[15] C. Bowles et al., "GAN Augmentation: Augmenting Training Data using Generative Adversarial Networks," *arXiv,* 2018, arXiv:1810.10863.

[16] S. Tripathy, J. Kannala and E. Rahtu, "ICface: Interpretable and Controllable Face Reenactment Using GANs," Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2020, pp. 3385-3394.

[17] J. Yu et al., "Toward Realistic Face Photo–Sketch Synthesis via Composition-Aided GANs," *IEEE Transactions on Cybernetics*, vol. 51, no. 9, pp. 4350-4362, Sept. 2021, doi: 10.1109/TCYB.2020.2972944.

[18] J. Despois, F. Flament and M. Perrot, "AgingMapGAN (AMGAN): High-Resolution Controllable Face Aging with Spatially-Aware Conditional GANs," *Bartoli, A., Fusiello, A. (eds) Computer Vision – ECCV*, 2020.

[19] A. Hassanpour et al., "E2F-GAN: Eyes-to-Face Inpainting via Edge-Aware Coarse-to-Fine GANs," in IEEE Access, vol. 10, pp. 32406-32417, 2022, doi: 10.1109/ACCESS.2022.3160174.

[20] A. Shoshan, N. Bhonker, I. Kviatkovsky and G. Medioni, "GAN-Control: Explicitly Controllable GANs," Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

[21] S. He, W. Liao and M. Yang, et al., "Disentangled Lifespan Face Synthesis," Proceedings of the IEEE/CVF International Conference on Computer Vision. 2021.

[22] H. Yang, L. Chai, Q. Wen, S. Zhao, Z. Sun and S. He, "Discovering Interpretable Latent Space Directions of GANs Beyond Binary Attributes," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021.

[23] H. Changhee et al., "Infinite Brain MR Images: PGGAN-based Data Augmentation for Tumor Detection," *Neural approaches to dynamics of signal exchanges*, pp. 291-303, 2020.

[24] K. PNVR, H. Zhou and D. Jacobs, "Sharingan: Combining synthetic and real data for unsupervised geometry estimation," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020.

[25] S. Zhao, H. Fu, G. Gong and D. Tao, "Geometry-Aware Symmetric Domain Adaptation for Monocular Depth Estimation," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

[26] S. Sengupta, et al., "Sfsnet: Learning shape, reflectance and illuminance of facesin the wild," Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[27] A. Geiger, P. Lenz, C. Stiller and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231-1237, 2013.

[28] S. Zafeiriou et al., "The Photoface database," pp. 132-139, CVPR 2011 WORKSHOPS, 2011.

[29] H. Ren and P. Tzvi, "Learning inverse kinematics and dynamics of a robotic manipulator using generative adversarial networks," *Robotics and Autonomous Systems*, vol. 124, p. 103386, 2020.

[30] O. Wiles, A. S. Koepke and A. Zisserman, "X2face: A network for controlling face generation using images, audio, and pose codes," Proceedings of the European conference on computer vision (ECCV). 2018.

[31] A. Nagrani, J. S. Chung and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv:1706.08612*, 2017.

[32] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021.

[33] Y. Shen, et al., "Interpreting the latent space of gans for semantic face editing," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020.

[34] Y. Deng, et al., "Disentangled and controllable face image generation via 3d imitative-contrastive learning," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020.

[35] M. Kowalski, et al., "Config: Controllable neural face image generation," European Conference on Computer Vision. Springer, Cham, 2020.

[36] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," arXiv preprint arXiv:1710.10196, 2017.

[37] M. Tancik et al., "Fourier features let networks learn high frequency functions in low dimensional domains," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7537-7547, 2020.

[38] M. A. Islam, S. Jia, and N. D. B. Bruce, "How much position information do convolutional neural networks encode?," *In Proc. ICLR*, 2020.