A performance evaluation of convolutional neural network architecture for classification of rice leaf disease

Afis Julianto, Andi Sunyoto
Magister of Informatics Engineering, Universitas AMIKOM Yogyakarta, Depok, Sleman, Yogyakarta, Indonesia

Article Info

Article history:

Received Mar 28, 2021
Revised Aug 28, 2021
Accepted Sep 17, 2021

Keywords:

CNN architecture
Image classification
Performance evaluation
Rice leaf disease

ABSTRACT

Plant disease is a challenge in the agricultural sector, especially for rice production. Identifying diseases in rice leaves is the first step to wipe out and treat diseases to reduce crop failure. With the rapid development of the convolutional neural network (CNN), rice leaf disease can be recognized well without the help of an expert. In this research, the performance evaluation of CNN architecture will be carried out to analyze the classification of rice leaf disease images by classifying 5932 image data which are divided into 4 disease classes. The comparison of training data, validation, and testing are 60:20:20. Adam optimization with a learning rate of 0.0009 and softmax activation was used in this study. From the experimental results, the InceptionV3 and InceptionResnetV2 architectures got the best accuracy, namely 100%, ResNet50 and DenseNet201 got 99.83%, MobileNet 99.33%, and EfficientNetB3 90.14% accuracy.

Corresponding Author:

Andi Sunyoto
Magister of Informatics Engineering
Universitas Amikom Yogyakarta
Depok, Sleman, Yogyakarta 55281, Indonesia
Email: andi@amikom.ac.id

1. INTRODUCTION

Rice is one of the food commodities for most people in world. For example Indonesia, rice production in 2019 amounted to 54.60 million tonnes of milled dry unhulled with a harvest area of 10.68 million hectares [1]. However, rice plants are often exposed to the disease during their growing period. Disease in rice frequently attacks the leaves. Fungi and bacteria are believed to be the main causes of the disease [2]. Some of the diseases that often attack rice leaves are Brown Spot, Tungro, Bacterial Leaf Blight, and Blast. Rice leaf disease can result in lowly rice plant growth, resulting in decreased production.

From one study, it is estimated that the decrease in rice production due to blast disease can reach 100% [3], bacterial leaf blight 15-25% [4], and brown leaf spot 40% [5]. If not handled seriously, it will bring huge economic losses for rice farmers. Some farmers, in general, do not know enough about rice leaf diseases, especially young farmers who lack expertise in agriculture so that it is difficult to identify the type of disease. Without knowing the type of disease, it will be difficult to choose suitable drugs and great handling procedures. Therefore, it is necessary to be able to classify the types of diseases in rice leaves.

In recent years, many studies have been conducted on the classification of rice leaf disease. Research has been conducted using the k-nearest neighbor (KNN) method to classify Blast and Brown Spot disease in rice leaves obtained an accuracy of 76.59% [6]. Other studies have been conducted to detect leaf smut, bacterial leaf blight, and brown spots on rice leaves. The study conducted an experiment using the logistic regression algorithm, obtained an accuracy of 70.83%, KNN 91.67%, 97.91% decision tree (DT), and 50% naïve bayes (NB) [7]. Further research was carried out using the support vector machine (SVM) method to diagnose Brown Spot and Blast disease in rice leaves with a total data of 120, obtained an accuracy of 95.5% [8]. SVM is also used for the classification process with the addition of the k-means clustering method and obtained an accuracy of 92.06% [9]. The artificial neural network (ANN) method has been used for the classification of rice leaf disease by segmenting and feature extraction on images [10].

However, to get a good level of accuracy still depends on the feature selection technique when using this method. Recent research on convolutional neural network (CNN) has contributed greatly to image-based identification by eliminating the need for pre-processing images and having built-in feature selection [11]. Research using the CNN method has been [12] carried out to identify 10 disease classes in rice leaves and stems using 500 image data, from the test results obtained an accuracy of 95.48% using 10 Fold Cross-Validation. Other studies were also conducted using CNN architectural models such as VGG 16 with 92.46% accuracy [11], AlexNet 91.23% accuracy [13], and VGG 19 with 92% accuracy [14]. In recent years, many researchers have been developing new CNN architecture. So it is necessary to research the performance of the new CNN architectural model for the classification of rice leaf disease.

This paper will conduct a study that focuses on the classification of 4 types of rice leaf diseases, namely Bacterial Leaf Blight, Blast, Tungro, and Brown Spot using 6 types of CNN architectures, namely InceptionV3, ResNet50, InceptionResnetV2, DenseNet201, MobileNet, and EfficientNetB3. By comparing all the CNN architectures, performance evaluation will be carried out to see the best way to recognize rice leaf disease.

2. RESEARCH METHOD

This section explains the research flow regarding the classification of rice leaf disease starting from the acquisition of datasets, pre-processing data, the use of the convolutional neural network (CNN) architectural model, and evaluation of the performance of the CNN architectural model as shown in Figure 1.

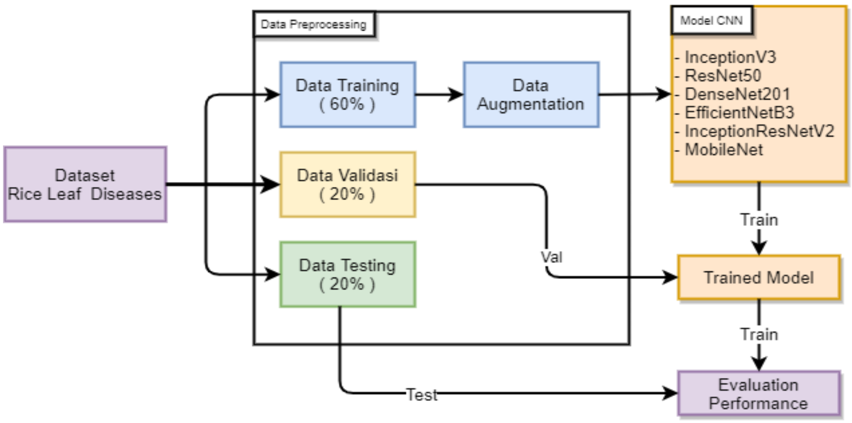

Figure 1. Research flow

2.1. Dataset

In this study, the dataset used was taken from Mendeley Data which is a collection of pictures of rice leaf disease [15]. This dataset has 5932 rice leaf disease image data consisting of 4 disease classes including Bacterial Blight, Blast, Bown Spot, and Tungro disease as shown in Figure 2. All image data obtained in this study are stored in JPG format.

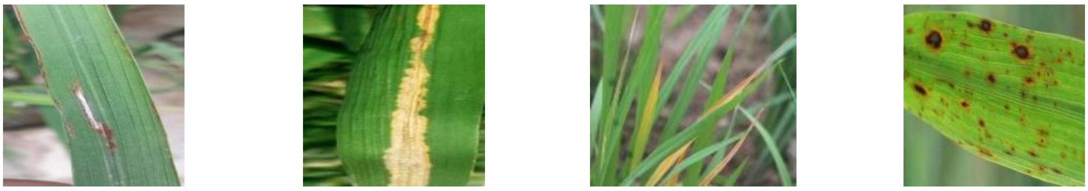
(a) (b) (c) (d)

Figure 2. Types of disease on rice leaves; (a) Blast, (b) Bacteria Blight, (c) Tungro, (d) Brown Spot

2.2. Pre-processing data

The dataset is divided into 60% training, 20% validation, and 20% testing. Then each image will be resized to 300x300 pixels. In the training data, an augmentation process will be carried out by rotating up to 40 degrees, shifting the image to a scale of 0.2, enlarging the image to a scale of 0.2, and flipping the image vertically and horizontally. So that the number of images in the training data increases 6 times including images that apply augmentation. Figure 3 shows the sample data from the augmentation results. The disease name and the data amount used in this study can be seen in Table 1.

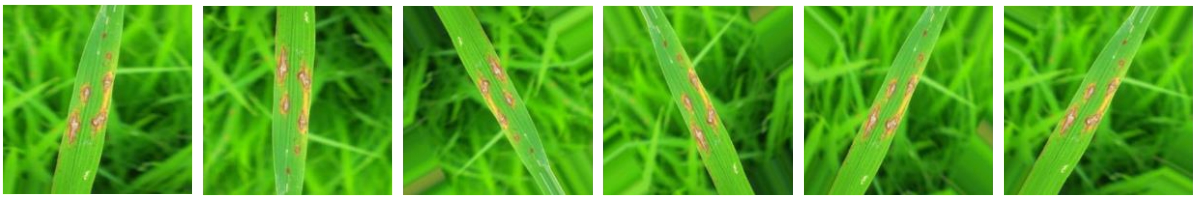

Figure 3. Augmentation results from the image of blast disease

Table 1. Details of rice leaf disease dataset

Leaf diseases	Original image	Training (60%)		Validation (20%)	Testing (20%)
		Before augmentation	After augmentation		
Bacteril Blight	1584	1267	7602	317	317
Blast	1440	1152	6912	288	288
Brown Spot	1600	1280	7680	320	320
Tungro	1308	1046	6276	262	262
Total	5932	4745	28470	1187	1187

2.3. Model CNN

Convolutional neural networks (CNNs) are the most popular deep learning algorithms for researchers to test [16]. CNN was first introduced in 1988 by Yann LeCun [17]. The introduction of CNN has changed the way problems are solved in the classification of an image [18]. In this study, six types of CNN architecture will be used to conduct experiments in classifying diseases in rice leaves, including InceptionV3 [19], ResNet50 [20], InceptionResnetV2 [21], DenseNet201 [22], MobileNet [23], and EfficientNetB3 [24]. The following will briefly explain the CNN architecture used in this study.

2.3.1. InceptionV3

InceptionV3 is a CNN architecture developed by Google at the imagenet large scale visual recognition challenge (ILSVRC) in 2012 [19]. InceptionV3 was developed to parse convolution [19]. This means that each convolution can be replaced by a convolution followed by a convolution, this can parse many parameters, avoid the problem of redundant fitting and strengthen the ability of nonlinear expressions [25].

2.3.2. ResNet50

ResNet50 is a CNN architecture that introduces the concept of shortcut connections [20]. The concept emergence of shortcut connections in the ResNet50 architecture is related to the vanishing gradient problem that occurs when efforts to deepen the structure of a network are carried out. However, deepening a network with the aim of improving its performance cannot be done simply by stacking layers. The deeper a network can lead to a vanishing gradient problem that can make the small gradient which results in decreased performance or accuracy [20].

2.3.3. InceptionResnetV2

InceptionResNetV2 was first introduced by Szegedy *et al.* [21]. InceptionResNetV2 architecture is a compound of the Inception structure and residual module. The convolution filter is combined with the residual connection which aims to avoid the problems caused by the deeper structure. Residual connection can reduce the time during the training process [26]. InceptionResnetV1 and InceptionResnetV2 have the same overall structure, but different modules in the network.

2.3.4. DenseNet201

DenseNet is a CNN architecture that introduces short connections that connect each layer directly to the other layers in a feed-forward manner. DenseNet has a narrow layer with a small set of feature maps that belong to the network [22].

2.3.5. MobileNet

MobileNet was first introduced by researchers from Google [23], to overcome the need for large computing resources so that this architecture can be used for mobile phones. MobileNet uses a convolution screen with a filter thickness that matches the thickness of the input. The MobileNet architecture is divided into 2 types of convolution, namely depthwise convolution and pointwise convolution.

2.3.6. EfficienNetB3

EfficienNet was first proposed by Tan and Lee in 2019 [24] and an architecture used to optimize classification networks. In general, there are 3 indicators used by most networks, including widening the network, deepening the network and increasing the resolution quality. Therefore, the application of the combined scaling model is applied to optimize the network width, depth and network resolution to improve accuracy [25].

Previously, the model used in this experiment was trained using a dataset from ImageNet. All pre-trained models on the CNN architecture by default have 1000 fully connected (FC) layer output nodes. For the output FC layer, it will be replaced with 4 nodes according to the number of classes in rice leaf disease and added with Softmax activation.

2.4. Evaluation of CNN architecture model

To evaluate and compare the performance of a tried CNN architecture. The first evaluation is carried out on training data and validation by calculating accuracy and loss and calculating the computation time of each CNN architecture. Furthermore, an evaluation of the architectural model that has been trained with testing data using the Confusion Matrix will be carried out, including calculating accuracy, precision, recall, and F1 Score. The following confusion matrix multiclass used in this study is shown in Table 2.

Table 2. Confusion matrix for multiclass

		Predicted Number			
		Class 1	Class 2	…	Class n
Actual Number	Class 1	X_{11}	X_{12}	…	X_{1n}
	Class 2	X_{21}	X_{22}	…	X_{2n}
	…	…	…	…	.
	Class n	X_{n1}	X_{n2}	…	X_{nn}

From the Table 2, we will get the number of true positives (TTP) for all classes, true negative (TTN), false positive (TFP), and false negative (TFN) for each class i which is calculated using (1)-(4) [27].

$$TPP_{all} = \sum_{j=1}^{n} x_{jj} \tag{1}$$

$$TTN_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} \sum_{\substack{k=1 \\ k \neq i}}^{n} x_{jk} \tag{2}$$

$$TFP_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ji} \tag{3}$$

$$TFN_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} x_{ij} \tag{4}$$

From the TPP, TTN, TFP and TFN results, then it will be used to calculate the overall accuracy (OA), precision (P), recall (R), f-measure score (F1) of each class i calculated using (5)-(8) [27], [28].

$$OA = \frac{TTP_{all}}{(Total\ Number\ of\ Entries)} \tag{5}$$

$$P_i = \frac{(TTP_{all})}{(TTP_{all} + TFP_i)} \tag{6}$$

$$R_i = \frac{(TPP_{all})}{(TTP_{all} + TFN_i)} \quad (7)$$

$$F_i = \frac{2\,(P_i)(R_i)}{P_i + R_i} \quad (8)$$

In this study, the experiment was carried out by applying batch size 32 to the trained model and batch size 1 to the test. The data randomization process was used during the training. Adam optimizer [29] with a learning rate of 0.0009 was used to minimize the loss function on the CNN model during the training process. All experiments conducted in this study use TensorFlow which is on Google Colaboratory as a cloud computing provider platform that has 12 Gb RAM specifications and an Nvidia K80s GPU.

3. RESULTS AND DISCUSSION

This section will describe the results of each experiment was conducted. Experiments were carried out on training data and validation data using the CNN InceptionV3, ResNet50, InceptionResnetV2, DenseNet201, MobileNet, and EfficientNetB3 architectural models. This experiment aims to find to accuracy, loss, and computation time required of each architectural model during the data training process. This experiment will be carried out using 50 training epochs. The results of the accuracy, loss, and training computation time of each CNN architecture can be seen in Table 3.

Table 3. Accuracy, loss and time computing training data and validation of CNN architecture after 50 training epochs

CNN Arsitektur	Train Acc (%)	Val Acc (%)	Train Loss	Val Loss	Time (Minute)
InceptionV3	99,34	99,92	0,0215	0,0039	77
ResNet50	98,65	99,75	0,0408	0,0104	86
InceptionResnetV2	99,61	99,83	0,0142	0,0043	117
DenseNet201	99,12	99,66	0,0242	0,0111	98
MobileNet	97,84	99,24	0,0573	0,0238	72
EfficientNetB3	85,48	89,81	0,5387	0,3465	112

From the result of experiments that have been done, InceptionResnetV2 gets the best results from all CNN architectural models that are trained with an accuracy value of 99.61%. Then followed by InceptionV3 architecture with 99.34% accuracy, DenseNet201 with 99.12% accuracy, ResNet50 with 98.65% accuracy, MobileNet with 97.84% accuracy and EfficientNetB3 with 85.48% accuracy. Although InceptionResnetV2 gets the best accuracy value of all architectures, it requires the longest computation time among all trained architectures to achieve 50 training epochs of 117 minutes. MobileNet gets the best compute time to reach 50 epoch, which is 72 minutes, followed by InceptionV3 with 77 minutes of computing time, ResNet50 with 86 minutes of computation time, DenseNet201 with 98 minutes of computation time, and EfficientNetB3 with 112 minutes of computation time.

Table 3 also displays the training loss of each trained CNN architectural model. The architecture that has the lowest training loss is InceptionResnetV2 with a value of 0.0142, followed by InceptionV3 with a value of 0.0215, DenseNet201 with a value of 0.0242, ResNet50 with a value of 0.0408, MobileNet with a value of 0.0573, and EfficienNetB3 with a value of 0.5387. The accuracy and loss charts in the training process can be seen in Figures 4-9.

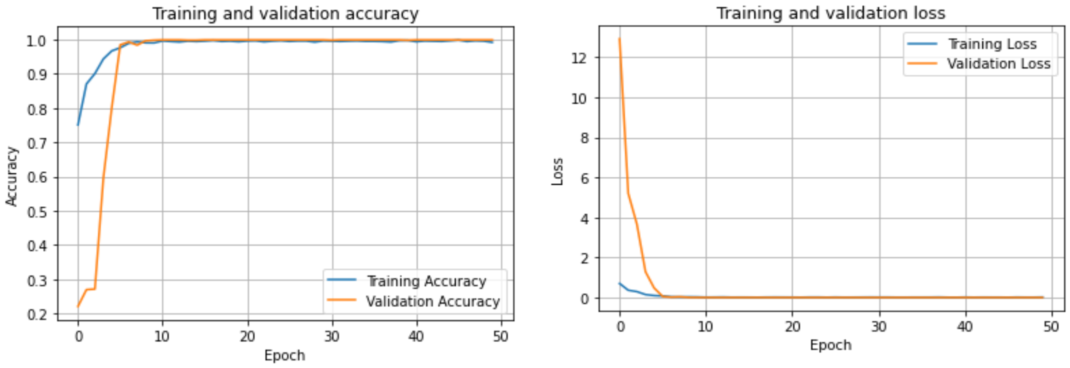

Figure 4. Accuracy dan loss training InceptionV3

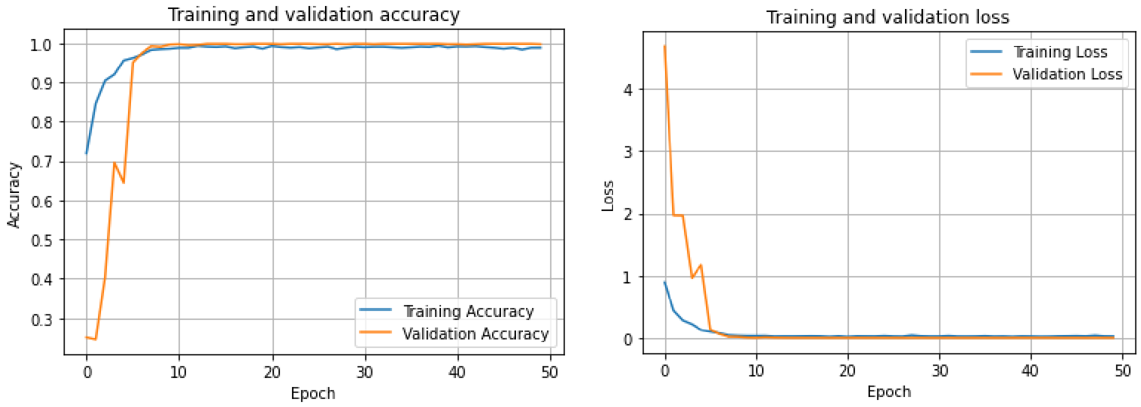

Figure 5. Accuracy dan loss training ResNet50

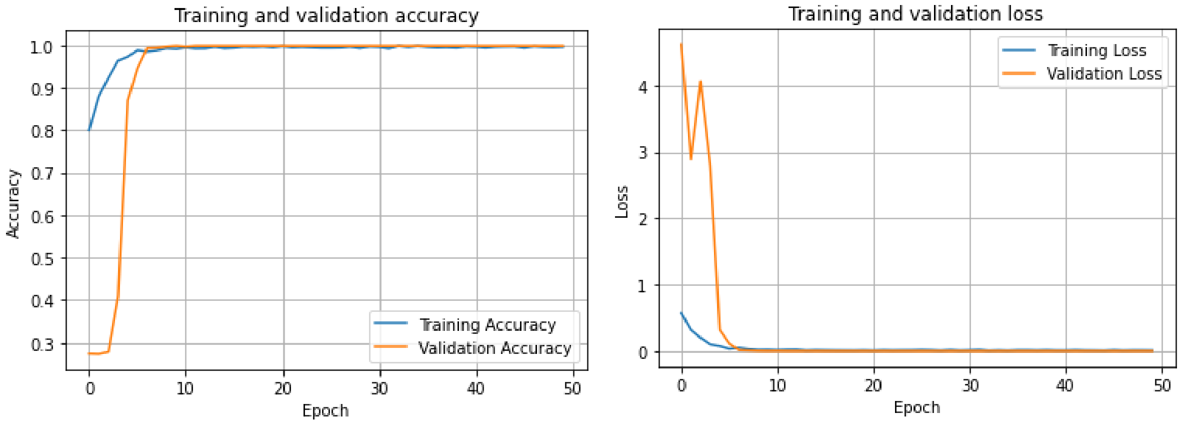

Figure 6. Accuracy dan loss training InceptionResNetV2

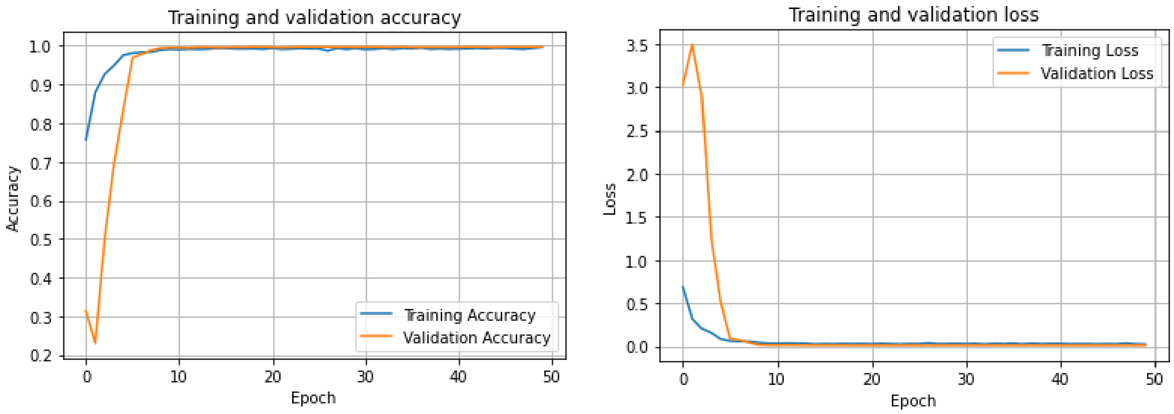

Figure 7. Accuracy dan loss training DenseNet201

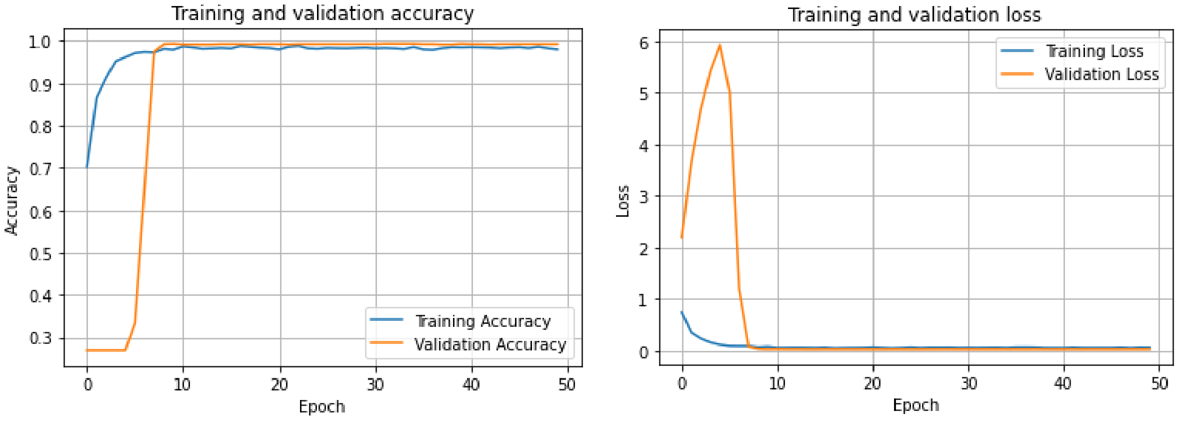

Figure 8. Accuracy dan loss training MobileNet

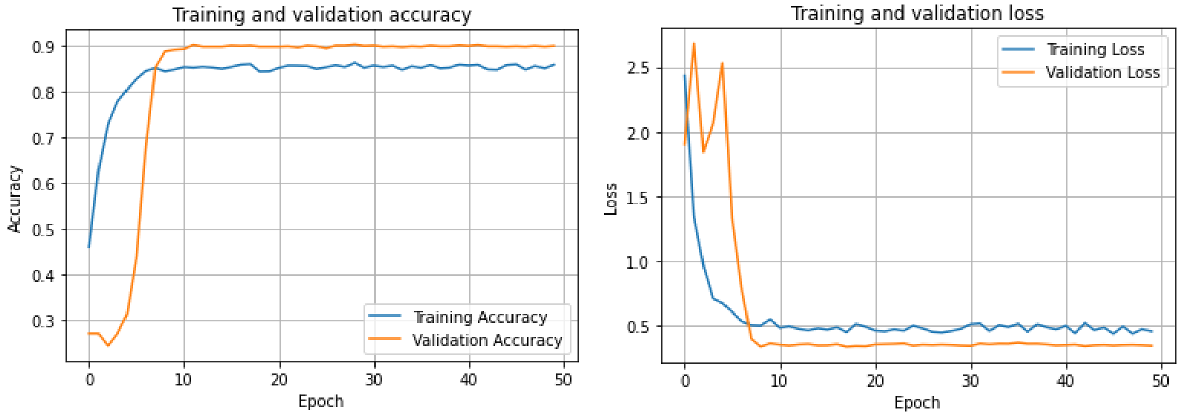

Figure 9. Accuracy dan loss training EfficientNetB3

Table 4 presents the evaluation matrix between the trained model and the testing data. The results of the test experiments conducted, InceptionResnetV2 and InceptionV3 got the best results from all the CNN architectural models tested with 100% accuracy, 100% precision, 100% Recall, and 100% F1 Score. Then followed by the Resnet50 and DenseNet201 architectures got an accuracy value of 99.83, precision 99.83%, Recall 99.83%, and F1 Score 99.83%. MobileNet gets an accuracy value of 99.33%, precision 99.36%, Recall 99.32%, F1 Score 99.34% and EfficientNetB3 gets an accuracy value of 90.14%, precision 90.46%, Recall 90.23%, F1 Score 90.28%.

Table 4. Evaluation of model training with data testing

CNN architecture	Test ACC (%)	Precision (%)	Recall (%)	FI score (%)
InceptionV3	100	100	100	100
ResNet50	99,83	99,83	99,83	99,83
InceptionResnetV2	100	100	100	100
DenseNet201	99,83	99,83	99,83	99,83
MobileNet	99,33	99,36	99,32	99,34
EfficientNetB3	90,14	90,46	90,23	90,28

Figure 10 shows the results of the InceptionV3 architecture model confusion matrix with data testing. From the 1187 sample data tested, no data were misclassified. All data were classified correctly as shown in Figure 10. The accuracy, precision, recall, and F1 score values can be seen in Table 4.

Figure 11 shows the results of the ResNet50 architecture model confusion matrix with data testing. From the 1187 sample data tested, there are 2 sample data which are misclassified. 1 sample of misclassified brown spot class data and 1 misclassified tungro class data sample as shown in Figure 11. The values of accuracy, precision, recall and F1 Score can be seen in Table 4.

Figure 12 shows the results of the confusion matrix of the InceptionResnetV2 architectural model with data testing. From the 1187 sample data tested, no data were misclassified. All data were classified correctly as shown in Figure 12. The value of accuracy, precision, recall and F1 Score can be seen in Table 4.

Figure 13 shows the results of the DenseNet201 architectural model confusion matrix with data testing. From the 1187 sample data tested, there are 2 data samples in the misclassified brown spot class as shown in Figure 13. The values for accuracy, precision, recall and F1 Score can be seen in Table 4.

Figure 14 shows the results of the MobileNet architecture model confusion matrix with data testing. From the 1187 sample data tested, 8 sample data were misclassified. Two samples of misclassified bacterial blight class data, and 6 misclassified blast class data samples as shown in Figure 14. The values for accuracy, precision, recall, and F1 Score can be seen in Table 4.

Figure 15 shows the results of the confusion matrix model for the EfficientNetB3 architecture with data testing. From the 1187 sample data tested, 117 sample data were misclassified. The 36 samples of misclassified bacterial blight class data, 57 misclassified blast class data samples, 18 misclassified brown spot class data samples and 6 misclassified tungro class data samples as shown in Figure 15. The accuracy, precision, recall, and F1-Score values can be seen in Table 4.

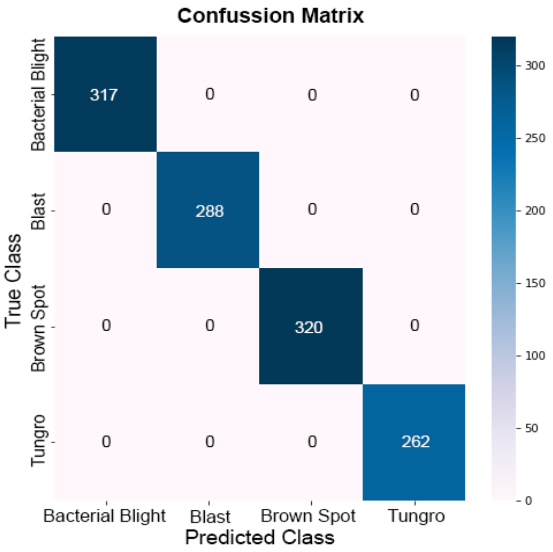

Figure 10. Confusion matrix of InceptionV3

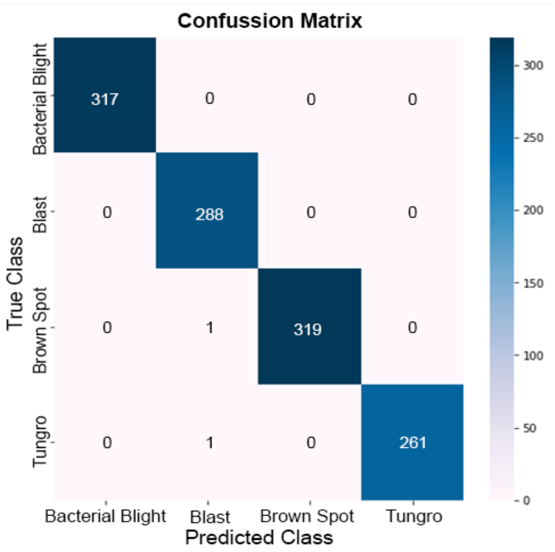

Figure 11. Confusion matrix of ResNet50

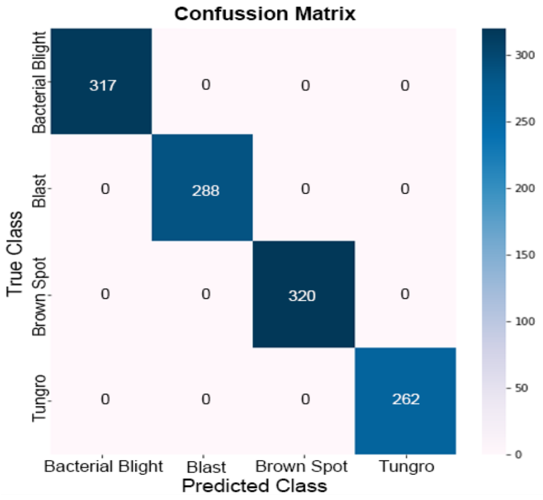

Figure 12. Confusion matrix of InceptionResnetV2

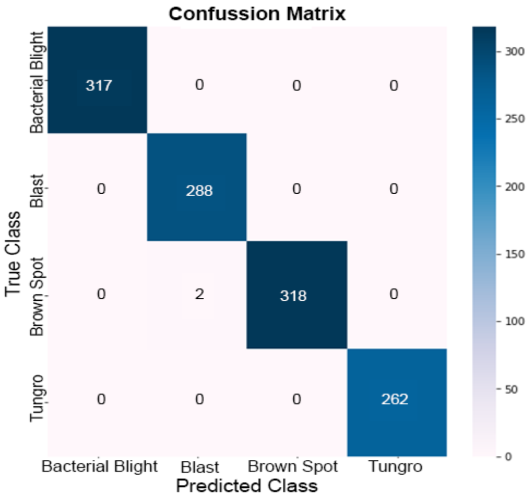

Figure 13. Confusion matrix of DenseNet201

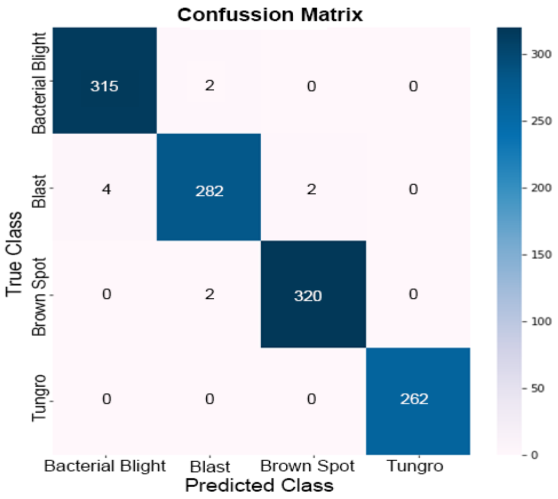

Figure 14. Confusion matrix of MobileNet

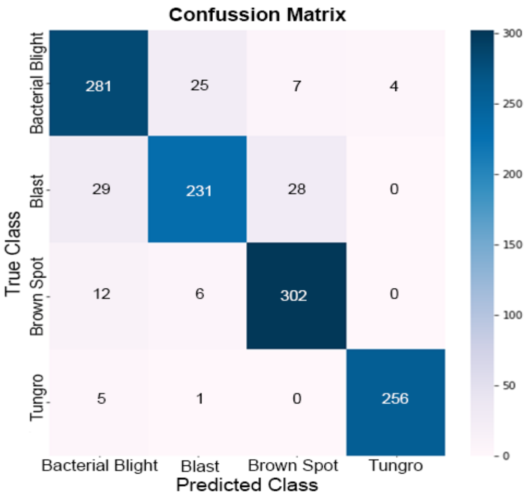

Figure 15. Confusion matrix of EfficientNetB3

Furthermore, after comparisons of the results of this experiment with several previous studies/research. By using the CNN architecture, the best performance can be obtained from this experiment for classifying diseases in rice leaves beyond conventional methods such as KNN [6], logistic regression, decision tree (DT), naïve bayes (NB) [7], SVM [8], and ANN [10]. The experiments in this study also have better performance than other CNN architectures, like VGG16 [11], AlexNet [13], and VGG19 [14].

4. CONCLUSION

After evaluating the experiments of each CNN architectural model, the best architectures for the classification of rice leaf disease are InceptionV3 and InceptionResnetV2, with an accuracy of 100%. Then followed by the ResNet50 architecture with an accuracy of 99.83%, DenseNet201 99.83%, MobileNet 99.33% and EffecientNetB3 90.14%. This experiment was carried out using Adam's optimization and modifying the batch size and learning rate. The result shows the proposed CNN model exceeds the conventional methods and other CNN architectures found in previous studies in the classification of rice leaf disease. It is important to develop a CNN model for further research that has better training time and accuracy. It is necessary to add more data on the types of diseases on rice leaves and more types of pests on rice leaves.

REFERENCES

[1] Central Bureau of Statistics, “Area of Rice Harvest and Production in Indonesia 2019,” *Berita Resmi Statistik*, no. 16, 2020.

[2] A. Leonard and P. Gianessi, “Importance of Pesticides for Growing Rice in South and South East,” *Croplife Foundation*, pp. 30–33, 2014.

[3] A. Nasution and B. Nuryanto, “Blas Pyricularia grisea Disease in Rice Plants and Control Strategy,” *Planting Food*, vol. 9, no. 2, pp. 85–96, 2015.

[4] Agricultural Research and Development Agency, “Technical Guidelines for Plant Disease Pests,” *Indonesian Center for Rice Researchers*, vol. 1, pp. 1–5, 2007.

[5] N. Lakshita, S. H. Poromarto, and H. Hadiwiyono, “Resistance of Some Rice Varieties to Cercospora oryzae,” *Agrotechnology Research Journal.*, vol. 3, no. 2, p. 75, 2019, doi: 10.20961/agrotechresj.v3i2.29976.

[6] M. Suresha, K. N. Shreekanth, and B. V. Thirumalesh, “Recognition of diseases in paddy leaves using knn classifier,” in *International Conference for Convergence in Technology (I2CT)*, 2017, pp. 663-666, doi: 10.1109/I2CT.2017.8226213.

[7] K. Ahmed, T. R. Shahidi, S. M. Irfanul Alam, and S. Momen, “Rice leaf disease detection using machine learning techniques,” in *International Conference on Sustainable Technologies for Industry 4.0 (STI)*, v 2019, pp. 1-5, doi: 10.1109/STI47673.2019.9068096.

[8] S. Pavithra, A. Priyadharshini, V. Praveena, and T. Monika, “Paddy Leaf Disease Detection Using Svm,” *International Journal of communication and computer Technologies*, vol.3, no. 1, pp. 16–20, 2015.

[9] F. T. Pinki, N. Khatun, and S. M. M. Islam, “Content based paddy leaf disease recognition and remedy prediction using support vector machine,” in *International Conference of Computer and Information Technology (ICCIT)* 2017, pp. 1-5, doi: 10.1109/ICCITECHN.2017.8281764.

[10] R. Deshmukh and M. Deshmukh, “Detection of Paddy Leaf Diseases,” *International Journal of Computer*

Applications, pp. 975–8887, 2015.

[11] S. Ghosal and K. Sarkar, “Rice Leaf Diseases Classification Using CNN with Transfer Learning,” in *IEEE Calcutta Conference (CALCON)*, 2020, pp. 230-236, doi: 10.1109/CALCON49167.2020.9106423.

[12] B. Knyazev, R. Shvetsov, N. Efremova, and A. Kuharenko, “Convolutional neural networks pretrained on large face recognition datasets for emotion classification from video,” *arXiv*, pp. 2–5, 2017.

[13] R. R. Atole and D. Park, “A multiclass deep convolutional neural network classifier for detection of common rice plant anomalies,” *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 1, pp. 67–70, 2018, doi: 10.14569/IJACSA.2018.090109.

[14] J. Chen, J. Chen, D. Zhang, Y. Sun, and Y. A. Nanehkaran, “Using deep transfer learning for image-based plant disease identification,” *Journal Computers and Electronics in Agriculture*, vol. 173, 2020, doi: 10.1016/j.compag.2020.105393.

[15] P. K. Sethy, “Rice Leaf Disease Image Samples,” *Mendeley Data, V1*, 2020, doi: doi: 10.17632/fwcj7stb8r.1.

[16] J. Gu *et al.*, “Recent advances in convolutional neural networks,” *Journal Pattern Recognition*, vol. 77, pp. 354–377, 2018, doi: 10.1016/j.patcog.2017.10.013.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2323, 1998, doi: 10.1109/5.726791.

[18] K. Annapurani and D. Ravilla, “CNN based image classification model,” *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 11, pp. 1106–1114, 2019, doi: 10.35940/ijitee.K1225.09811S19.

[19] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 2818–2826, 2016, doi: 10.1109/CVPR.2016.308.

[20] K K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[21] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-ResNet and the impact of residual connections on learning,” in *Conference on Artificial Intelligence*, pp. 4278–4284, 2017.

[22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 2261–2269, 2017, doi: 10.1109/CVPR.2017.243.

[23] A. G. Howard *et al.*, “MobileNets: Efficient convolutional neural networks for mobile vision applications,” *arXiv*, 2017.

[24] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in *International Conference on Machine Learning (ICML)*, pp. 10691–10700, 2019.

[25] W. Jia, J. Gao, W. Xia, Y. Zhao, H. Min, and J. T. Lu, “A Performance Evaluation of Classic Convolutional Neural Networks for 2D and 3D Palmprint and Palm Vein Recognition,” *International Journal of Automation and Computing*, vol. 18, no. 1, pp. 18–44, 2021, doi: 10.1007/s11633-020-1257-9.

[26] D. Krisnandi *et al.*, “Diseases Classification for Tea Plant Using Concatenated Convolution Neural Network,” *Journal CommIT (Communication and Information Technology)*, vol. 13, no. 2, pp. 67–77, 2019, doi: 10.21512/commit.v13i2.5886.

[27] C. Manliguez, “Generalized Confusion Matrix for Multiple Classes,” pp. 5–7, 2016.

[28] D. Zhang, J. Wang, and X. Zhao, “Estimating the uncertainty of average F1 scores,” in *Proceedings of the 2015 International Conference on The Theory of Information Retrieval*, pp. 317–320, 2015, doi: 10.1145/2808194.2809488.

[29] D. P. Kingma and J. L. Ba, “Adam: A Method for Stochastic Optimization,” *arXiv*, pp. 1–15, 2014.