Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.030557
Article

Multi-Generator Discriminator Network Using Texture-Edge Information

Kyeongseok Jang[1], Seongsoo Cho[2] and Kwang Chul Son[3,*]

[1]Department of Plasma Bio Display, Kwangwoon University, 20, Gwangun-ro, Nowon-gu, Seoul, 01897, Korea
[2]School of Software, Soongsil University, 369, Sangdo-ro, Dongjak-gu, Seoul, 06978, Korea
[3]Department of Information Contents, Kwangwoon University, 20, Gwangun-ro, Nowon-gu, Seoul, 01897, Korea
*Corresponding Author: Kwang Chul Son. Email: kcson@kw.ac.kr
Received: 29 March 2022; Accepted: 23 June 2022

Abstract: In the proposed paper, a parallel structure type Generative Adversarial Network (GAN) using edge and texture information is proposed. In the existing GAN-based model, many learning iterations had to be given to obtaining an output that was somewhat close to the original data, and noise and distortion occurred in the output image even when learning was performed. To solve this problem, the proposed model consists of two generators and three discriminators to propose a network in the form of a parallel structure. In the network, each edge information and texture information were received as inputs, learning was performed, and each character was combined and outputted through the Combine Discriminator. Through this, edge information and distortion of the output image were improved even with fewer iterations than DCGAN, which is the existing GAN-based model. As a result of learning on the network of the proposed model, a clear image with improved contour and distortion of objects in the image was output from about 50,000 iterations.

Keywords: Deep learning; convolution neural network; generative adversarial network; edge information; texture information

1 Introduction

In recent years, deep learning is the most popular field and active research has been conducted [1]. A study is underway to incorporate deep learning in a variety of areas, including device diagnosis and repair [2], exploration of the optimal location of the wireless network [3], analysis of the impact of learning posture on learning [4], personal authentication [5], security [6], medical [7], and robots [8,9]. There is little difference between humans and deep learning, especially in the field of image and image processing [10]. Deep learning can be largely classified as Supervised learning and Unsupervised learning. Supervised learning performs learning using a training data set consisting of original data and a correct answer data set, a labeled data set. Since learning performance is affected by the size of the training data set, various and sufficient data are collected. However, in most cases, it is difficult to construct a training data set personally, except for the training data set provided because it is difficult to collect sufficient data. Unsupervised learning, in contrast, is learning performed using only the original

data. The original data is used to analyze the shape and pattern of data the model learns by itself so that even if only a small amount of data is provided, the success of the learning process is not compromised unduly. A representative model of unsupervised learning is the generative adversarial network (GAN) [11]. The derivative models of the GAN include deep convolutional generative adversarial network (DCGAN), Least-Squares Generative Adversarial Network (LS GAN), CycleGAN, and StarGAN, and BigGAN [12–16]. Recently, Semi-Supervised models using the characteristics of supervised learning and unsupervised learning are being studied, as to how to add training data sets using DCGAN to improve the learning performance of the supervised learning model and others [17–19]. However, DCGAN has a disadvantage in that it takes a lot of learning time to output an image with improved pixel noise and image distortion.

In this study, we propose a learning model based on DCGAN, which uses the Texture of the Original Image and Edge information. The name of the model was proposed by newly creating the term Texture-Edge GAN (T-E GAN). In the existing DCGAN, the original image as input, and overall learning was performed without using specific features for learning. Therefore, much learning was required before obtaining an improved image. In the proposed method, in order to obtain improved results with less learning amount than the existing DCGAN, the Texture information and Edge information of the original image were learned by the network composed of two Generators and three Discriminators, respectively. The dataset used in the training of the proposed model is 202,599 images of Large-Scale Celeb-Faces Attributes (CelebA) [20]. The model provided through learning improved pixel noise and distortion with less learning amount than the existing DCGAN.

2 Background Theory

2.1 Convolution Neural Network

Convolution Neural Networks (CNN) is a type of Artificial Neural Network (ANN) that uses convolution arithmetic [21,22]. It is one of the most used models in the field of deep learning-based image processing and has been proposed to effectively process images using convolution operations. The CNN is divided into a part for extracting image features and a part for extracting image features. In the feature extraction stage, multiple convolutions and pool hierarchies are arranged. In the classification stage, the feature values extracted through the Fully-connected Layer (FC) are classified. Typical CNN structures include LeNet, which classifies cursive datasets (MNIST), and AlexNet, which classifies using the GPU [23]. Fig. 1 shows the structure of LeNet.

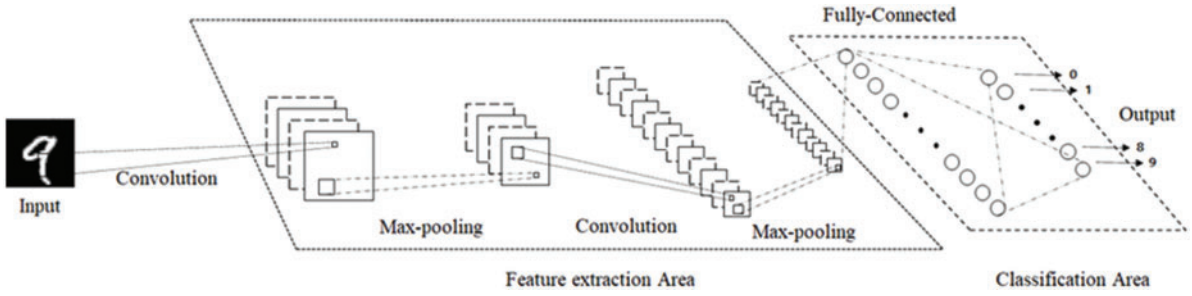


Figure 1: Structure of LeNet

The structure of CNN is composed of Convolution Layer, Pooling Layer, and Fully Connected Layer. The Convolution Layer in the feature extraction stage is a very important component in the CNN and uses the Convolution Filter to maintain the location information of the input image and

extract features. Moreover, the feature of the extracted input image is derived from one sheet. The Pooling Layer reduces the size of the image while preserving important information in the input image. Therefore, it is used for the purpose of emphasizing specific data or canceling noise. The types of pooling are Max, Average, and Min, and the maximum pooling is mainly used [24]. In the classification stage, the Fully Connected Layer is used to classify images using all extracted nodes, with all nodes in the previous layer connected to all nodes in the next layer. Recently, various feature extraction methods such as EfficientNet have been proposed using CNN, and a method of improving recognition and detection rate while using feature maps of various sizes such as M2Det has been proposed [25,26]. In this study, it was used to generate an image in a generator in combination with GAN and to recognize an image in a discriminator.

2.2 Generative Adversarial Networks

GAN [9] is an unsupervised learning model proposed by Ian Goodfellow in 2014 and is constructed using a Generator and Discriminator. The generator uses the randomly generated noise to generate the fake image most similar to the original image. The generated fake image is determined as a real/fake image by comparing it with the original image in the discriminator. Therefore, the generator aims to determine the generated fake image as a real image in the discriminator. On the other hand, the Discriminator aims to discriminate between Fake images and Original images generated by Generator [27,28]. Therefore, as learning is repeated, the Fake image generated by the Generator becomes more similar to the original image, and the Discriminator discriminates the Fake image generated by the Generator better. Eq. (1) is the loss function of GAN.

$$min_G max_D V(D, G) = E_{x \sim P_{data}(x)}[\log D(x)] + E_{z \sim P_z(z)}[\log(1 - D(G(z)))] \quad (1)$$

where E means the original data set, x refers to the data selected by the probability distribution, $P_{data}(x)$, of the original data, and z refers to the noise selected by the Gaussian distribution, $P_z(z)$. where $x \sim P_{data}(x)$ are probability distributions, and x is sampling data of the original data. $z \sim P_z(z)$ is an arbitrary noise distribution generated through the Gaussian distribution, and z is sampled from an arbitrary noise distribution. z is generated as a Fake image through Generator G and discriminated in Discriminator D. When the discriminator discriminates, $\log(1 - D(G(z)))$ so $D(G(z))$ is 1, and when it is discriminated, $D(G(z))$ is 0, so $\log(1 - D(G(z)))$ is close to 0. Therefore, Generator and Discriminator perform learning in the direction that V(D, G) minimizes and maximizes, respectively. Fig. 2 shows the structure of the GAN.

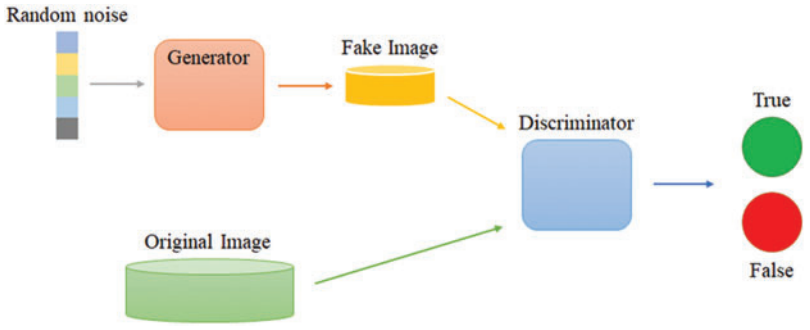


Figure 2: Generative adversarial network structure

The random noise expressed in Fig. 2 is expressed as z in Eq. (1), and it means a value randomly sampled from a Uniform Distribution or Normal Distribution. The random noise generated in this

way is generated as a fake image in the generator, and the discriminator is determined as true if it is 1 and true if it is 0. Recently, various derivative models have been studied based on GAN. StarGAN [13], which performs style transfer by performing multi-domain mapping as a model, and BigGAN [14], which uses a deep network to create high-resolution images. In this study, it was used as the basic structure of DCGAN in combination with CNN.

2.3 Deep Convolutional Generative Adversarial Network

Since the existing GAN model uses the FC layer to generate and output an image having a low resolution, it has a drawback that an image having a high resolution is not output correctly. DCGAN [10] applies the Deconvolution of Deep Convolutional Neural Network (DCNN) [29] to the GAN model in order to solve such a defect. Fig. 3 shows the generator structure of DCGAN.

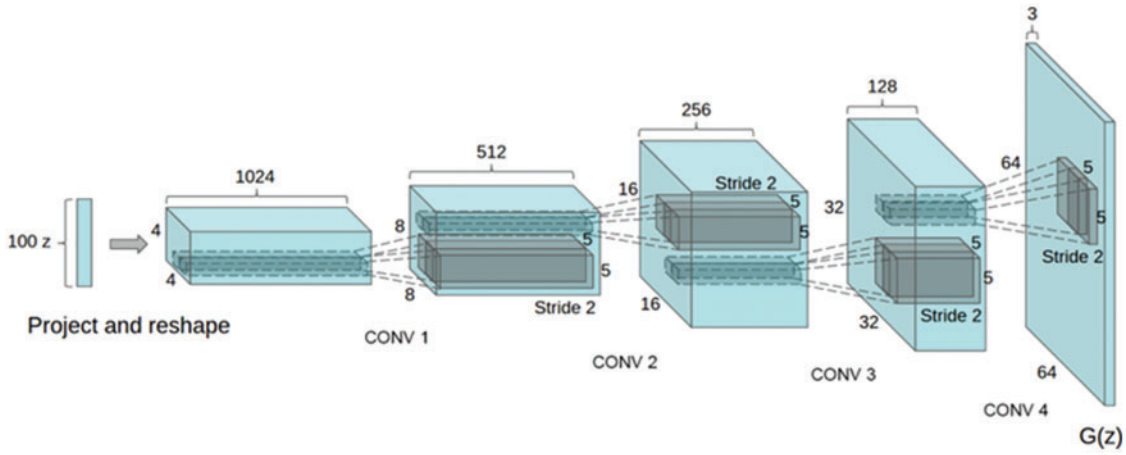


Figure 3: DCGAN generator structure

The FC layer was deleted from the deep structure of the conventional GAN model, and Batch-Normalization was used in all layers except the Generator Output Layer and Discriminator Input Layer. Also, we have replaced the non-differentiated parts of max pooling with convolution. The activation function used in the Generator is Rectifier Linear Unit (RELU), and the Hyperbolic tangent function (Tanh) is used in the last layer [30]. On the other hand, Discriminator uses a Leaky Rectified Linear Unit (Leaky ReLU) in all layers [31]. Recently, various models utilizing DCGAN have been studied. It is used in various fields such as black and white sketches and automatic color painting models of images and hyperspectral image retrieval method [32–34].

3 Proposed Method

The method proposed in this paper was studied to solve the disadvantages of feature loss and long learning time. In an image, the contour is the most important characteristic to distinguish objects. However, this contour is very lossy in CNN models, especially in deeply constructed networks. In the existing GAN-based model, a large contour loss occurs because a network is constructed based on CNN and an image is generated by receiving a random value as an input. In addition, there is a disadvantage in that it takes a long time to learn because many learning iterations are required to create an object in the image. In this paper, we propose a parallel GAN model to improve the contour loss and the long learning time. In the GAN model configured in parallel, two networks, the GAN for

general learning and the GAN for learning contours that are easily lost in training, were configured in parallel and used for learning. Fig. 4 shows the structure of the proposed method.

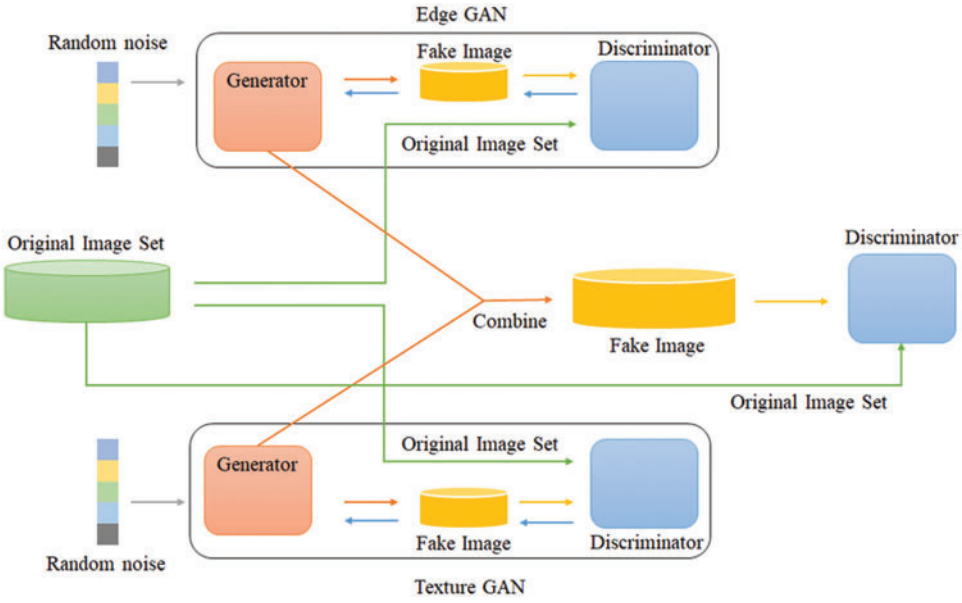


Figure 4: Structure of the multi generator and discriminator model

3.1 Edge Generative Adversarial Network

Edge Generative Adversarial Network (E-GAN) is a network constructed to learn contour. Like the existing GAN, a single generator and discriminator are arranged to form a network. The image set used for network training was constructed using the canny edge algorithm. The canny edge algorithm is sensitive to image noise, so even if it is a small noise, it is considered as the contour of the object and extracted. Therefore, before extracting the contour, the noise of the image was removed using a Gaussian filter, and then the contour was extracted. Fig. 5 shows the result of extracting the contour of the image using the canny edge algorithm.

Figure 5: Edge image set extracted using the Canny edge algorithm: (a) Original image set and (b) Edge image

As shown in Fig. 5a, the contour was extracted using the Canny edge algorithm and the contour was extracted from the original image set received as an input, and the extracted result is shown in Fig. 5b. The extracted contour images were used as training data sets in E-GAN. In E-GAN learning, the generator generates a fake image from random noise received as an input to create an image including contour information as shown in Fig. 5b. The discriminator of E-GAN determines whether the image generated by the generator is fake or true as shown in the figure. Therefore, the image generated through E-GAN generates an image including contour information without loss from the image set input to T-E GAN.

3.2 Texture Generative Adversarial Network

The Texture Generator Adversarial Network (T-GAN) is a network constructed to learn textures from images. Like the existing GAN and E-GAN, the network is composed of one generator and one discriminator. The image set used in network learning was constructed by excluding the contour from the original image input by T-E GAN. (a) of Fig. 6 is the original image input from T-E GAN, and (b) is the texture image with only texture information remaining excluding contour information.

Figure 6: Texture image set excluding edge information: (a) Original image set and (b) Texture image

In the training of T-GAN, a fake image is generated using random noise input from the generator, and the generated image creates an image with texture information. On the other hand, Discriminator uses the image set received as input from TE-GAN and the texture image with contour information excluded to determine whether the image generated by the generator is a fake image or a true image. Therefore, the image generated through the T-GAN is generated as an image including more texture information from the image set input to the T-E GAN.

3.3 Combine Discriminator and Loss Function

The fake image, used in the combine discriminator, was created by combining each noise generated through the edge GAN and the texture GAN. The combine discriminator distinguishes between the generated fake image and original image set. The loss that occurs during the training of the combine discriminator appears to be transmitted to the edge GAN and texture GAN so that it affects the learning process in each network. Learning is the process of optimizing the network parameters. The loss function in the learning process is a function that calculates the loss in the network. To minimize

these loss functions, the network parameters are modified to minimize the loss function results. Eq. (2) is a loss function used in the edge GAN.

$$L_{adv}^{C} = E_{x^c \sim C_{data}(x^c)}[\log D^c(x^c)] + E_{z \sim P_z(z)}[\log(1 - D^c(G^c(z)))] \tag{2}$$

where $C_{data}(\cdot)$ represents a dataset in which the edge is extracted using the Canny edge algorithm, and x^c the edge image. z indicates the noise generated according to the Gaussian distribution that is used in the generators of the edge GAN and texture GAN. Eq. (3) is the loss function used in the texture GAN.

$$L_{adv}^{T} = E_{x^t \sim T_{data}(x^t)}[\log D^t(x^t)] + E_{z \sim P_z(z)}\left[\log\left(1 - D^t\left(G^t(z)\right)\right)\right] \tag{3}$$

where $T_{data}(\cdot)$ refers to a dataset excluding the edge information obtained via the Canny edge algorithm, and x^t a texture image. $D^t(\cdot)$, and $G^t(\cdot)$ refer to the discriminator and generator used in the texture GAN, respectively. Eq. (4) is a loss function used for learning the original image of the combine discriminator.

$$L_{adv}^{O} = E_{x^o \sim O_{data}(x^o)}[\log D^o(x^o)] + E_{z \sim P_z(z)}\left[\log\left(1 - D^o\left(G^c(z) + G^t(z)\right)\right)\right] \tag{4}$$

where $O_{data}(\cdot)$ represents an original image set and x^o an original image. $D^o(\cdot)$ represents the combine discriminator, and $G^c(\cdot)$ and $G^t(\cdot)$ represent the generators used in the edge GAN and texture GAN, respectively. In Eq. (5), L is the final loss function.

$$L = L_{adv}^{O} + \lambda_C L_{adv}^{C} + \lambda_T L_{adv}^{T} \tag{5}$$

where λ_C and λ_T represent the learning parameters that set the influence of each edge image and texture image when performing learning from the edge GAN and texture GAN.

4 Experiments and Results

The system environment used in the experiment of the T-E GAN is as follows. The CPU was an Intel i7-8700 with 16 GB RAM, the GPU was an NVIDIA GeForce RTX 2060 with 6 GB RAM, and the Operating System used was Windows 10. The dataset used for learning was the 202,599 images of the Large-Scale CelebFaces Attributes (CelebA dataset) [18]. In this study, we trained both the DCGAN and T-E GAN and compared the experimental results. The optimizer used for learning was the Adam optimizer [34]; the learning rate of the optimizer was 0.0001, β_1 was 0.5, and β_2 was 0.999. The λ_C and λ_T parameters used in the loss function were both 0.001. The following images were the result of the learning epochs. Due to the characteristics of DCGAN, the output image is random. Fig. 7 shows images created by performing the learning iterations 3,000 times. Noise was generated in both the output images generated in the DCGAN and proposed method. However, in the output images of the proposed method, the characteristic points of the eyebrows were better expressed.

Fig. 8 shows the images created by performing the learning iterations 6,000 times. The noise was improved, and the facial expressions are better than in the 3,000 learning iterations experiment. In particular, the output images of the proposed method exhibit better edge and texture information than the output images of the DCGAN.

Fig. 9 shows the images created by performing the learning iterations 18,000 times. It was confirmed that noise was still generated with the output images in the DCGAN. Noise generation was improved in the case of the images output from the T-E GAN, along with the texture information.

(a) (b)

Figure 7: Comparison of results of (a) DCGAN with that of the (b) proposed method (iterations: 3,000 times)

(a) (b)

Figure 8: Comparison of results of (a) DCGAN with that of the (b) proposed method (iterations: 6,000 times): (a) DCGAN and (b) proposed method

(a) (b)

Figure 9: Comparison of results of (a) DCGAN with that of the (b) proposed method (iterations: 18,000 times)

Fig. 10 shows the images created by performing the learning iterations 52,000 times. For both the DCGAN and proposed method, all face expressions were naturally expressed, and the feature expressions such as eyebrows and lips were well defined. However, for the DCGAN, it was confirmed that the noise was being generated continuously, whereas almost no noise was generated by the T-E GAN.

(a) (b)

Figure 10: Comparison of results of (a) DCGAN with that of the (b) proposed method (iterations: 52,000 times)

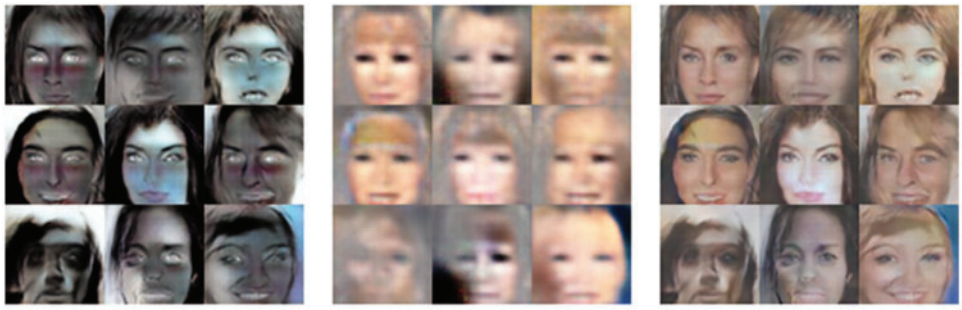
Fig. 11 shows the images created by performing the learning iterations 79,000 times. In the case of the images output from the DCGAN, the noise was still continuously generated, and there was some distortion. For the proposed method, the generated noise in the output image was minimized while some limited distortion occurred. It was confirmed that the feature points were expressed better in the proposed method than in the DCGAN.

(a) (b)

Figure 11: Comparison of results of (a) DCGAN with that of the (b) proposed method (iterations: 79,000 times)

Figs. 12a and 12b show the images created using the edge GAN and texture GAN, respectively. Each image contained edge and texture information, and the two images were then combined

and delivered to the combine discriminator. Fig. 12c is the final image output from the combine discriminator and contains both edge and texture information.



(a) (b) (c)

Figure 12: Images created from the model components: (a) Edge generator, (b) Texture generator, and (c) Combine discriminator

Fig. 13 shows a change in loss value per learning iterations of the discriminator and generator in the DCGAN model. It can be observed that as the number of learning iterations increases, the loss values of the discriminator and generator continue to exhibit significant differences. As the number of learning iterations increased, the noise and distortion of the output images were improved, but the resolution of the output images did not converge to a constant resolution.

Figure 13: Loss graphs for the (a) discriminator and (b) generator of the DCGAN

Fig. 14 is a loss graph for the generator and discriminator of the T-E GAN. Unlike the loss graph for the DCGAN, the loss became fixed at approximately 50,000 learning iterations. The discriminator demonstrated a constant loss value close to approximately 0.5 from approximately 50,000 iterations, and the generator demonstrated a constant loss value close to approximately 2.3. Therefore, the noise and distortion from approximately 50,000 iterations with a certain loss value were improved, and the images that contained the edge and texture information of the object exhibited improvement.

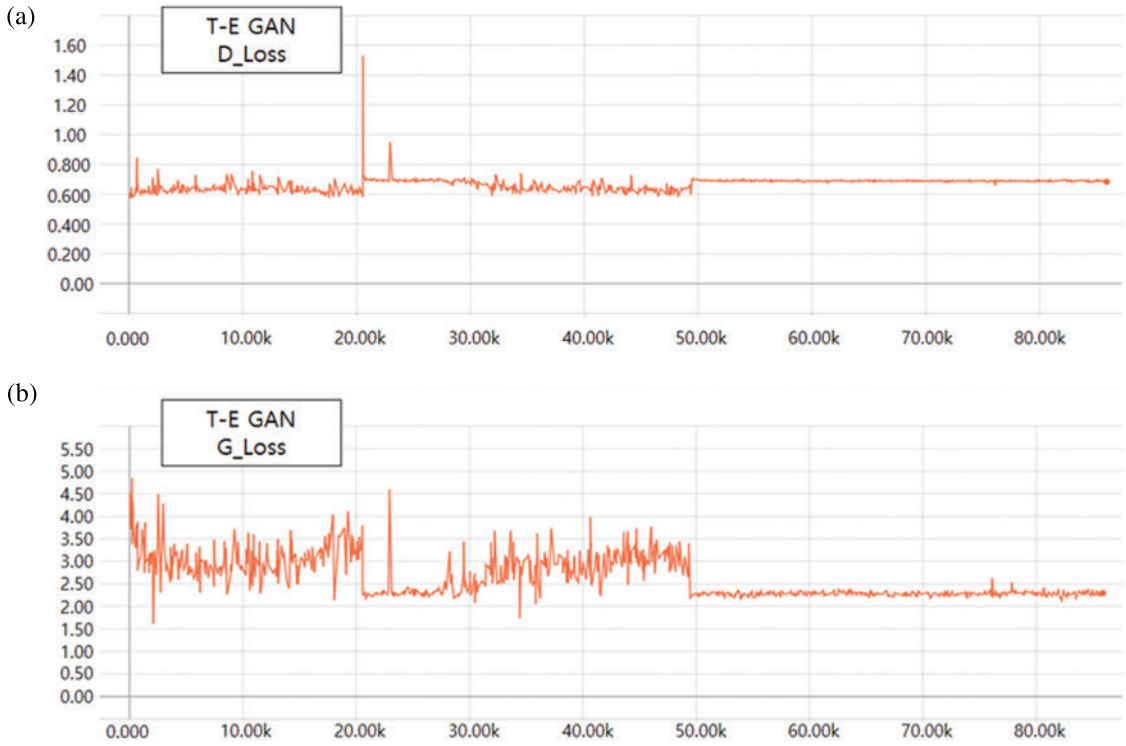


Figure 14: Loss graphs for the (a) discriminator and (b) generator of the proposed method

Table 1 shows the loss values for the proposed multi generator and discriminator model as well as the DCGAN for specific numbers of iterations. The variation of the loss value of the proposed method was smaller than that of the DCGAN.

Table 1: Comparison of loss between the multi generator and discriminator and the DCGAN

	DCGAN		Multi generator and discriminator	
Iterations	D_loss	G_loss	D_loss	G_loss
1,000	0.7248	7.228	0.6078	3.125
3,000	0.4786	3.659	0.6187	3.469
6,000	0.8315	1.143	0.6557	2.537
9,000	0.4966	4.729	0.6373	2.768
12,000	0.2411	3.332	0.6384	2.708
15,000	0.6669	4.318	0.6188	3.059

(Continued)

Table 1: Continued

	DCGAN		Multi generator and discriminator	
18,000	0.3045	3.025	0.6234	2.883
21,000	0.4708	4.463	0.6930	2.283
24,000	0.2504	2.008	0.6920	2.313
27,000	0.1328	5.201	0.7009	2.351
30,000	0.1260	4.847	0.6842	2.514
33,000	0.3767	4.806	0.6419	2.532
36,000	0.2571	5.904	0.6435	3.625
39,000	0.2121	4.3613	0.6465	2.782
42,000	0.6667	5.668	0.6191	2.629
45,000	0.03606	6.162	0.6388	2.873
48,000	0.1204	2.490	0.6277	3.163
51,000	0.1007	4.386	0.6930	2.238
54,000	0.06781	5.226	0.6967	2.268
57,000	0.3492	5.898	0.6897	2.225
60,000	0.01808	4.987	0.6875	2.305
63,000	0.05610	5.237	0.6896	2.351
66,000	0.08979	6.067	0.6939	2.295
69,000	0.1153	5.207	0.6870	2.320
72,000	0.1241	4.743	0.6861	2.275
75,000	0.1465	5.356	0.6911	2.295
78,000	0.1303	6.109	0.6886	2.291
79,000	0.05050	7.340	0.6921	2.343

Fig. 15 presents the information from Table 1 plotted graphically. The DCGAN model graph is represented by a solid line. The graph indicated by the triangle is the loss of the generator, and the graph indicated by the circle is the loss of the discriminator. The T-E GAN is indicated by a dotted line, the graph indicated by the square is the loss of the generator, and the graph indicated by the rhombus is the discriminator. The graphs confirm that the loss values of the DCGAN model fluctuated regardless of the number of learning iterations, while those of the proposed method converged to constant values at approximately 50,000 iterations.

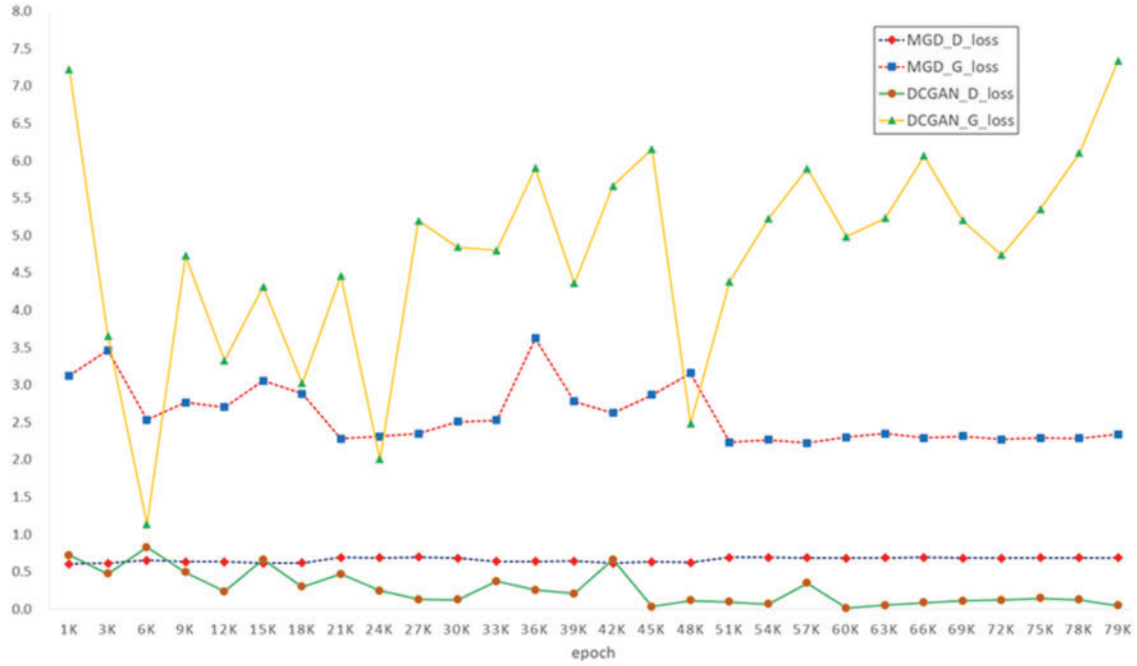


Figure 15: Loss plot of the proposed method and the DCGAN for different numbers of learning iterations

5 Conclusion

The T-E GAN consists of a network using edge information and texture information and a combined discriminator. Texture Generator and Edge Generator created images with texture information and edge information through competitive learning with each discriminator. In addition, the generated image was combined and passed to the Combine Discriminator to perform discrimination with the Original Image Set. In the case of the generator in GAN, since similar results to the original image should be output, it is said that the closer G_loss is to 1, the higher the similarity. The image generated by Generator and Discriminator had less loss of edge information and texture information than the image generated by the existing DCGAN model. In addition, it showed a loss graph with a constant size from about 50,000 training iterations, which is less than the existing DCGAN. In the case of the existing DCGAN, D_loss and G_loss are not close to 1 and 0.5, respectively, even in the case of about 80,000 times, but the proposed model shows improved D_loss and G_loss compared to DCGAN. In addition, in the case of the output image of the existing DCGAN model, as the learning iteration increased, the noise decreased and the object shape of the image became natural. The output of noise was continuously present. In the case of the proposed Multi Generator and Discriminator model, the object of the image was natural with reduced noise output even with relatively few training iterations. From about 3000 learning iterations, facial feature points were expressed, and from 6000 learning iterations, edge information and texture information were well expressed. In addition, a loss graph of a constant size was shown from a small training iteration of about 50,000 times, and an image with a constant resolution was output. Through the Multi Generator and Discriminator, a high-resolution image containing edge information and texture information was output even in fewer learning iterations.

The T-E GAN proposed in this paper still takes a lot of learning time, as the number of learning iterations reaches about 50,000. Also, image distortion due to training could not be completely eliminated. In the future, it will be necessary to study the model structure and loss function to improve the distortion of the image generated through T-E GAN and to generate an image with at least an improved learning iteration. In addition, objective model evaluation is required by developing an evaluation index that can objectively evaluate the GAN model.

Acknowledgement: This research was supported by the Mid-Career Researcher program through the National Research Foundation of Korea (NRF) funded by the MSIT (Ministry of Science and ICT) under Grant 2020R1A2C2014336.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] J. Chen, Z. Zhou, Z. Pan and C. Yang, "Instance retrieval using region of interest based cnn features," *Journal of New Media*, vol. 1, no. 2, pp. 87–99, 2019.

[2] B. I. Epureanu, L. X. Nassehi and A. Y. Koren, "Self-repair of smart manufacturing systems by deep reinforcement learning," *CIRP Annals*, vol. 69, no. 1, pp. 421–424, 2022.

[3] A. B. Adege, H. P. Lin, G. B. Tarekegn and S. S. Jeng, "Applying deep neural network (DNN) for robust indoor localization in multi-building environment," *Applied Sciences*, vol. 8, no. 7, pp. 1062, 2018.

[4] G. P. Joshi, S. Jha, S. Cho, C. Seo, L. H. Son *et al.,* "Influence of multimedia and seating location in academic engagement and grade performance of students," *Computer Applications in Engineering Education*, vol. 28, no. 2, pp. 268–281, 2020.

[5] M. M. Abuqadumah, M. A. Ali and R. R. Al-Nima, "Personal authentication application using deep learning neural network," in *2020 16th IEEE Int. Colloquium on Signal Processing & Its Applications (CSPA)*, Langkawi Island, Malaysia, pp. 186–190, 2020.

[6] Y. M. Kwon, J. J. An, M. J. Lim, S. Cho and W. M. Gal, "Malware classification using simhash encoding and PCA (MCSP)," *Symmetry*, vol. 12, no. 5, pp. 830, 2020.

[7] H. P. Chan, R. K. Samala, L. M. Hadjiiski and C. Zhou, "Deep learning in medical image analysis," *Deep Learning in Medical Image Analysis*, vol. 19, pp. 221–248, 2017.

[8] P. R. Gankidi and J. Thangavelautham, "FPGA architecture for deep learning and its application to planetary robotics," in *Proc. In 2017 IEEE Aerospace Conf.*, Montana, USA, pp. 1–9, 2017.

[9] S. Cho, B. Shrestha, W. Jang and C. Seo, "Trajectory tracking optimization of mobile robot using artificial immune system," *Multimedia Tools and Applications*, vol. 78, no. 3, pp. 3203–3220, 2019.

[10] A. Maier, C. Syben, T. Lasser and C. Riess, "A gentle introduction to deep learning in medical image processing," *Zeitschrift für Medizinische Physik*, vol. 29, no. 2, pp. 86–101, 2019.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley *et al.,* "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, Quebec, Canada, pp. 2672–2680, 2014.

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

[13] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang *et al.,* "Least squares generative adversarial networks," in *Proc. of the IEEE Int. Conf. on Computer Vision*, Venice, Italy, pp. 2794–2802, 2017.

[14] J. Y. Zhu, T. Park, P. Isola and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. of the IEEE Int. Conf. on Computer Vision*, Venice, Italy, pp. 2223–2232, 2017.

[15] Y. Choi, M. Choi, M. Kim, J. W. Ha, S. Kim *et al.,* "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, pp. 8789–8797, 2018.
[16] A. Brock, J. Donahue and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint*, 2018.
[17] J. Kim, T. Kim, S. Kim and C. D. Yoo, "Edge-labeling graph neural network for few-shot learning," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, California. USA, pp. 11–20, 2019.
[18] K. Gajowniczek, Y. Liang, T. Friedman, T. Ząbkowski and G. V. D. Broeck, "Semantic and generalized entropy loss functions for semi-supervised deep learning," *Entropy*, vol. 22, no. 3, pp. 334, 2020.
[19] Q. Wu, Y. Chen and J. Meng, "DCGAN-Based data augmentation for tomato leaf disease identification," *IEEE Access*, vol. 8, pp. 98716–98728, 2020.
[20] Z. Liu, P. Luo, X. Wang and X. Tang, "Deep learning face attributes in the wild," in *Proc. of the IEEE Int. Conf. on Computer Vision*, Santiago, Chile, pp. 3730–3738, 2015.
[21] J. Yang, T. Li, G. Liang, W. He and Y. Zhao, "A simple recurrent unit model based intrusion detection system with dcgan," *IEEE Access*, vol. 7, pp. 83286–83296, 2019.
[22] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[23] A. K. Jain, J. Mao and K. M. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 29, no. 3, pp. 31–44, 1996.
[24] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, Nevada, USA, pp. 1097–1105, 2012.
[25] H. Wu and X. Gu, "Max-pooling dropout for regularization of convolutional neural networks," in *Proc. In Int. Conf. on Neural Information Processing*, Istanbul, Turkey, pp. 46–54, 2015.
[26] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *arXiv preprint*, 2019.
[27] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen *et al.,* "M2det: A single-shot object detector based on multi-level feature pyramid network," in *Proc. of the AAAI Conf. on Artificial Intelligence*, Honolulu, Hawaii, USA33, pp. 9259–9266, 2019.
[28] T. Karras, T. Aila, S. Laine and J. Lehtinen, "Progressive growing of gans for improved quality, stability and variation," *arXiv preprint*, 2017.
[29] C. Y. Park, Y. S. Choi and K. J. Lee, "Evaluation of sentimental texts automatically generated by a generative adversarial network," *KIPS Transactions on Software and Data Engineering*, vol. 8, no. 6, pp. 257–264, 2019.
[30] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, Haifa, Israel, pp. 807, 2010.
[31] J. T. Springenberg, A. Dosovitskiy, T. Brox and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint*, 2014.
[32] P. L. Suárez, A. D. Sappa and B. X. Vintimilla, "Infrared image colorization based on a triplet dcgan architecture," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, Honolulu, Hawaii, USA, pp. 18–23, 2017.
[33] H. Heo and Y. Hwang, "Automatic sketch colorization using DCGAN," in *Proc. of 2018 18th Int. Conf. on Control, Automation and Systems (ICCAS)*, PyeongChang, Korea, pp. 1316–1318, 2018.
[34] J. Zhang, L. Chen, L. Zhuo, X. Liang and J. Li, "An efficient hyperspectral image retrieval method: Deep spectral-spatial feature extraction with DCGAN and dimensionality reduction using t-SNE-based NM hashing," *Remote Sensing*, vol. 10, no. 2, pp. 271, 2018.