Accredited Ranking SINTA 2
Decree of the Director General of Higher Education, Research and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online on: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 7 No. 5 (2023) 1049 - 1055 ISSN Media Electronic: 2580-0760

Application of Deep Convolutional Generative Adversarial Networks to Generate Pose Invariant Facial Image Synthesis Data

Jagad Nabil Tuah Imanda[1], Fitra Abdurrachman Bachtiar[2*], Achmad Ridok[3]
[1,2,3]Informatics Engineering, Faculty of Computer Science, Brawijaya University, Malang, Indonesia
[1]imandajagad01@gmail.com, [2]fitra.bachtiar@ub.ac.id, [3]acridokb@ub.ac.id

Abstract

Even though Artificial Intelligence is advancing, artificial intelligence can still find it difficult to solve problems that are easy for humans to do but difficult for computers to describe, such as facial recognition. There are problems related to the existing facial recognition model, namely the facial recognition model. The model is still unable to recognize facial shapes that are not in a perfect state due to several factors. Among several factors, the most influencing factor is the position of the face. Therefore, in this study, Deep Convolutional Generative Adversarial Networks (DCGAN) will be applied to generate fake image data with varying face positions. This research will be carried out starting from collecting data, processing data, designing, and training models, hyperparameter tuning, and lastly analyzing test results. Based on the results of hyperparameter tuning that carried out sequentially, the best hyperparameter combination produced is 200 epoch, 0.002 Generator learning rate, 0.5 Generator momentum/beta1, Adam as Generator optimizer, 0.0002 Discriminator learning rate, 0.5 Discriminator momentum/beta1, and Adam as Discriminator optimizer. The hyperparameter combination gives a result with FID score of 74.05. Based on testing with human observer, generated fake images has relatively good results, but there are still few bad fake image results.

Keywords: *face recognition; pose invariant; generative adversarial networks; deep convolutional generative adversarial networks; hyperparameter tuning*

1. Introduction

Nowadays, the field of technology is growing rapidly. One of the results of developments in the field of technology is artificial intelligence which has a level of intelligence that almost resembles human intelligence. One area of artificial intelligence is computer vision. Computer vision is a field of artificial intelligence that develops techniques to help computers see and understand content from digital images such as photos and videos [1]. Problems that exist in the field of computer vision easy for humans to recognize or to detect because indirectly these problems have been resolved by humans. However, this problem is still difficult to overcome in the field of computer vision because of the limited perception of vision and recognition on computers.

Face recognition is an important research topic in the field of computer vision. With the advancement of a carefully designed Convolutional Neural Network (CNN), the performance of facial recognition models has improved significantly. The performance of a Convolutional Neural Network (CNN) based facial recognition model is highly dependent on the amount of data and the variety of data used for training. However, collecting data to conduct large and varied amounts of training is costly. A face recognition model can fail to recognize a face if the test data contains a face that is not present in the training data [2]. The problem that is also experienced in the field of facial recognition is regarding the facial pose that will be recognized by the model. Some models unable to recognize facial shapes that are not in a perfect state due to several factors that cause faces to be unrecognizable. This condition includes face position, lighting, expression, and obstacles covering the face. Among several factors mentioned, the position of the face is the most influencing factor in the facial recognition model [3].

Various efforts and research have been carried out to solve the problem of the amount and variety of data needed to train facial recognition models. One of the ways is to generate fake face image data based on a dataset using Generative Adversarial Networks (GAN) method [4]. There has been research conducted to generate fake face images with Generative Adversarial Networks (GAN), some examples are research conducted using Distangled Representation Learning Generative Adversarial Networks (DR-GAN) method

Accepted: 24-05-2023 | Received in revised: 16-09-2023 | Published: 30-09-2023

[5] with Generators that receive input in the form of images, not in the form of random noise and this research succeeded in increasing the accuracy of the model by up to 20%, then the research was continued that proposed the CR-GAN method [6] which is a development of the DR-GAN method by using the self-supervised learning method to produce several poses from -90° to 90° and produce better identity similarity than the DR-GAN method, then there was research conducted using the PosIX Generative Adversarial Networks (PosIX-GAN) method [7] using the Head Pose Image Database (HPID) [8] and Multi-PIE datasets [9] and resulting in a drastic increase in accuracy of up to 16% for face recognition with a side view of several degrees.

Based on previous studies regarding fake image data generation, there are still deficiencies in research that has been done in this field. There are two main drawbacks in previous study. First, the deficiencies in previous studies were the lack of hyperparameter information on the model training that was carried out. Second, lack of results of the tests carried out. The lack or results is because these studies only displayed performance improvements from fake data that were created to be used for training. In addition, there were no scores or parameters to measure the results of the fake data produced by research that has been done.

In this study, data creation of fake face images was carried out with various poses. The creation of the fake image data is done by the DCGAN method [10]. This study uses the DCGAN method because in the research that was conducted [11], [12] have proven that DCGAN can create augmented data that can help a classification model to better recognize data. Based on [13], the image results created by the DCGAN method outperform several other methods such as Variational Auto-Encoder (VAE) [14]. Due to the lack of information on the hyperparameters used in model training and scores to measure fake image results, a search for the best hyperparameter combination and measurement of the Frechet Inception Distance (FID) score [15] on fake face images will be carried out to measure image quality generated by the DCGAN model in this research. In this study, testing was also carried out with the help of human observers to assess generated fake face image data and to compare the results of the assessment with the FID score result.

2. Research Methods

This research was conducted in several steps as can be seen in Figure 1.

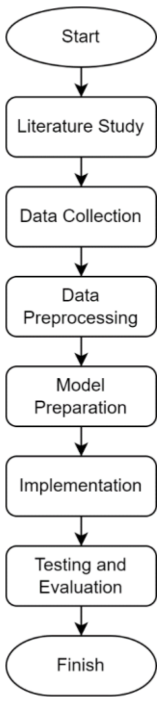


Figure 1. Research Steps

2.1. Dataset

This study uses Queen Mary University of London Multiview Face (QMUL Multiview) Dataset [16]obtained from Queen Mary University of London website. This dataset consists of facial images from 48 respondents, where one respondent has 133 facial images with different poses ranging from facing forward to facing up or down a few degrees. After preprocessing, there is only 4565 total images that are ready to be used as training data from this dataset.

2.2. Data Preprocessing

The dataset that has been collected will be preprocessed. The first step of the preprocessing is to convert the image data to RGB format. Next, the image will be converted to *.jpg format to make sure the torchvision library can process the image. After that, the image will be resized to 64×64 because it will be processed by DCGAN. Lastly, the image will be standardized to a range of -1 to 1.

2.3. Model Architecture

This study uses DCGAN, which consists of Generator and Discriminator model. Architecture of Generator model that used in this study can be seen in Figure 2, and architecture of Discriminator model that used in this study can be seen in Figure 3.

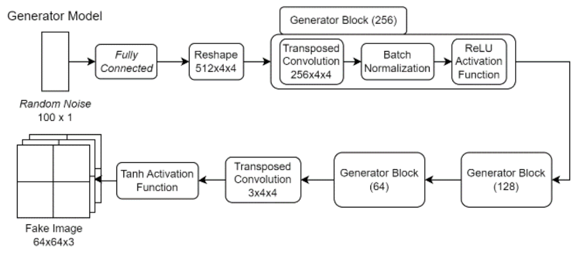


Figure 2. Generator Model Architecture

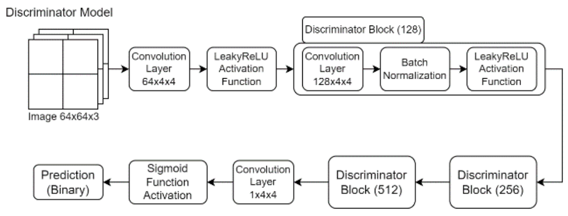

Discriminator Model
Image 64x64x3
Convolution Layer 64x4x4
LeakyReLU Activation Function
Discriminator Block (128)
Convolution Layer 128x4x4
Batch Normalization
LeakyReLU Activation Function
Discriminator Block (256)
Discriminator Block (512)
Convolution Layer 1x4x4
Sigmoid Function Activation
Prediction (Binary)

Figure 3. Discriminator Model Architecture

Generator model accepts random noise as input. The random noise is then entered into the fully connected layer and get reshaped so that is has a size of 512×4×4. After reshaped, it will be entered into the Generator Block three times. Generator block consists of transposed convolution layer, batch normalization layer, and ReLU activation function [17]. After that, it will be closed by Tanh activation function [18]. The output of Generator model is 64×3×3 matrix which represents an image with RGB color.

Discriminator model accepts an image with 64×64×3 size as input. The image is then inserted into the convolution layer with ReLU activation function, and then entered to the Discriminator Block three times. Discriminator block consists of convolution layer, batch normalization layer, and LeakyReLU activation function [19]. Then, the vector re-entered into the convolution layer, and it will end in the Sigmoid activation function [20]. The output of Discriminator model is binary prediction that predicts if the input image is fake or real.

2.4 Testing Scenario

In this study, testing will be carried out by hyperparameter tuning to find the best hyperparameter combination. The hyperparameter that will be tested are number of epochs, Generator learning rate, Generator momentum, Generator optimizer, Discriminator learning rate, Discriminator momentum, and Discriminator optimizer. The process of hyperparameter tuning will be carried out sequentially because of the limitation of time and computing power. The testing is also carried out by human observers to assess generated fake image.

2.5 Evaluation Method

In this study, FID score is used to determine best hyperparameter value during hyperparameter tuning. If the FID score is smaller, then the result is better than higher FID score. Besides FID score, human observer is also used to give scores to generated fake images. The score that given to the generated fake image can be good, bad, or neutral.

3. Results and Discussions

3.1 Results of Test on Number of Epochs

Hyperparameter tuning on number of epochs is carried out using six values, which are 25, 50, 100, 200, 400, and 700. Other hyperparameter values are fixed. The results of hyperparameter tuning on number of epochs can be seen in Table 1 and Figure 4.

Table 1. Results of Test on Number of Epochs

Number of Epochs	FID Score
25	261.66
50	190.31
100	150.61
200	**119.00**
400	149.48
700	205.12

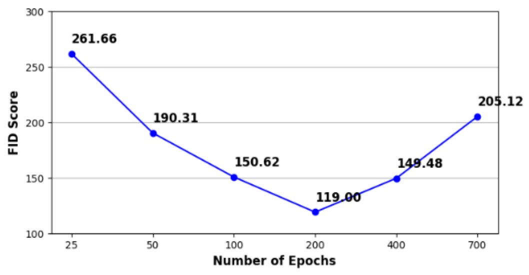


Figure 4. Graph of Number of Epochs Test Result

Based on the results of the test, the best number of epochs obtained was 200 with an FID score of 119.00. The data can be seen in Table 1.

Number of epochs affects fake images produced by the model. The number of epochs cannot be too small because it will produce fake images that have mosaic shape, and the number of epochs cannot be too large because the model will be overfit and produces fake images that have mosaic shape and that image do not form face images.

3.2. Results of Test on Generator Learning Rate

Hyperparameter tuning on Generator learning rate is carried out using three values, which are 0.00002, 0.0002, and 0.002. Other hyperparameter values are fixed, while for number of epochs will have a value of 200 which is the best result from the previous test. The results of hyperparameter tuning on Generator learning rate can be seen in Table 2 and Figure 5.

Based on the results of the test, the best Generator learning rate obtained was 0.002 with FID score of 74.05. The data can be seen in Table 2.

Table 2. Results of Test on Generator Learning Rate

Generator Learning Rate	FID Score
0.00002	221.07
0.0002	119.00
0.002	**74.05**

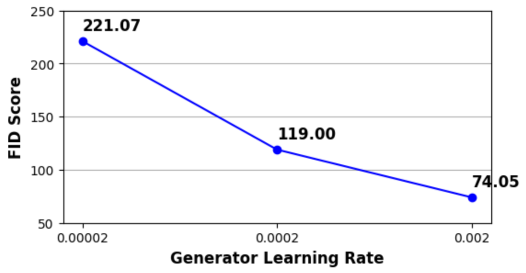


Figure 5. Graph of Generator Learning Rate Test Result

Generator learning rate also affects fake images produced by the model. Generator learning rate can't be too small, because the smaller Generator learning rate, the changes of the weight will progress slowly, and otherwise. Fake images that are generated while the learning rate is 0.00002 and 0.0002 still have many mosaic shapes.

3.3 Results of Test on Generator Momentum

Hyperparameter tuning on Generator momentum is carried out using three values, which are 0.1, 0.5, and 0.9. Other hyperparameter values are fixed, while for number of epochs and Generator learning rate will have a value of 200 and 0.002 which is the best result from the previous test. The results of hyperparameter tuning on Generator momentum can be seen in Table 3 and Figure 6.

Table 3. Results of Test on Generator Momentum

Generator Momentum	FID Score
0.1	122.96
0.5	**74.05**
0.9	166.05

Based on the results of the test, the best Generator momentum obtained was 0.5 with FID score of 74.05. The data can be seen in Table 3.

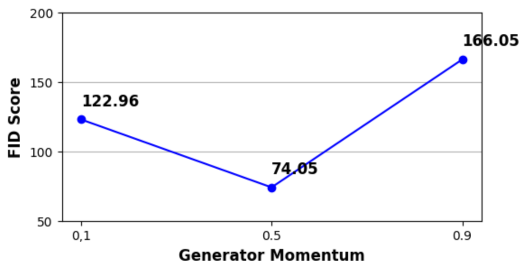


Figure 6. Graph of Generator Momentum Test Result

Generator momentum also affects fake images produced by the model. Small Generator momentum tends to produce bad fake images that have mosaic shape because the model training is slower. Meanwhile, high Generator momentum also tends to produce better fake images, but not as good as 0.5 Generator momentum, because the higher the Generator momentum, the model training will be overfit.

3.4 Results of Test on Generator Optimizer

Hyperparameter tuning on Generator optimizer is carried out using three optimizers, which are SGD, Adam, and RMSProp. Other hyperparameter values are fixed, while for number of epochs, Generator learning rate, and Generator momentum will have a value of 200, 0.002, and 0.5 which is the best result from the previous test. The results of hyperparameter tuning on Generator optimizer can be seen in Table 4 and Figure 7.

Table 4. Results of Test on Generator Optimizer

Generator Optimizer	FID Score
SGD	173.17
Adam	**74.05**
RMSProp	161.54

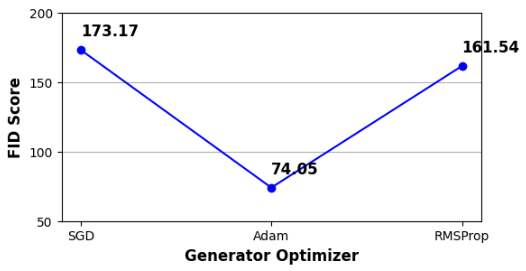


Figure 7. Graph of Generator Optimizer Test Result

Based on the results of the test, the best Generator optimizer was Adam with FID score of 74.05. The data can be seen in Table 4.

When using SGD as Generator optimizer, the model produces fake images that did not form faces at all and have many mosaic shapes on the image. When using RMSProp as Generator optimizer, the model produces fake images that formed face image but there's still many mosaic shapes on the image.

3.5 Results of Test on Discriminator Learning Rate

Hyperparameter tuning on Discriminator learning rate is carried out using three values, which are 0.00002, 0.0002, and 0.002. Other hyperparameter values are fixed, while for number of epochs, Generator learning rate, Generator momentum, and Generator optimizer will have a value of 200, 0.002, 0.5, and Adam which is the best result from the previous test. The results of hyperparameter tuning on Discriminator learning rate can be seen in Table 5 and Figure 8.

Table 5. Results of Test on Discriminator Learning Rate

Discriminator Learning Rate	FID Score
0.00002	256.76
0.0002	**74.05**
0.002	177.36

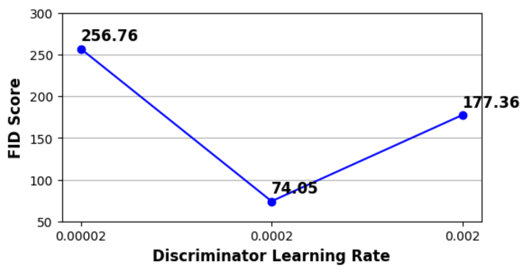


Figure 8. Graph of Discriminator Learning Rate Test Result

Based on the results of the test, the best Discriminator learning rate obtained was 0.0002 with FID score of 74.05. The data can be seen in Table 5.

Discriminator learning rate also affects the image produced by the model. The lower the Discriminator learning rate, the weight will be not changed rapidly hence the produced fake images did not form face image yet. The higher the Discriminator learning rate, the weight changed rapidly hence the produced fake images forms face image but there is a lot of blur mosaics.

3.6. Results of Test on Discriminator Momentum

Hyperparameter tuning on Discriminator momentum is carried out using three values, which are 0.1, 0.5, and 0.9. Other hyperparameter values are fixed, while for number of epochs, Generator learning rate, Generator momentum, Generator optimizer, and Discriminator learning rate will have a value of 200, 0.002, 0.5, Adam, and 0.0002 which is the best result from the previous test. The results of hyperparameter tuning on Discriminator momentum can be seen in Table 6 and Figure 9.

Table 6. Results of Test on Discriminator Learning Rate

Discriminator Momentum	FID Score
0.1	138.11
0.5	**74.05**
0.9	261.83

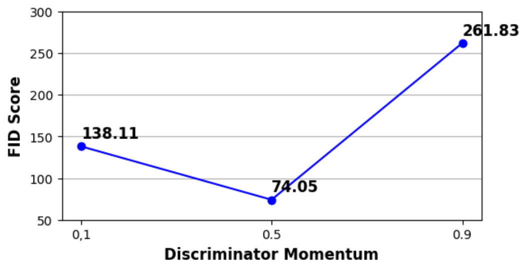


Figure 9. Graph of Discriminator Momentum Test Result

Based on the results of the test, the best Discriminator momentum obtained was 0.5 with FID score of 74.05. The data can be seen in Table 6.

Discriminator momentum also affects fake images that is produced by the model. The lower the Discriminator momentum value, the weight updates on the model happened slowly hence the result of the image still have many mosaics. The higher the Discriminator momentum value, the weight updates happened quickly hence the result of the image did not look like face image because of the overfit.

3.7 Results of Test on Discriminator Optimizer

Hyperparameter tuning on Discriminator optimizer is carried out using three optimizers, which are SGD, Adam, and RMSProp. Other hyperparameter values such as number of epochs, Generator learning rate, Generator momentum, Generator optimizer, Discriminator learning rate, and Discriminator momentum will have a value of 200, 0.002, 0.5, Adam, 0.0002, and 0.5 which is the best result from the previous test. The results of hyperparameter tuning on Discriminator optimizer can be seen in Table 7 and Figure 10.

Table 7. Results of Test on Discriminator Optimizer

Discriminator Optimizer	FID Score
SGD	518.87
Adam	**74.05**
RMSProp	125.60

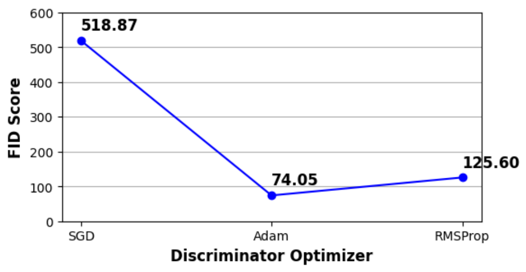


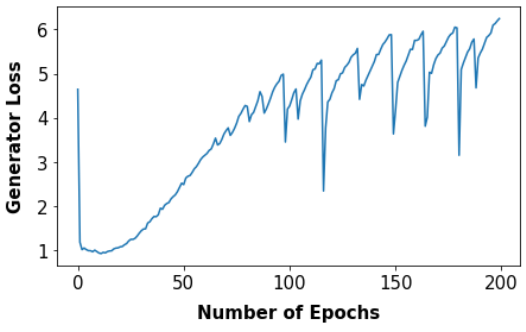
Figure 10. Graph of Discriminator Optimizer Test Result

Based on the results of the test, the best Discriminator optimizer was Adam with an FID score of 74.05. The data can be seen in Table 7.

Discriminator optimizer also affects fake images produced by the model. When using SGD as Discriminator optimizer, fake images that is produced by the model only contains mosaic blur. When using RMSProp as Discriminator optimizer, fake images that is produced by the model forms face image but it still has many mosaics.

3.8 Results of Best Hyperparameter Combination

After obtaining best hyperparameter value in previous tests, each hyperparameter value is then collected to obtain best hyperparameter combination. Based on the results, best hyperparameter combination consists of 200 number of epochs, 0.002 Generator learning rate, 0.5 Generator momentum, Adam Generator optimizer, 0.0002 Discriminator learning rate, 0.5 Discriminator momentum, and Adam Discriminator optimizer. From these hyperparameter combination, the results of generated fake image produced 74.05 FID score, 6.2477 Generator loss value, and 0.0089 Discriminator loss value at 200th epoch. Graphs of Generator loss progression and Discriminator loss progression can be seen in Figure 10.



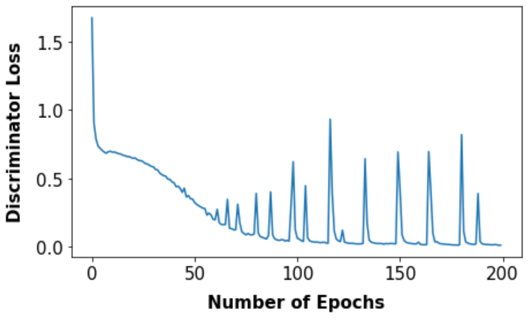


Figure 10. Graph of Generator and Discriminator Loss Progression

Based on Figure 10 that can be seen above, the Generator loss tend to be large during first epoch because the training process has just started so that the weights and noise are still initialized randomly and do not form a certain pattern. However, starting in the next epoch, the Generator loss value immediately decreased because the Generator model already recognized the pattern from training data. Furthermore, the progression of Generator loss will continuously be increasing, this is because the fake image generated by the Generator will get better.

Based on Figure 10 that can be seen above, the Discriminator loss tend to be large during first epoch because the training process has just started so that the weights are still initialized randomly and do not form a certain pattern. Then the Discriminator loss decreased gradually upon the training process because the Discriminator will find it more difficult to distinguish between fake and original images.

3.9 Results of Human Observer Test

Human observer test is carried out by providing a form containing a sample of the fake image generated by DCGAN model with best hyperparameter combination. The human observer then gives a score between good, bad, or neutral in the form provided. Generated image that will be scored by human observer can be seen in Figure 11.

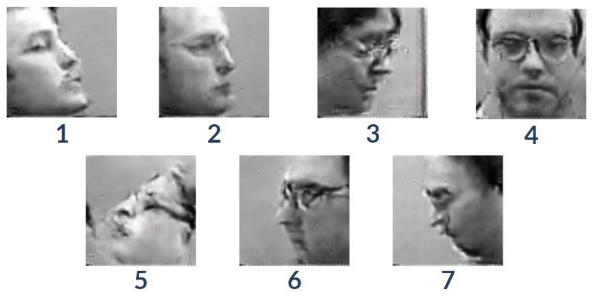


Figure 11. Generated Images that used for testing

The test with human observer was carried out with the help of 5 respondents, namely 5 students from the Faculty of Computer Science, Brawijaya University. The results of testing with human observers can be seen in Table 8.

Table 8. Results of Human Observer Testing

Evaluator	Score Accumulation		
	Good	Bad	Neutral
1	**4**	2	1
2	2	2	**3**
3	1	2	**4**
4	2	**4**	1
5	**6**	0	1

After testing has been done, there were 35 total results given by 5 human observers. The summary of the human evaluation composition are as follows: a total score of 15 with a good predicate, 10 scores with a bad predicate, and 10 scores with a neutral predicate. Based on the results, it can be concluded that the generated fake image has quite good results based on human observation.

3.10 Generated Images

There are some samples of Generated images that can be seen in Figure 12. Most of generated fake images are not very good because some of them still have not formed facial image. Moreover, there are many fake images that still have noises.

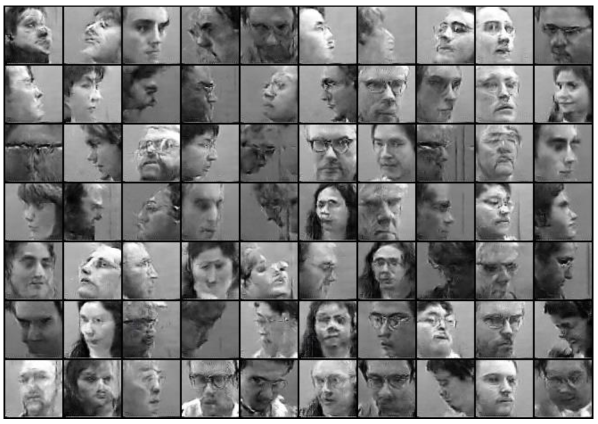


Figure 12. Sample of Generated Images

4. Conclusion

Based on hyperparameter tuning testing that have been done, the best hyperparameter combination are 200 number of epochs, 0.002 Generator learning rate, 0.5 Generator momentum, Adam Generator optimizer, 0.0002 Discriminator learning rate, 0.5 Discriminator momentum, and Adam Discriminator optimizer. With the best hyperparameter combination that has been obtained from test, the result of FID Score is 74.05.

Based on test conducted with human observers, generated fake face image data gets relatively good results. However, there are generated fake image that have bad results because the image has not yet formed a face shape and there is still plenty of noises. As for the generated fake image that is considered good, it has formed face image perfectly but still has a little noise.

This study is an early attempt in generating image using GAN. Further study may consider implementing the model to make the research result even better. Next study may analyze the model thoroughly, namely in the Discriminator model that tends to be stronger than Generator model so that the generated fake image is still not optimal. A dropout layer can be added to Discriminator model to avoid Discriminator getting stronger than Generator. Other direction may analyze the hyperparameter tuning. This study uses sequential method due to time and computing power limitations. It is hoped that in future research, other hyperparameter tuning methods such as Grid Search can be used to look for even better hyperparameter combinations.

Acknowledgment

The server used in this study was provided by Intelligent Systems Laboratory, Faculty of Computer Science Brawijaya University.

References

[1] J. Brownlee, *Deep Learning for Computer Vision Image Classification, Object Detection and Face Recognition in Python*. Deep Learning For Computer Vision, 2019.

[2] H.-C. Shao, K.-Y. Liu, C.-W. Lin, and J. Lu, "DotFAN: {A} Domain-transferred Face Augmentation Network for Pose and Illumination Invariant Face Recognition," *CoRR*, vol. abs/2002.09859, 2020, [Online]. Available: https://arxiv.org/abs/2002.09859.

[3] E.-J. Tsai and W.-C. Yeh, "{PAM:} Pose Attention Module for Pose-Invariant Face Recognition," *CoRR*, vol. abs/2111.11940, 2021, [Online]. Available: https://arxiv.org/abs/2111.11940.

[4] I. J. Goodfellow *et al.*, "Generative Adversarial Networks," 2014, doi: https://doi.org/10.48550/arXiv.1406.2661.

[5] L. Tran, X. Yin, and X. Liu, "Representation Learning by Rotating Your Faces," *CoRR*, vol. abs/1705.11136, 2017, [Online]. Available: http://arxiv.org/abs/1705.11136.

[6] Y. Tian, X. Peng, L. Zhao, S. Zhang, and D. N. Metaxas, "{CR-GAN:} Learning Complete Representations for Multi-view Generation," *CoRR*, vol. abs/1806.11191, 2018, [Online]. Available: http://arxiv.org/abs/1806.11191.

[7] A. Bhattacharjee, S. Banerjee, and S. Das, "PosIX-GAN: Generating Multiple Poses Using GAN for Pose-Invariant Face Recognition BT - Computer Vision – ECCV 2018 Workshops," 2019, pp. 427–443.

[8] N. Gourier and J. Crowley, "Estimating Face orientation from Robust Detection of Salient Facial Structures," *FG Net Work. Vis. Obs. Deictic Gestures*, Jan. 2004.

[9] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-PIE," in *2008 8th IEEE International Conference on Automatic Face & Gesture Recognition*, 2008, pp. 1–8, doi: 10.1109/AFGR.2008.4813399.

[10] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," 2016, doi: https://doi.org/10.48550/arXiv.1511.06434.

[11] K. Wu, Y. Yu, X. Zhang, J. Li, and Q. Zhang, "Application of Face Data Augmentation Based on Rotate-and-Render-DCGAN in Campus Security," in *2020 IEEE 3rd International Conference of Safe Production and Informatization (IICSPI)*, 2020, pp. 561–564, doi: 10.1109/IICSPI51290.2020.9332396.

[12] O. Adedeji, P. Owoade, O. Ajayi, and O. Arowolo, "Image Augmentation for Satellite Images," 2022, doi: https://doi.org/10.48550/arXiv.2207.14580.

[13] S. Bond-Taylor, A. Leach, Y. Long, and C. G. Willcocks, "Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7327–7347, 2022, doi: 10.1109/TPAMI.2021.3116668.

[14] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," 2022, doi: https://doi.org/10.48550/arXiv.1312.6114.

[15] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, G. Klambauer, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Nash Equilibrium," *CoRR*, vol. abs/1706.08500, 2017, [Online]. Available: http://arxiv.org/abs/1706.08500.

[16] *Head Pose Estimation*. Queen Mary University of London, 2001.

[17] V. Nair and G. Hinton, *Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair*, vol. 27. 2010.

[18] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[19] A. L. Maas, "Rectifier Nonlinearities Improve Neural Network Acoustic Models," 2013, [Online]. Available: https://api.semanticscholar.org/CorpusID:16489696.

[20] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986, doi: 10.1038/323533a0.