IMPLEMENTATION OF DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS (DCGAN) FOR DATA AUGMENTATION IN RICE LEAF DISEASE DETECTION

Winda Asmarawati[1] , Kania Evita Dewi[2]

[1,2] Informatics Engineering Study Program, Universitas Komputer Indonesia
Jl. Dipati Ukur No.112-116, Lebakgede, Coblong District, Bandung City, West Java 40132
E-mail : kania.evita.dewi@email.unikom.ac.id[2]

Abstract

This study evaluates the effect of data augmentation using Deep Convolutional Generative Adversarial Networks (DCGAN) in detecting leaf diseases in rice (Oryza sativa L.), such as bacterial leaf blight, brown spot, and leaf smut, with a limited dataset consisting of 40 images per disease. Pre-processing was done by resizing the images to 128x128 pixels using bilinear interpolation, as well as normalizing the pixels to a range of -1 to 1, as required by the DCGAN network that uses tanh activation. Data augmentation was performed by increasing the number of datasets 1 to 4 times the original dataset, and the results were evaluated using Fréchet Inception Distance (FID). The best results were obtained when augmentation was performed 3 times, resulting in 360 new images with an FID score of around 300, indicating that the synthesized images were still less similar to the original dataset. The transfer learning-based GoogLeNet classification model trained with this augmented dataset achieved 93% accuracy in detecting rice leaf diseases. This research proves that data augmentation using DCGAN can improve detection accuracy, although the quality of the synthesized images still needs to be improved.

Keywords: *Rice leaf disease, Data augmentation, DCGAN, Transfer learning, Fréchet Inception Distance*

1. INTRODUCTION

Rice (Oryza sativa L.) is one of the main food commodities in Indonesia and an essential source of carbohydrates for millions of people in Asia. Apart from being a staple food, rice also has a high economic value and affects food stability in many developing countries. However, rice production often faces serious challenges due to the attack of various diseases, especially on the leaves, such as Bacterial leaf blight, Brown spot, and Leaf smut. These diseases not only cause a significant reduction in the quality, but also the quantity of the crop if they are not immediately detected and treated with appropriate methods [1].

In recent years, technological advances in the field of Computer Vision, especially those based on Convolutional Neural Networks (CNN), have opened up new opportunities in automatic detection of rice leaf diseases. This technology enables faster and more accurate early detection, allowing farmers to take preventive action before the disease becomes widespread [2], [3]. However, the successful implementation of CNN is highly dependent on the availability of sufficiently balanced and diverse datasets. Limited and unrepresentative datasets are often an obstacle, making it difficult for CNN models to generalize when applied to new data or different field conditions [4], [5].

Traditional data augmentation methods such as rotation, cropping, and image brightness change are often used to increase the number and diversity of datasets. Data augmentation has also been proven effective in improving CNN performance on Sundanese script handwriting recognition [6] and music notation images [7]. However, these methods have limitations in adding enough realistic variety and often fail to capture the entire spectrum of rice leaf disease symptoms that may occur in the field. This causes models trained with traditionally augmented datasets to still be suboptimal in recognizing more complex disease patterns [8], [9].

To overcome the limitations of traditional data augmentation, Deep Convolutional Generative Adversarial Networks (DCGAN) method has been introduced as an effective solution. DCGAN is capable of generating realistic and varied synthetic images, increasing the diversity of the dataset significantly, especially on small or limited datasets [10], [11]. Wu et al. [3] showed that DCGAN can improve the

accuracy of tomato leaf disease detection models, while Talukdar [5] found that DCGAN is effective in handling class imbalance in plant disease classification. In addition, Zhu et al. [13] revealed that improved DCGAN can reduce overfitting, and Patmawati et al. [14] demonstrated the benefits of DCGAN in soil image data augmentation. Therefore, the application of DCGAN in rice leaf disease detection can be further explored.

This study aims to evaluate the use of DCGAN in data augmentation to improve the accuracy of rice leaf disease classification, especially on limited datasets. Tests were conducted by measuring the quality of synthetic images using Frechet Inception Distance (FID) [12] and applying transfer learning for classification, in line with the findings of Wu et al. [3] who showed improved health detection accuracy through the combination of DCGAN and transfer learning. The results of this study are expected to strengthen the effectiveness of DCGAN as a data augmentation tool, improve the accuracy and generalizability of the model, and make a positive contribution to rice plant disease management in the agricultural sector.

2. METHODOLOGY

The following is an explanation of the research methodology that will be applied in this study as shown in Figure 1.

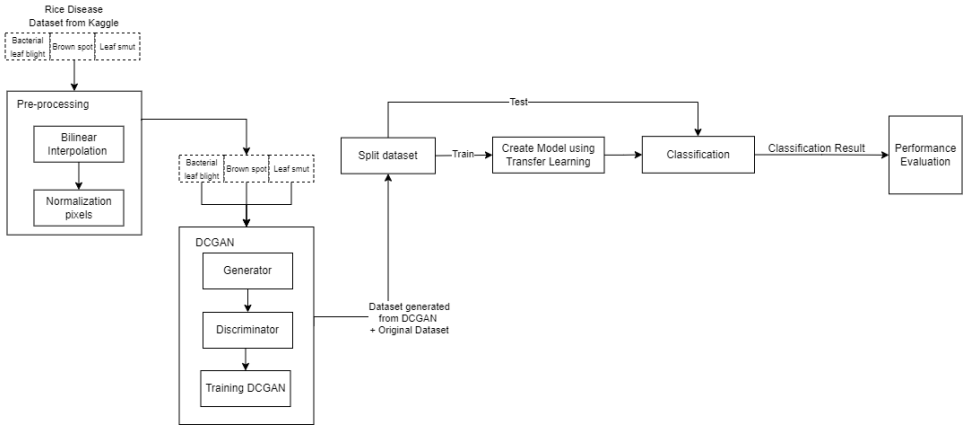

Figure 1. Research Methodology

2.1 Dataset

The dataset used is taken from the Kaggle vbookshelf source containing 120 images that are grouped into 3 types of diseases on rice leaves, namely: Bacterial Leaf Blight, Brown Spot, and Leaf Smut [15]. Each type of disease has the same number of images, namely 40 images.

2.2 Pre-processing

Pre-processing is the initial stage in data processing that aims to prepare the dataset before it is used in model training. In this research, there are two stages of pre-processing:

2.2.1 Bilinear Interpolation

Bilinear interpolation is an image scaling method used to ensure the image has a consistent resolution, in this case 128x128 pixels, before being incorporated into the DCGAN model. The technique works by using the four nearest neighbor pixels to estimate the value of new pixels in the resized image. The process involves scaling in two directions, horizontal and vertical, described by equation 1.

$$I' = S(I, r_x, r_y) = S_y(S_x(I, r_x), r_y) = S_x(S_y(I, r_y), r_x) \quad (1)$$

With S as the scaling function, and $r_x$and $r_y$as the ratio of horizontal and vertical scaling [16]. The pixel value at position (Y′,X′) is calculated by blending the pixel values of the four nearest neighbors, based on their distance from the point of interest, giving a smoother result. Rounding down is often used to ensure the coordinates of neighboring pixels stay within the boundaries of the original image [17].

2.2.2 Normalization Pixels

Normalization Pixels or commonly called pixel normalization is a technique used to change the values of image pixels so that they are within a certain range, in this case from [-1, 1]. This technique is important to improve the stability and convergence speed of the DCGAN model, which uses activation functions such as tanh in the output generator layer. The following will be explained in equation 2 [18].

$$X'(i,j) = \frac{X(i,j)}{127.5} - 1.0 \tag{2}$$

Where X′(i,j) is the normalized feature layer pixel value, and X(i,j) is the original pixel value which is in the range [0, 255] [28]. In this process, the original pixel value is first divided by 127.5 to change its range from [0, 255] to [0, 2]. Next, the obtained value is subtracted by 1.0, which shifts the range to [-1, 1].

2.3 Deep Convolutional Generative Adversarial Networks (DCGAN)

Deep Convolutional Generative Adversarial Networks (DCGAN) aims to overcome the stability problem of GAN training and improve the quality of feature representation from unlabeled image data. The following Figure 2 is the architecture used in the DCGAN implementation process in data augmentation adapted from the Chest X-ray journal [19].

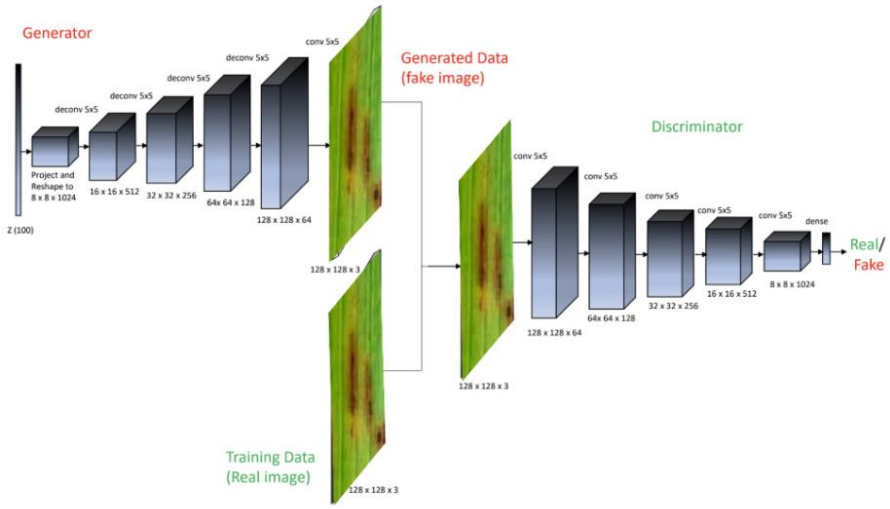

Figure 2. Architecture of Deep Convolutional Generative Adversarial Networks (DCGAN) [17]**.**

2.3.1 Generator

The generator in DCGAN starts with a low-dimensional latent vector containing random values from a normal (Gaussian) distribution with mean 0 and standard deviation 1 [10]. This vector is then projected into a 3D shape and resized through a series of transpose convolution layers followed by a ReLU or LeakyReLU activation function to add non-linearity, with the last layer using a Tanh activation function to generate pixel values in the range [-1, 1]. This structure allows the generator to transform latent vectors into realistic images similar to the original data. Full details regarding the architecture of this generator can be seen in Table 1.

Table 1. Generator Architecture

Layer Type	Output Shape	Filters	Kernel Size	Stride	Padding	Activation
Input	(100, 1) noise vector	-	-	-	-	-
Dense	8x8x1024	-	-	-	-	ReLU
Reshape	(8, 8, 1024)	-	-	-	-	-
Batch Normalization (BN)						
Conv2DTranspose + BN	(16, 16, 512)	512	5x5	2	same	ReLU
Conv2DTranspose + BN	(32, 32, 256)	256	5x5	2	same	ReLU
Conv2DTranspose + BN	(64, 64, 128)	128	5x5	2	same	ReLU
Conv2DTranspose + BN	(128, 128, 64)	64	5x5	2	same	ReLU
Output	(128, 128, 3)	3	5x5	1	same	Tanh

2.3.2 Discriminators

The discriminator in DCGAN uses a standard CNN architecture with multiple convolution layers in charge of classifying the image as real or fake. The process starts with a 128x128 pixel image that has been normalized to the range [-1, 1]. The discriminator successively reduces the spatial resolution of the data through five convolution layers, each followed by a LeakyReLU activation function to handle negative values and speed up convergence during training. And batch normalization is applied in each layer to stabilize and speed up the training process. The last layer uses a sigmoid activation function to generate the probability that the input image is real (1) or fake (0) [10]. Full details of the generator architecture can be seen in Table 2.

Table 2. Discriminator Architecture

Layer Type	Output Shape	Filters	Kernel Size	Stride	Padding	Activation
Input	128x128x3	-	-	-	-	-
Conv2D	(128, 128, 64)	-	5x5	1	same	LeakyReLU
Conv2D + Batch Normalization	(64, 64, 128)	-	5x5	2	same	LeakyReLU
Conv2D + Batch Normalization	(32, 32, 356)	512	5x5	2	same	LeakyReLU
Conv2D + Batch Normalization	(16, 16, 512)	256	5x5	2	same	LeakyReLU
Conv2D + Batch Normalization	(8, 8, 1024)	128	5x5	2	same	LeakyReLU
Output	(0, 1)	64	-	-	-	Sigmoid

2.3.2 DCGAN Training

DCGAN training involves the Generator and Discriminator being trained alternately within the framework of a minimax game. The Discriminator is trained to distinguish between real and fake images, with a loss function L_D that maximizes the Discriminator's ability to recognize real images and reject fake images. Conversely, the Generator is trained to produce fake images that the Discriminator considers genuine, by minimizing the loss function. L_G. The following can be seen in the equation below [20].

$$L_D = -(\mathbb{E}_{x\sim P_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z)))]) \quad (3)$$

Where:

$\mathbb{E}_{x\sim P_{data}(x)}[\log D(x)]$: expectation of the log probability that a real sample x is considered real by the discriminator D

$\mathbb{E}_{z\sim P_z(z)}[\log(1 - D(G(z)))]$: expectation of the log probability that a false sample G(z) is considered false by the discriminator D

$$L_G = -\mathbb{E}_{z\sim P_z(z)}[\log D(G(z))] \quad (4)$$

Where:

$\mathbb{E}_{z\sim P_z(z)}[\log D(G(z))]$: expectation of the log probability that a false sample G(z) is considered real by the discriminator D

This process is performed iteratively, with the ultimate goal of reaching a Nash equilibrium where the Generator produces images that are very similar to the original images, and the Discriminator can no longer consistently distinguish between the original and fake images. During training, performance evaluations are performed to ensure that a balance between the Generator and Discriminator is achieved, avoiding issues such as mode collapse [20].

2.4 Spliting Data

After the additional data from DCGAN was generated, the original data was manually divided into two parts, with most of it being included in the training dataset and the other part being used for the testing dataset.

a. Training dataset (train)

This dataset consists of the original data and the data generated by DCGAN. The DCGAN-generated data is directly incorporated into the training dataset without further subdividing. The proportion of images used in the training dataset ranges from 75% to 88.89% of the total available images. The more images generated from the augmentation process, the larger the proportion allocated for training.

b. Test dataset (test)

This dataset uses only the original data. It consists of 60 images from the original data in each test. This number remained constant in each test, which overall represents 11.11% up to 25% of the total image.

2.5 Transfer Learning

After the rice leaf disease synthesis dataset is obtained, the next step is to train the CNN model using transfer learning with the GoogleNet architecture (Inception V1). This architecture introduces the Inception module, which incorporates various convolution kernel sizes (1x1, 3x3, 5x5) and max-pooling layers to capture features at various scales, improving efficiency and accuracy. GoogleNet uses fewer parameters than AlexNet and relies on average pooling and dropout to prevent overfitting. With 22 layers and 9 Inception modules, GoogleNet is an effective choice for image classification [21]. The following Figure 3 is an architecture that illustrates the classification process using the GoogleNet architecture adapted from the journal [3].

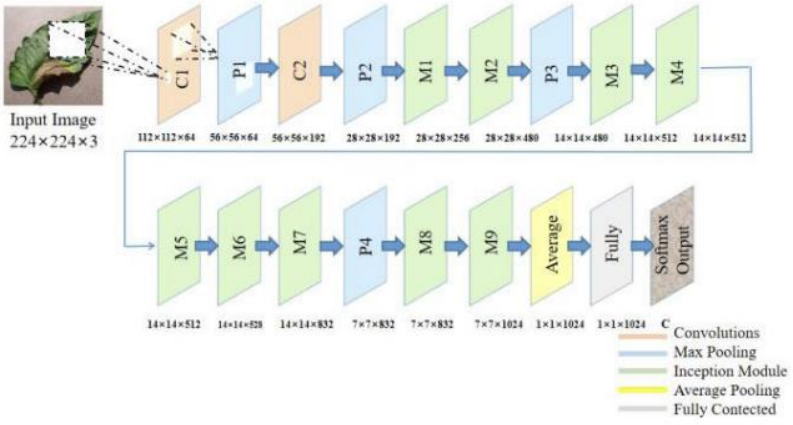

Figure 3. GoogleNet Architecture [3]

2.6 Performance Evaluation

After training, the performance of the model is evaluated using the test dataset. The evaluation is done by calculating accuracy as the main metric, which gives an idea of how effective the model is in correctly classifying rice leaf disease images. This accuracy reflects the percentage of correct predictions compared to the total number of predictions made by the model.

3. RESULTS AND DISCUSSION

3.1 Training Configuration

The training of DCGAN models for data augmentation and classification using transfer learning with GoogleNet architecture was conducted on the Google Colab platform. To speed up the training process, a TPU (Tensor Processing Unit) provided by Colab was used. Some of the libraries and frameworks used include TensorFlow for DCGAN implementation and PyTorch for GoogleNet training.

3.2 FID Testing

This test was conducted using the Fréchet Inception Distance (FID) method that utilizes the InceptionV3 pre-trained model as a feature extractor in Google Colab. This FID method is used to measure how similar the resulting image distribution is to the original image distribution, which directly describes the quality of the image synthesis. This FID test will be presented in Figure 4 to Figure 6, which displays the FID values for the various configurations and test scenarios that have been applied.

Then the descriptions for the parameters used in the figure are as follows: G is the learning rate for the generator, ranging from 0.0001 to 0.0003, setting how large the generator learning step is. D is the learning rate for the discriminator with the same range of values, but often slightly higher or equal to maintain balance. Beta 1 refers to the momentum in Adam's optimizer, usually set between 0.5 to 0.9 to reduce fluctuations and maintain GAN stability. BS denotes the batch size, with a range of 8 to 32, adjusted for memory capacity and amount of data.

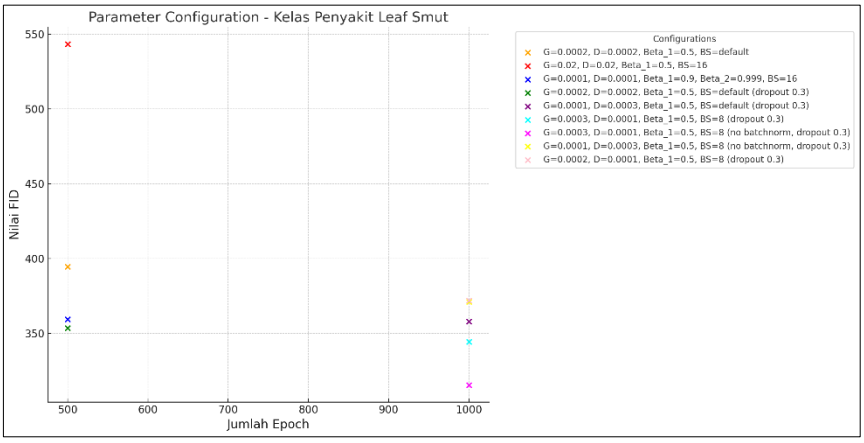

Figure 4. Performance testing of Fréchet Inception Distance (FID) for Leaf smut disease

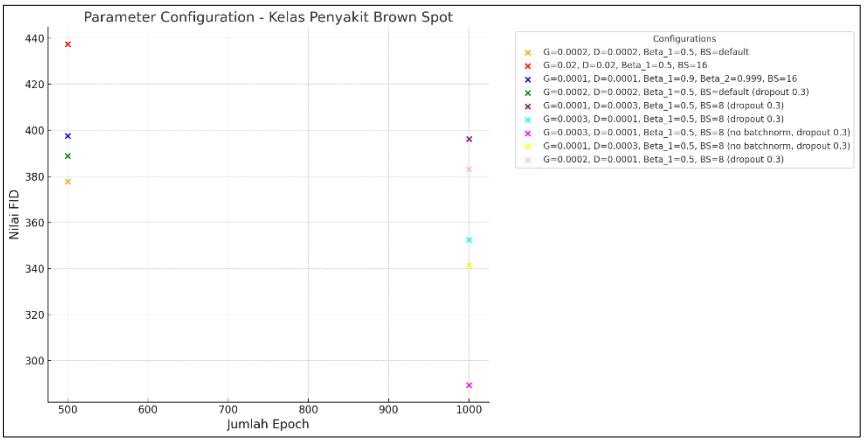

Figure 5. Brown spot disease Fréchet Inception Distance (FID) Performance Testing

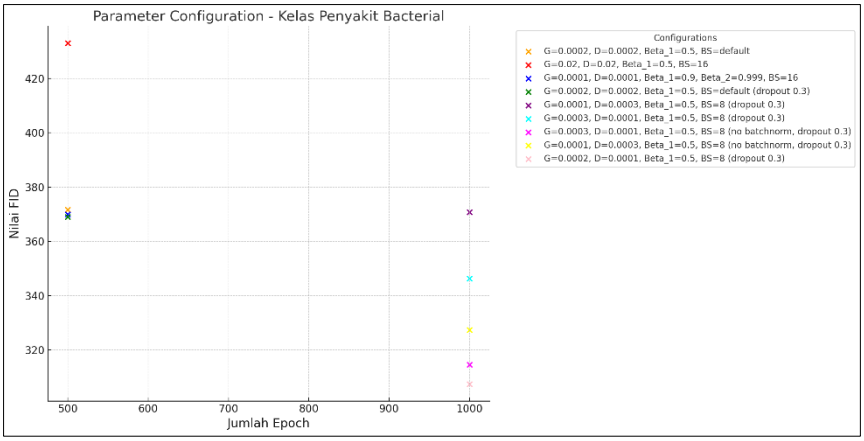

Figure 6. Bacterial leaf blight Fréchet Inception Distance (FID) Performance Testing

3.3 Classification Testing

This test was conducted using GoogleNet architecture and focused on evaluating the accuracy of the model to identify leaf diseases in rice. The model was tested with various combinations of hyperparameters, namely learning rate of 0.001, SGD optimizer, batch size of 32, and epoch of 10, using augmentation data generated through two methods, namely TensorFlow and DCGAN libraries. The scale of data augmentation used varies from 1-fold to 4-fold. The 1-fold scale (1st test) increased the number of original images to 120 images; the 2-fold scale (2nd test) increased the number of original images to 240 images; the 3-fold scale (3rd test) increased the number of original images to 360 images; and the 4-fold scale (4th test) increased the number of original images to 480 images.

3.4 FID Testing Results

FID (Fréchet Inception Distance) testing is performed to evaluate the quality of images produced by the model based on several parameter combinations. The model with the lowest FID value produces an image that is closest to the original dataset [17]. The best results were obtained using the architecture in the 7th test, namely the generator without batch normalization, the application of dropout on the discriminator, and different learning rate settings between the generator and discriminator [20]. In addition, the use of a small batch size also contributed positively, mainly due to the relatively small amount of data [21]. This test was conducted on three disease classes in the dataset, namely Leaf Smut, Brown Spot, and Bacterial Leaf Blight. The results show that this configuration provides the best performance in replicating images similar to the original data, as shown in Figure 7.

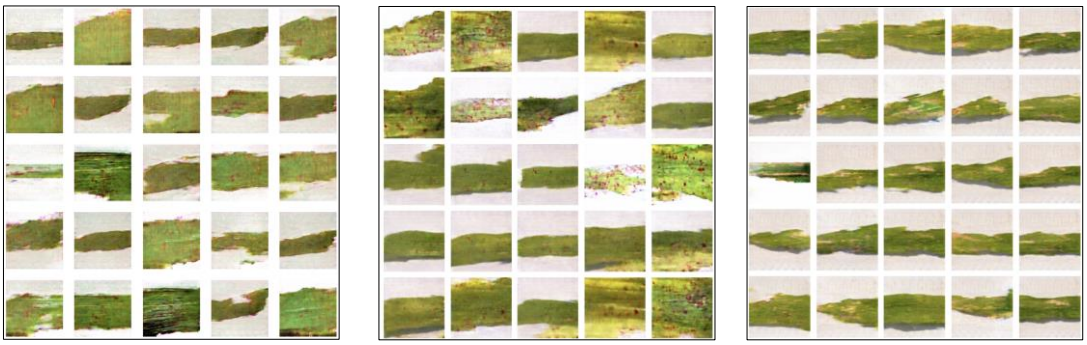

Figure 7. New Image of Leaf Smut, Brown Spot, and Bacterial Leaf Blight Disease 7th Testing

The loss graph in this best test shows that the generator loss increases gradually during the training process. This indicates that the generator model is finding it increasingly difficult to produce realistic images as the quality of the discriminator improves. Meanwhile, the discriminator loss remains relatively stable and low, indicating that the discriminator is able to effectively distinguish between the original and generator images throughout the training as shown in Figure 8.

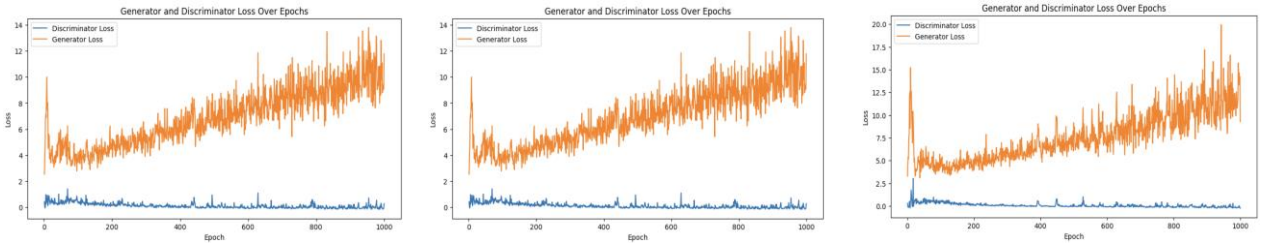

Figure 8. Graph of Loss of Leaf Smut, Brown Spot, and Bacterial Leaf Blight in the 7th Test

3.5 Classification Testing Results

This classification test aims to evaluate the performance of the model in detecting rice leaf diseases using a combination of data augmentation methods and classification techniques. Several tests were conducted with varying amounts of data generated by augmentation methods from TensorFlow and DCGAN. Each test involved a total of 30 test data images, consisting of 10 images for each disease. The best results were shown by the confusion matrix of the model using DCGAN data augmentation by generating a total of 360 new images, shown in Table 3, where the highest accuracy of 93% was achieved. This model managed to classify most of the images correctly, indicating that the use of DCGAN data augmentation significantly improved the classification performance.

Table 3. Confusion Matrix of the 3rd Test of Classification with DCGAN Augmentation

	Bacterial leaf blight	**Brown spot**	**Leaf smut**
Bacterial leaf blight	10	0	0
Brown spot	0	9	1
Leaf smut	1	0	9

3.6 Discussion

The results of the implementation of the DCGAN method on a limited dataset of 40 images per disease class, showed the ability to produce a fairly good synthesized image visually as shown in Figure 7, despite some constraints. Furthermore, as shown in Figure 4 to Figure 6 above, in the 7th test visualized in dark pink, setting the generator architecture without batch normalization, applying dropouts to the discriminator, as well as setting different learning rates between the generator and discriminator, contributed to the results obtained [22]. The use of a small batch size also made a positive contribution, especially considering the limited amount of data [23].

Ideally, in GAN training, both the generator loss and discriminator loss should decrease and approach each other, indicating that the generator is increasingly capable of producing realistic images, while the discriminator is finding it increasingly difficult to distinguish between the original and synthesized images [12], [20]. However, in this 7th test shown in Figure 8, the generator loss gradually increases, while the discriminator loss remains stable and low, indicating an imbalance in the training process.

Although the resulting synthesized images are visually quite good, the Fréchet Inception Distance (FID) score obtained is still quite high, which is around 300 for all three disease classes, indicating that the generated images do not fully resemble the original dataset [22]. This high FID score has an impact on classification accuracy, so the classification performance with DCGAN is lower than the data augmentation method using the TensorFlow library. In the augmentation method using TensorFlow, various transformations such as scale transformation, horizontal and vertical inversion, rotation, width and height shift, and image magnification are applied.

The recapitulated accuracy results for each test are presented in Figure 9, where the model shows the best performance during the 3rd classification test where the number of augmented images reaches 360 images, or three times the number of the original images.

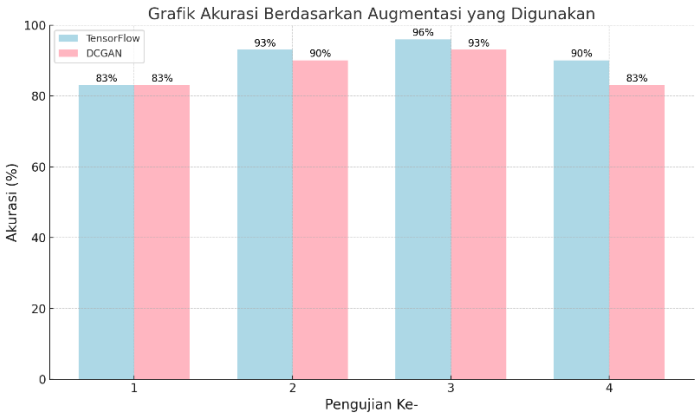

Figure 9. Recapitulation of Accuracy of Each Test

The classification test results show that the model achieved the best accuracy of 96% using the TensorFlow augmentation method, while the DCGAN method achieved an accuracy of 93%. These results indicate that the model works well under data augmentation conditions, but there is still room for improvement, especially in optimizing the FID score to improve the overall performance of the model.

4. CLOSING

This research successfully applied a data augmentation method using Deep Convolutional Generative Adversarial Networks (DCGAN) to increase the number of rice leaf disease datasets, which were then used in combination with transfer learning using GoogleNet. Although the synthesized images generated by DCGAN were visually quite good, the high Fréchet Inception Distance (FID) score indicated that the images were still not similar enough to the original dataset, which affected the classification performance. However, the combination of DCGAN augmentation and transfer learning proved to be able to increase the accuracy of rice leaf disease detection to 93%, showing significant potential despite the suboptimal quality of the synthesized images. For future research, improving the quality of the synthesized images through adjusting the DCGAN architecture and parameters as well as exploring other Generative Adversarial Networks (GAN) methods is recommended. The use of larger and more diverse datasets is also expected to improve the robustness of the model and overall detection accuracy.

LITERATURE

[1] S. A. Burhan, D. S. Minhas, D. A. Tariq, and M. Nabeel Hassan, "Comparative Study of Deep Learning Algorithms for Disease and Pest Detection in Rice Crops," in Proceedings of the 12th International Conference on Electronics, Computers and Artificial Intelligence, ECAI 2020, Institute of Electrical and Electronics Engineers Inc, Jun. 2020. doi: 10.1109/ECAI50035.2020.9223239.

[2] A. Julianto and A. Sunyoto, "A performance evaluation of convolutional neural network architecture for classification of rice leaf disease," IAES International Journal of Artificial Intelligence (IJ-AI), vol. 10, no. 4, pp. 1069-1078, Dec. 2021, doi: 10.11591/ijai.v10.i4.pp1069-1078.

[3] Q. Wu, Y. Chen, and J. Meng, "Dcgan-based data augmentation for tomato leaf disease identification," IEEE Access, vol. 8, pp. 98716-98728, 2020, doi: 10.1109/ACCESS.2020.2997001.

[4] G. Yang, G. Chen, C. Li, J. Fu, Y. Guo, and H. Liang, "Convolutional Rebalancing Network for the Classification of Large Imbalanced Rice Pest and Disease Datasets in the Field," Front Plant Sci, vol. 12, Jul. 2021, doi: 10.3389/fpls.2021.671134.

[5] B. Talukdar, "Handling of Class Imbalance for Plant Disease Classification with Variants of GANs," in 2020 IEEE 15th International Conference on Industrial and Information Systems, ICIIS 2020 - Proceedings, Institute of Electrical and Electronics Engineers Inc, Nov. 2020, pp. 466-471. doi: 10.1109/ICIIS51140.2020.9342728.

[6] I. Maliki and A. S. Prayoga, "Implementation of Convolutional Neural Network for Sundanese Script Handwriting Recognition with Data Augmentation," Journal of Engineering Science and Technology, vol. 18, no. 2, pp. 1113-1123, Apr. 2023
[7] D. M. Hakim and E. Rainarli, "Convolutional Neural Network for Music Notation Image Recognition," Techno.COM, vol. 18, no. 3, pp. 214-226, Aug. 2019, doi: 10.33633/tc.v18i3.2387
[8] J. G. A. Barbedo, "Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification," Comput Electron Agric, vol. 153, pp. 46-53, Oct. 2018, doi: 10.1016/j.compag.2018.08.013.
[9] Y. Lu, X. Tao, N. Zeng, J. Du, and R. Shang, "Enhanced CNN Classification Capability for Small Rice Disease Datasets Using Progressive WGAN-GP: Algorithms and Applications," Remote Sens (Basel), vol. 15, no. 7, Apr. 2023, doi: 10.3390/rs15071789.
[10] J. Wang and L. Perez, "The Effectiveness of Data Augmentation in Image Classification using Deep Learning," arXiv preprint arXiv:1712.04621, Dec. 2017. Available: https://arxiv.org/abs/1712.04621.
[11] L. Taylor and G. Nitschke, "Improving Deep Learning using Generic Data Augmentation," arXiv preprint arXiv:1708.06020, Aug. 2017. Available: https://arxiv.org/abs/1708.06020.
[12] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," presented at the International Conference on Learning Representations (ICLR), 2016. [Online]. Available: arXiv:1511.06434.
[13] F. Zhu, M. He, and Z. Zheng, "Data augmentation using improved cDCGAN for plant vigor rating," Computers and Electronics in Agriculture, vol. 175, p. 105603, 2020. Available: https://doi.org/10.1016/j.compag.2020.105603
[14] Patmawati, A. Sunyoto, and E. T. Luthfi, "Data Augmentation Using DCGAN on Land Images," TEKNIMEDIA, vol. 4, no. 1, pp. 45-52, Jun. 2023.
[15] Vbookshelf, "Rice Leaf Diseases," Kaggle, 2020. [Online]. Available: https://www.kaggle.com/datasets/vbookshelf/rice-leaf-diseases/data.
[16] R.-G. Zhou and C. Wan, "Quantum Image Scaling Based on Bilinear Interpolation with Decimals Scaling Ratio," International Journal of Theoretical Physics, vol. 60, no. 7, pp. 2159-2174, Jul. 2021, doi: 10.1007/s10773-021-04829-6.
[17] O. Rukundo, "Effects of Improved Floor Function on the Accuracy of Bilinear Interpolation Algorithm," Computer and Information Science, vol. 8, no. 4, pp. 1-11, 2015, doi: 10.5539/cis.v8n4p1.
[18] X. Mou, T. Zhu, and X. Zhou, "Visible-Image-Assisted Nonuniformity Correction of Infrared Images Using the GAN with SEBlock," Sensors, vol. 23, no. 6, p. 3282, Mar. 2023, doi: 10.3390/s23063282.
[19] S. Kora Venu and S. Ravula, "Evaluation of Deep Convolutional Generative Adversarial Networks for Data Augmentation of Chest X-ray Images," Future Internet, vol. 13, no. 1, p. 8, Jan. 2021, doi: 10.3390/fi13010008.
[20] I. J. Goodfellow et al., "Generative Adversarial Nets," in Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS), Montreal, QC, Canada, Dec. 2014, pp. 2672-2680.
[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Boston, MA, USA, Jun. 2015, pp. 1-9.
[22] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S., "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," arXiv preprint arXiv:1706.08500, Jan. 2018. [Online]. Available: https://arxiv.org/abs/1706.08500.
[23] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks," arXiv preprint arXiv:1812.04948v3, Mar. 2019. [Online]. Available: https://arxiv.org/abs/1812.04948v3.