Jurnal Ilmu Komputer dan Informatika (JIKI) Vol. 6, No. 1, June 2026, Page. 1-16
P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

Application of CNN VGG16 for Digital Image-based Rice Leaf Disease Classification with Optimizer Analysis and Real-Time Validation

Lailia Rahmawati[*1], Nur Habib Nasidik2 ,Winarti[3]

[1,2,3] Informatics Engineering, Faculty of Engineering, Darul 'Ulum University Jombang, Indonesia

Email: [1]liaundarjombang@gmail.com

Abstract

Rice leaf diseases can significantly reduce yields, while manual identification takes a long time and risks producing errors. This research proposes a deep learning-based rice leaf disease classification system by utilizing the VGG16 Convolutional Neural Network (CNN) architecture through a transfer learning approach. The dataset was obtained from Kaggle with five categories, namely bacterial leaf blight, rice blast, brown spot, healthy, and tungro, which were processed through normalization and augmentation before being divided into training and validation data. The model was trained using two optimizers, Adam and Adadelta, with epoch variations of 20, 25, 35, and 50 to compare their performance. Experimental results showed that Adam produced the best validation accuracy of 96.24% and testing accuracy of 97.17%, with an average F1-score of 0.98, while Adadelta only achieved a maximum validation accuracy of 85.3%. Evaluation using confusion matrix and classification report further confirmed the reliability of the model, with Adam even achieving 100% accuracy on real-time testing. These findings show that the VGG16 CNN with Adam optimizer is capable of detecting rice leaf diseases quickly and accurately, thus contributing to early detection in precision agriculture and supporting food security.

Keywords: *CNN, deep learning, image classification, optimizer, rice leaf disease, VGG16*

1. INTRODUCTION

One of the factors that causes a decrease in the quality and quantity of agricultural products is the presence of diseases in plants. Infections in rice leaves can drastically reduce crop yields. In the growth process, rice plants are very susceptible to various diseases and pests, such as leaf wilting, tungro, rice blight, and grass dwarf. Usually, farmers are quick to take action by using pesticides or other preventive methods when their rice crops are infected with diseases or pests. However, such measures are sometimes ineffective in dealing with certain diseases or pests that attack plants.[1]

The three most important food commodities in the world are rice, corn, and wheat. Most people around the world rely on rice as their staple food source because of the rice crop[2]. The need for rice continues to increase as a result of the increase in population. As a result, rice production must be increased by more than 40% to maintain food security, social welfare, and economic development[3].Data shows that Indonesia's milled dry grain (GKG) production in 2023 will decrease by around 2.05% from the previous year to 53.63 million tons. The World Food and Agriculture Organization (FAO) says that diseases and pests cause 20–40% of crop losses worldwide. Meanwhile, the International Rice Research Institute (IRRI) says that rice crop diseases cause losses of about 37% every year[4].

As an agrarian country, Indonesia is highly dependent on the agricultural sector, especially rice, as its main source of food [5]. Rice leaf disease, which accounts for about a quarter of total crop failures, is one of the causes of production losses.[6]. Leaf image processing to detect diseases is essential for farmers to prevent diseases early. Diseases such as bacteria and fungi usually cause leaf disease. Blisters, bacterial blight, and brown spots are the most common types.[7]

P-ISSN: 2807-6664
https://jiki.jurnal-id.com
E-ISSN: 2807-6591
DOI: https://doi.org/10.54082/jiki.287

As technology advances, manual methods for detecting plant diseases are considered less effective and prone to delays. Deep learning, especially Convolutional Neural Network (CNN), can detect plant diseases through digital image analysis very well. With their ability to extract complex visual features, CNNs do not require a manual morphological feature extraction process[8]. In plant disease classification research, popular CNN architectures such as VGG16[9], ResNet, and MobileNet are widely used.[10].

Previous studies have demonstrated CNN's ability to classify rice leaf diseases. For example, a study using ResNet-50 found six types of rice diseases with an accuracy of 79.52% in the test data and a validation accuracy of 90% in the classification report.[11]Another study used MobileNet for the classification of rice leaves with 91% training accuracy and 90% validation, but overfitting on the loss value[12]. Research conducted by [13] implement the MobileNet architecture to classify the health conditions of rice plants based on the inputted images. Using the MobileNet base model as a feature extractor, the study achieved an accuracy of 91% on training data and 90% on validation data after 20 epochs. However, there was significant overfitting with a loss value of 27% in training data and 37% in validation data. Research conducted by [14] achieved the best performance in terms of accuracy of Adam validation across all three CNN architectures of 91.1%, followed by Rmsprop (87.7%), and Nadam (86.5%), although there was variation in terms of convergence time and loss. Other optimizers showed no significant performance improvement compared to simple CNN, except for a marginal improvement in terms of validation convergence time for Nadam on the CNN depth model and better validation losses for Rmsprop at depth/with CNN. The worst overall performance was that Adadelta achieved accuracy (67%). Research conducted [15] The CNN-VGG19 model—a convolutional neural network developed by the Visual Geometry Group—with a transfer-learning approach to accurately diagnose and classify rice leaf disease. This plan uses a transfer learning method that is able to identify the class brown spots and is based on the VGG19 architecture. In the application of rice leaf disease datasets, this model obtained an accuracy of 93,0%**.** Other metrics, namely F1-score, sensitivity, specificity, and precision**,** each has a value 89.9%, 94.7%, 92.4%, and 90.5%**.** Based on the review,suggests that previous research still has some limitations. These include limited dataset sizes, lack of real-time model testing, and lack of comparisons between optimizers. As a result, this study was conducted to implement CNN with the VGG16 architecture using a transfer learning approach, conduct real-time testing and compare the performance of the Adam and Adadelta optimizers. The main focus of this research is to find more accurate and useful methods of detecting rice leaf disease and support the development of AI-based precision agriculture.

2. RESEARCH METHODS

2.1. Convolutional Neural Network (CNN)

CNN's ability to identify various objects in images makes it one of the most widely used deep learning algorithms for processing visual data. The CNN is made up of rows of neurons that compose filters of a certain size in terms of length and width. In addition, CNN also comes with activation, bias, and weight functions. CNN has similarities with other types of neural networks in deep learning. The structure of a CNN consists of several main layers: an input layer, a convolutional layer, a nonlinear layer, a pooling layer, and a fully connected layer, all of which play a role in processing images to find and recognize objects.[16]

2.2. VGG16 Architecture

VGG16 has 16 layers, consisting of 13 convolutional layers and 3 fully connected layers, which makes it one of the most preferred convolutional neural network designs. The uniqueness of this architecture lies in the use of a small filter measuring 3 x 3 with a stride of 1 and "same" type padding on each layer of convolution, which makes it possible to maintain the spatial dimensions of the image. With kernel sizes 2 × 2 and stride 2 applied consistently across the network, the pooling process is done using max pooling. This repetitive pattern of using small filters allows the VGG16 to extract visual features with high depth while maintaining computational efficiency.

In this study, a transfer learning approach with the initial weight of the ImageNet dataset was used. In VGG16, the entire convolution layer is frozen so it doesn't need to be retrained. On the other

P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

hand, the fully connected part is changed to fit the number of existing classes of rice leaf diseases, which are five classes. Other layers used are the Flatten layer, Dense layer with 256 neurons and the ReLU activation function, the Dropout layer with a value of 0.5 to prevent overfitting, the Dense output layer with 5 neurons and the Softmax activation function for multi-class classification.

For the Adam and Adadelta optimizers, a batch size of 64, a maximum number of epochs of 50, and a training rate of 0.00005 are used. On the hidden layer, the ReLU activation function is used because it is able to overcome the vanishing gradient problem and speed up the calculation. On the output layer, Softmax is used as it is suitable for multi-class classification. It is hoped that with this configuration, the model can make good generalizations on various images of rice leaves.

2.3. Research Methods

The purpose of this study is to identify the variety of diseases that can attack rice leaves. This study applies a quantitative approach combined with experimental design. Deep learning technology was implemented, specifically using convolutional neural network (CNN) algorithms with VGG16 structures. Data collection, preprocessing stage, classification, and model assessment are the main focuses of this study. Figure 1 shows the four sequential stages of the process.

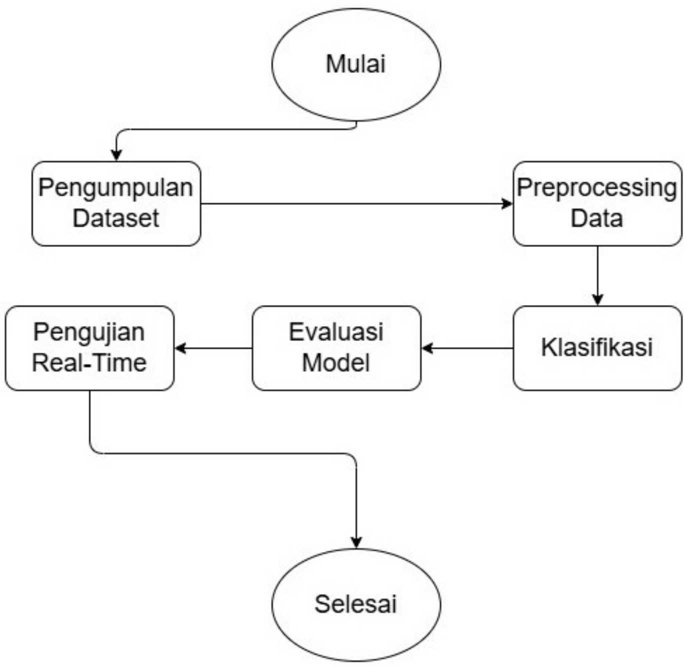

Figure 1. Research Method Flow

The research flow shown in Figure 1 begins with the collection of data in the form of images of rice leaves from various categories, namely Bacterialblight, Blast, Brownspot, Healthy, and Tungro. The collected data then undergoes a pre-processing process that includes resizing the image to 224×224 pixels to match the input standards of the VGG16 architecture, as well as the application of augmentation techniques such as rotation, flipping, zooming, shifting, and brightness adjustment to increase data variation. The goal of this stage is to improve the quality of the dataset and prevent overfitting during model training. Then, the processed data was used in the classification stage by applying the VGG16-based CNN model through the transfer learning method. This model is trained to recognize visual patterns in each disease category so that it can make precise predictions. After the training is completed, the model evaluation is carried out using test data as well as metrics such as accuracy, precision, recall, F1-score, and confusion matrix to assess the performance of the classification. This process is crucial to determine how effective the model is in detecting diseases in rice leaves. The end of this study is the

P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

drawing of conclusions based on the results of the evaluation to assess the success of the method applied and its potential use in the real-time rice leaf disease detection system.

2.4. Dataset Collection

The dataset used to instruct the model is taken from [17]. This dataset is downloaded from the Kaggle dataset repository in csv format. The contribution is Tedi Setiady and is available at the following link: https://github.com/maimunul/Rice-Leaf-Disease-Classification-using-CNN.git. The contents of the dataset are presented on Table 1.

Table 1. Dataset Content

Sempel	**Amount of Data**	**Data Source**
Bacterial Blight	1.584	*Kaggle (Rise Leaf Disease Dataset)*
Blast	1.440	
Brown Spot	1.600	
Tungro	1.308	
Healthy	1.528	
Total	7.460	

In table 1, the dataset used in this study consists of 7,460 images of rice leaves divided into five classes: Bacterial Blight (1,584 images), Blast (1,440 images), Brown Spot (1,600 images), Tungro (1,308 images), and Healthy (1,528). To increase data variety and avoid overfitting, ImageDataGenerator from Keras is used with the following parameters:

1) rotation_range = 30°
2) width_shift_range = 0.2
3) height_shift_range = 0.2
4) shear_range = 0.2
5) zoom_range = 0.2
6) horizontal_flip = True
7) brightness_range = [0.8, 1.2]
8) fill_mode = 'nearest'

After augmentation, the overall dataset was divided into 90% for training data and 10% for validation data, with 8 192 training photos and 905 validation photos. Table 2 shows the details for each class.

Table 2. Number of Datasets After Augmentation

Class	**Training**	**Validation**	**Total**
Bacterialblight	1 742	194	1 936
Blast	1 584	176	1 760
Brownspot	1 760	195	1 955
Healthy	1 680	187	1 867
Tungro	1 426	153	1 579
Total	8 192	905	9 097

2.5. Preprocessing

Data preprocessing allows data to be ready for processing[18]. Before data processing, the dataset is resized to 224 x 224 pixels to meet the VGG16 input standard, To speed up the convergence of the model, the pixel value is reset to the range [0, 1] (rescale=1./255). Furthermore to enhance the image, rotation, flipping, zooming, shifting, and brightness are used. This is done to prevent the image from becoming too large. 90% of the dataset is used for instruction and 10% for validation. The validation_split parameter in the imagedatagenerator incorporates this difference. And the augmentation technique aims to increase the variation of the data without including the original image, to improve the resistance of the model to changes in exposure, rotation, position, and image shift. Reduces the possibility of overfitting and improves the model's ability to generalize to new data.

P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

2.6. Classification

The classification in this study classifies images and has their own way of exploiting features[19]. At this classification stage, the CNN algorithm of the VGG16 architecture is used with a 90:10 ratio of training and validation data. Then conducting tests with different optimizers, namely adam and adadelta with epochs 20, 25, 35, and 50.CNN consists of input layer and fully connected layer, the layer is an algorithmic process to recognize objects. This VGG16 uses a filter with a size of 3x3 to capture visual features from the input imagery in the classification process, extracting texture, pattern, color and shape features to identify the disease class of leaves.

2.7. Model Evaluation

Evaluate algorithm performance in the classification process such as algorithm performance. Displays all the results of the classification process when it is carried out.starting from the results of the model run to find out the results of each epoch, graph, metrix and F1 score[20].

Based on the values of true positive (TP), true positive (FP), and true negative (FN), the accuracy metrics, recall, and F1 score are used to evaluate the model's performance, with the calculation formula shown below.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FP} \tag{2}$$

$$F1 = \frac{2\ x\ Precision\ x\ Recall}{Precision + Recall} \tag{3}$$

2.8. Real-Time Testing

Real-time testing was carried out with the laptop's internal webcam with Google Colab to test the model's performance on new data. The relatively stable light intensity is generated by the white LED lamp used in the test in a normal room environment. The camera used has a resolution of 720p and takes images from a distance of 30 to 40 cm from the object. A total of 25 images were used to test each class of disease. In order for other researchers to conduct experiments with the same level of accuracy, the details of these conditions are clearly written.

2.9. Device Parameters and Specifications

The CNN VGG16 model is trained and tested on systems with an Intel Core i7-8665U processor, 16 GB of RAM, and an NVIDIA GPU, which can be accessed through Google Colab to improve computing performance. Software used includes TensorFlow 2.x and Keras for modeling, as well as scikit-learn, Matplotlib, and Seaborn for analysis and visualization of results, as well as Windows 10 and Python 3.9 programming.

The training parameters are set as follows: batch size 64, as this compensates for the GPU memory efficiency and gradient stability. While Softmax is used on the output layer as it is suitable for multi-class classification, the ReLU activation function is used on the hidden layer to solve gradient vanishing issues and speed up calculations. A dropout of 0.5 is used to prevent overfitting. Because Adam is adaptive and able to accelerate convergence, the Adam Optimizer with a learning rate of 0.00005 is chosen. Adadelta is used as a comparator. The sum of fifty epochs was used for the training, which was evaluated at several points (20, 25, 35, and 50 epochs) to evaluate the influence of the number of epochs on the model's performance.

3. RESULTS AND DISCUSSION

The results of this experiment used the cnn vgg16 algorithm by testing the model with different optimizers to find the best accuracy results.

P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

3.1. Dataset Description

This research dataset contains five categories of rice leaf imagery, namely Bacterialblight, Blast, Brownspot, Healthy, and Tungro. Data is stored in Google Drive with a separate folder structure for each class. The proportion of dataset sharing is 90% for training and 10% for validation, which is set using the validation_split parameters in the ImageDataGenerator.

To improve the model's ability to recognize image variations, an augmentation process is carried out which includes rotation, *zoom*, *shearing*, brightness level adjustment, and *horizontal flip*. After augmentation, the total training images amounted to 8,192 images, while the validation data amounted to 905 images. The resolution of the entire image was changed to 224×224 pixels to match the input of the VGG16 model. Figure 2 shows the contents of the dataset.

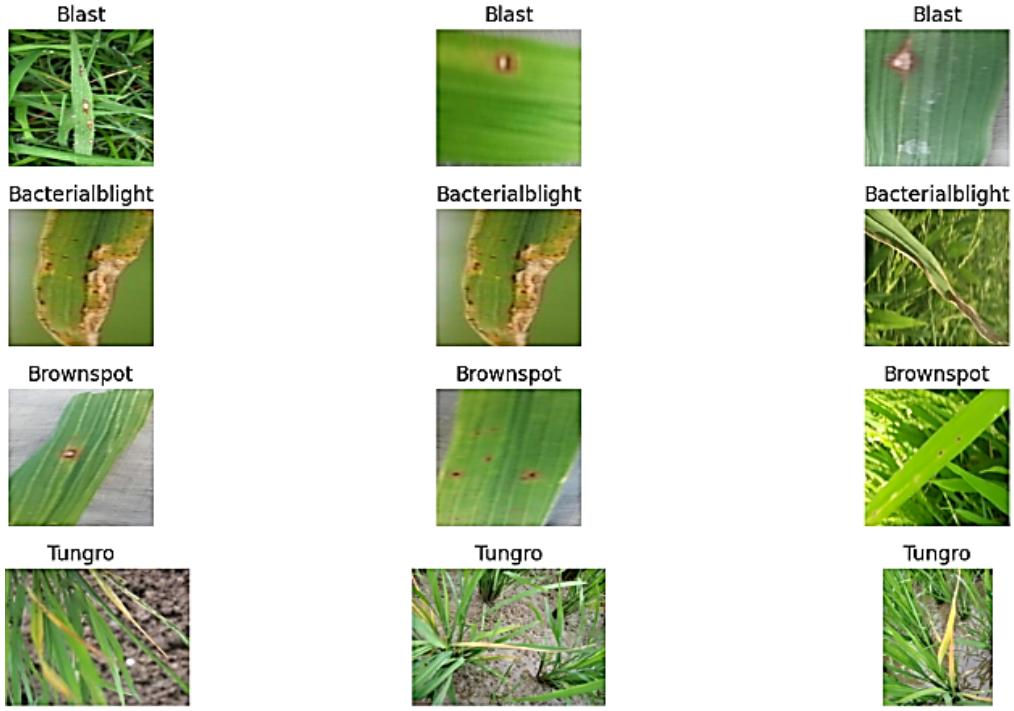

Figure 2. Dataset Content

Each category of rice leaves used in this study is illustrated in Figure 2. Each class has specific visual characteristics. Characteristics such as small patches on the leaves for Blast, leaf edge damage for Bacterial blight, brown spots on Brownspots, discoloration of Tungro to yellow-orange, and the Healthy class has fresh green leaves. To make a more accurate classification, these image variations help the model learn the visual patterns of each disease.

3.2. Preprocessing

In the preprocessing phase, the rice leaf image is divided by a pixel value of 255 and normalized to the range [0,1]. The data enhancement method is used to increase the variety of the training data without increasing the number of original images. These techniques include random rotation ±30°, horizontal and vertical shift of up to 20%, 0.2 filter, zoom ±20%, horizontal reversal, and 80–120 percent brightness adjustment. The transformed blank pixels are populated with the nearest method. To make the model more resistant to differences in orientation, position, and lighting, this strategy aims to increase image diversity. To ensure an objective model evaluation, the data is divided by a ten percent validation distribution. Figure 3 shows more clearly.

Vol. 6, No. 1, June 2026, Page. 1-16
https://jiki.jurnal-id.com
DOI: https://doi.org/10.54082/jiki.287

```
datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.1,
    rotation_range=30,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    brightness_range=[0.8, 1.2],
    fill_mode='nearest'
)
```

Figure 3. Preprocessing

In Figure 3 the ImageDataGenerator code is used for image preprocessing and enhancement. The data is divided 90% for training and 10% for validation, and scaled to the range of 0–1. For augmentation, rotation, shift, shear, zoom, horizontal reversal, and brightness variation are used. Meanwhile, the nearest method is used to fill in the empty area. So that the model is stronger and not easily overfitting, this technique increases the variety of data.

3.3. Classification Results

The model used is the VGG16 architecture with the initial weight of the ImageNet dataset. The entire convolutional layer is frozen so that training is only done on the additional layer at the top of the model. The layer consists of *a Flatten layer*, *Dense* contains 256 neurons with a ReLU activation function, *a 0.5 Dropout* to reduce the risk of *overfitting*, and *a Dense output* of five neurons with a *softmax activation function*. The training was conducted using two *optimizers*, namely Adam and Adadelta, with variations in the number of *epochs* 20, 25, 35, and 50. The results of the training with the epoch carried out can be seen in table 3.

Table 3. CNN VGG16 Model Training Results with Adam and Adadelta Optimizer

Optimizer	Epoch	Train Acc	Val Acc	Train Loss	Val Loss
Adam	20	0.962	0.966	0.113	0.112
Adam	25	0.960	0.965	0.100	0.099
Adam	35	0.976	0.986	0.067	0.070
Adam	50	0.980	0.980	0.061	0.066
Adadelta	20	0.269	0.433	1.509	1.495
Adadelta	25	0.809	0.829	0.537	0.534
Adadelta	35	0.849	0.851	0.446	0.470
Adadelta	50	0.865	0.853	0.390	0.423

Table 3 shows that accuracy values tend to increase over time, while loss values tend to decrease. When compared to Adadelta, Optimizer Adam results in higher validation accuracy on all epoch variations.

In addition to numerical results, the training process is also visualized in accuracy and loss graphs. The results of training with the Adam optimizer are shown in Figure 4 and Figure 5 of the training results with the Adadelta optimizer.

Vol. 6, No. 1, June 2026, Page. 1-16
https://jiki.jurnal-id.com
DOI: https://doi.org/10.54082/jiki.287

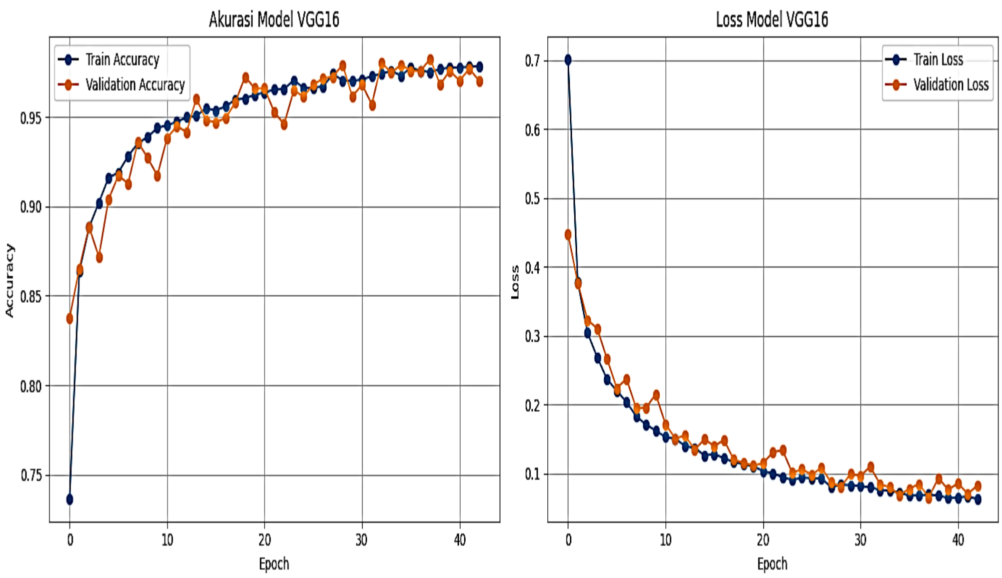

Figure 4. Adam's Accuracy Graph

In figure 4 the accuracy graph shows a consistent improvement in training and validation data from the beginning of training, until both reached a value of around 98% by the end of *the epoch*. The next stage is to display a graph with an adadelta optimizer for the results shown in figure 5 below.

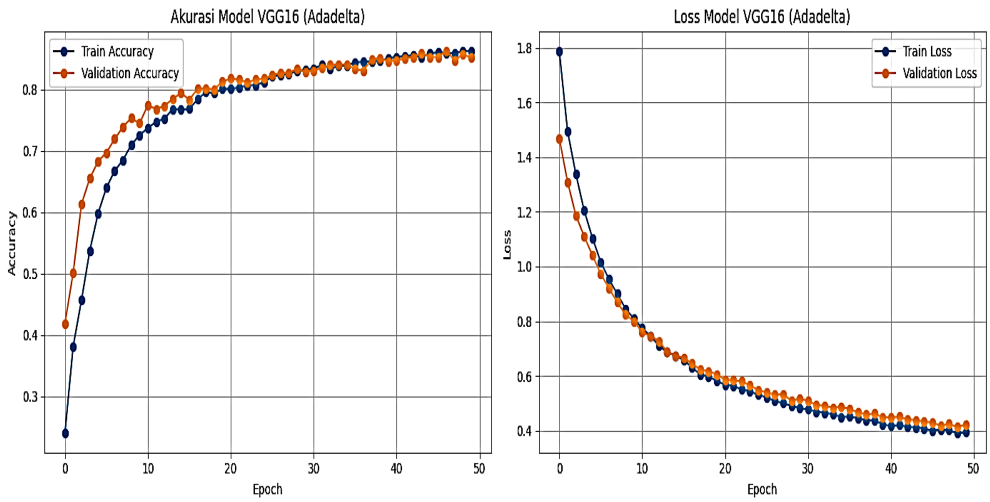

Figure 5. Adadelta Accuracy Chart

Figure 5 shows gradual progress on training and validation data since the beginning of the training process, reaching figures of about 86% and 85% by the end of the period.

4. DISCUSSION OF CLASSIFICATION RESULTS

Based on the results of the experiments contained in Table 2, it can be seen that the use of the Adam optimizer consistently results in superior performance compared to Adadelta for various epoch numbers. At the 35th epoch, Adam managed to achieve the highest validation accuracy of 0. 986 with a validation loss of 0. 070, which shows an excellent generalization capacity of the model. Even at epoch 50, Adam's performance remained high with a validation accuracy of 0. 980 and validation loss 0. 066, which indicates that the model remains consistent without any indication of overfitting.

In contrast, the Adadelta optimizer showed relatively low performance in the 20th epoch with a validation accuracy of 0.433 and a validation loss of 1.495, although its performance gradually improved as the number of epochs increased. Adadelta's best improvement occurred in the 50th epoch, with a validation accuracy of 0.853 and a validation loss of 0.423**,** but its value was still below Adam's.

P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

This difference is in line with research showing that Adam excels in the classification of plant imagery due to the ability to adapt learning speed adaptively, resulting in more stable gradients and faster convergence. However, Adadelta tends to take longer to achieve its best performance and is often stuck at higher loss values.

In figure 4 Adam's optimizer accuracy graph shows a consistent improvement in training and validation data from the beginning of training, until both reached a value of about 98% by the end of the epoch. The proximity of the two curves indicates the generalization ability of the model to be good in the absence of significant differences between the training and validation data. Meanwhile, the loss chart shows a sharp decline in the early phase of training, followed by a gradual decline to a steady in the range of 0.06. The similar tendency of the loss curve in the training and validation data reinforces the indication that the model has reached convergence with stable performance.

Figure 5 of the accuracy graph of the Adadelta optimizer shows the gradual progress on training and validation data since the beginning of the training process, reaching figures of about 86% and 85% by the end of the period. The similarity in the loss graphs of the two data sets suggests that the model can be applied in general without any significant differences between training and validation data. Meanwhile, the loss graph shows a consistent decline since the beginning of the training, from a figure of more than 1.4 to a stable value of around 0.39 for training data and 0.42 for validation data. Similar patterns on the loss graph of the two data sets reinforce the signal that the model has achieved convergence, although its performance is still not optimal.

Previous research results showed that Adam has a higher convergence speed and accuracy compared to other optimizers, such as Adadelta. Research conducted by [14] achieved the best performance in terms of accuracy of Adam validation across all three CNN architectures of 91.1%, followed by Rmsprop (87.7%), and Nadam (86.5%), although there was variation in terms of convergence time and loss. Other optimizers showed no significant performance improvement compared to simple CNN, except for a marginal improvement in terms of validation convergence time for Nadam on the CNN depth model and better validation losses for Rmsprop at depth/with CNN. The worst overall performance was that Adadelta achieved accuracy (67%).

4.1. Evaluation

After the training process is completed, a thorough analysis is carried out to assess the performance of the model in identifying various diseases in rice leaves. This analysis involves the use of a confusion matrix and the calculation of metrics such as precision, recall, and F1 score for each disease category. In addition, a comparison of the effectiveness of various optimizers is carried out to ensure that the model can accurately recognize the existing classes. The results of the matrix can be seen in Figure 6.

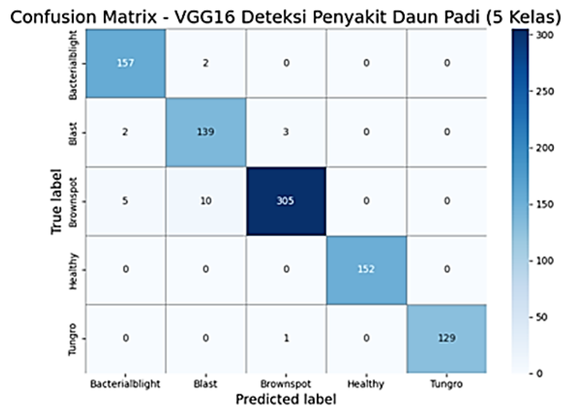

Figure 6. Adam's Matrix

Using this Adam, in figure 6, the prediction is correct by 305 for brownspots, 157 for bacterial blight, and 152 for well-being. In addition, adadelta also generates matrix values, as shown in figure 7.

P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

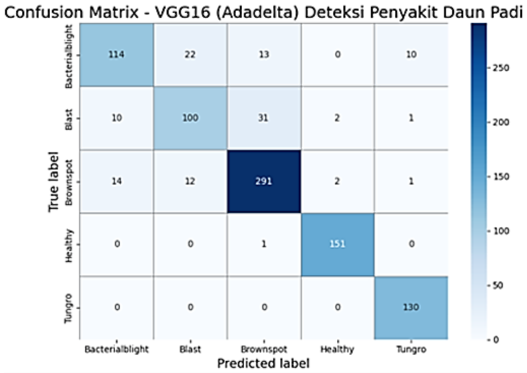

Figure 7. Adadelta Matrix

Table 4. Shows the results of the precision values, recalls, and F1-Adam scores.

Class	Accuracy	Recall	F1 Score	Support
Bacterialblight	0.96	0.99	0.97	159
Blast	0.92	0.97	0.94	144
Brownspot	0.99	0.95	0.97	320
Healthy	1.00	1.00	1.00	152
Tungro	1.00	0.99	1.00	130
Accuracy			0.97	905
Macro Avg	0.97	0.98	0.98	905
Weighted Avg	0.98	0.97	0.97	905

In figure 7 using adadelta this achieved the correct prediction of 291 Brownspot, 151 Healthy, and Tungro 130.The confusion matrix validates the model's capability in classifying five categories of rice diseases. The performance of the model is particularly effective in identifying Brownspots and Healthy. However, this matrix also reveals challenges in distinguishing between Bacterialblight, Blast, and Brownspot, where there is a significant misclassification between these classes.

For each disease class, the results of the metrix calculation such as F1 score, precision, and recall are given after the matrix results. The value of this metrix is based on the matrix. Table 4 provides more clarity.

In Table 3 the classification metrics with the adam optimizer show that the model has very superior performance with an overall accuracy of 97%. The model's performance was solid across all disease categories, with F1-Score values close to perfect in the Healthy (1.00) and Tungro (1.00) classes. This strong performance was also seen in the Brownspot (0.97), Bacterialblight (0.97), and Blast (0.94) classes. These data confirm the model's ability to detect and differentiate all five classes of diseases with a very high level of precision and sensitivity, making them highly reliable for this classification task. In addition to adam, adadelta also produces the matrix value shown in table 5.

Table 5. Adadelta Precision Score, Recall and f1-Score Results

Class	Accuracy	Recall	F1 Score	Support
Bacterialblight	0.83	0.72	0.77	159
Blast	0.75	0.69	0.72	144
Brownspot	0.87	0.91	0.89	320
Healthy	0.97	0.99	0.98	152
Tungro	0.92	1.00	0.96	130
Accuracy			0.87	905
Macro Avg	0.87	0.86	0.86	905
Weighted Avg	0.87	0.87	0.87	905

In addition to precision values, recall, and f1-score on the adadelta optimizer, it gets pretty good accuracy. Classification metrics show the model performs well, with an overall accuracy of 87%. The model is particularly effective in identifying the *Healthy* and *Tungro* classes, both of which show very high F1 scores (0.95), and also show strong performance in the *Brownspot* class (0.89). However, this model shows a challenge in predicting *the Bacterialblight* and *Blast* classes, where the F1-Score score is relatively lower (0.77 and 0.72). Overall, these data indicate that although the model has solid classification capabilities, there is still room for improvement, particularly in the identification of some specific types of diseases.

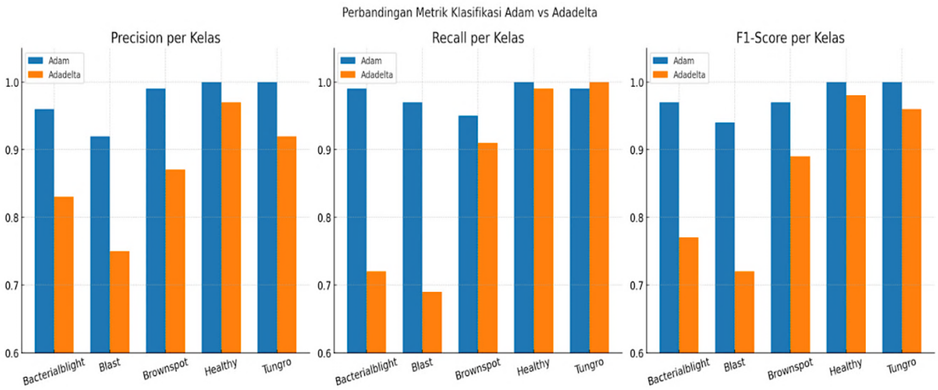

Figure 8. Comparison Chart Results

A thorough analysis of the matrix and matrix values reveals that the selection of optimizers has a very important influence on the performance of the model. Models optimized using Adam consistently provide much better results than those using Adadelta. Not only does Adam show better convergence speeds and is able to achieve validation accuracy of more than 95%. But it also produces a final accuracy of 97% with very few classification errors. On the other hand, Adadelta showed relatively lower results, with a final accuracy of 87% and a fairly high level of confusion between classes. Therefore, it can be concluded that Adam is a more effective and optimal optimizer in the classification of rice diseases using the VGG16 model.The results of the model assessment, which are displayed in the form of a comparison graph, show the performance of the precision classification, recall, and F1 score metrics for each type of disease in rice leaves, using the Adam and Adadelta optimizers.

Overall, Adam's optimizer performs much more consistently and better than Adadelta in almost all disease categories. In the bacterial blight and blast category, the F1 score for Adam reached 0.97 and 0.94, while Adadelta only achieved 0.77 and 0.72. This suggests that models with Adadelta have difficulty detecting both diseases, which could result in misclassification in the field.

In the "Brownspot," "Healthy," and "Tungro" categories, the performance of both optimizers is quite high, but Adam shows better consistency with an F1 score close to 1. 00. Although Adadelta was able to score high scores for "Healthy" (0.98) and "Tungro" (0.96), the score was still slightly below Adam's 1. 00. For the results of the comparison are shown in figure 8.

Figure 8 highlights the importance of optimizer selection in increasing model effectiveness. Not only did Adam provide better overall accuracy (97% compared to 87% for Adadelta), but it also showed a more balanced distribution of performance across all disease categories. This suggests that the model using Adam is more reliable for real-world use in detecting a wide variety of diseases in rice leaves, including those that are difficult to identify visually.

4.2. Realtime Testing

Trial with real-time testing through a laptop webcam on google colab. This trial uses an adam optimizer that has been proven to be better in adam with a model run using epoch 15, and conducted a trial on all diseases and displayed the matrix and indigo of the classification results. The test results are shown in figure 9 of one of the experiment results in real time.

Vol. 6, No. 1, June 2026, Page. 1-16
https://jiki.jurnal-id.com
DOI: https://doi.org/10.54082/jiki.287

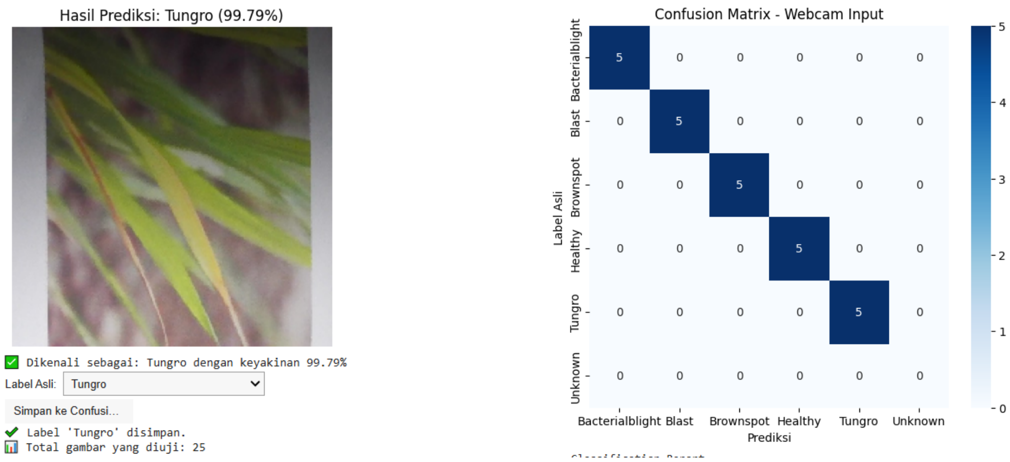

Figure 9. Realtime Test Results

Table 6. Results of Precision Score, Recall and f1-Score

Class	Accuracy	Recall	F1 Score	Support
Bacterialblight	1.00	1.00	1.00	5
Blast	1.00	1.00	1.00	5
Brownspot	1.00	1.00	1.00	5
Healthy	1.00	1.00	1.00	5
Tungro	1.00	1.00	1.00	5
Unknown	0.00	0.00	0.00	0
Accuracy			1.00	25
Macro Avg	0.83	0.83	0.83	25
Weighted Avg	1.00	1.00	1.00	25

The results of figure 9 show the prediction of the CNN VGG16 model against the rice leaf image taken with webcam input. A model with a 99.79 percent confidence rate managed to identify the leaf as Tungro disease according to the original label. As an illustration, the image shows only one example of the prediction result, while the complete evaluation is displayed through a confusion matrix constructed from twenty-five test images, with five images each for the Bacterialblight, Blast, Brownspot, Healthy, and Tungro classes, as well as one unknown class. In the main diagonal, the value is "5" and in the other cell, the value is "0" indicating that the entire test image is correctly identified. This shows that the model can function consistently without misclassification of the test data performed in real-time. Table 6 shows a summary of the results of further evaluation.

In table 6, based on the results of the evaluation in the classification report, the CNN model with the VGG16 architecture shows an extraordinary ability to classify five types of rice leaves: Bacterialblight, Blast, Brownspot, Healthy, and Tungro. Each class had five images for testing, and all were correctly identified. This is indicated by the precision, recall, and f1 scores that reach 1.00 each, which indicates that there are no misclassifications. However, because the Unknown class has no test data, evaluation metrics are not calculated. As a result, the macro average value drops to 0.83, but the weighted average value remains at 1.00 because it takes into account the distribution of data for each class. Overall, the model showed a perfect accuracy of 100% on 25 test data, suggesting that it could quickly find rice leaf disease on the selected dataset.

5. DISCUSSION

This study shows that the CNN VGG16 architecture with the Adam optimizer can produce very optimal classification results. The developed model has a validation accuracy of 98.6% and a test accuracy of 97.17%, and even achieves a perfect accuracy of 100% in real-time testing. This achievement not only demonstrates the model's consistency in identifying new images, but also proves Adam's ability to accelerate the convergence process.

P-ISSN: 2807-6664 https://jiki.jurnal-id.com
E-ISSN: 2807-6591 DOI: https://doi.org/10.54082/jiki.287

These results are much better than previous studies. For example, studies using ResNet-50 only got a test accuracy of 79.52%, while MobileNet got a validation accuracy of 90%, but had overfitting issues. Research conducted by Paneru et al. (2024) using VGG16 for the classification of rice leaf diseases even found that the validation accuracy was only about 84%. Therefore, this study shows that the combination of Adam optimizer, data augmentation strategy, and real-time validation shows a significant improvement in performance. This makes VGG16 more reliable for classifying rice leaf diseases.

There is a clear comparison between the optimizer and the live trial in real-time, which is a great advantage of this study. The results of field tests conducted through webcams show that the model can be used practically. This paved the way for the development of systems that automatically monitor agriculture based on the Internet of Things and drones. However, this study is still limited to datasets with almost identical lighting and backgrounds. Therefore, to make the model more in line with the real world, future research will have to include data from a variety of environments, such as natural lighting, weeds, and partially damaged leaves.

The results of this study are compared with the findings of previous studies that used various architectures and VGG16 in the classification of rice leaf diseases. The results of the comparison are presented in Table 7.

Table 7. Comparison of Research Results

Research	Architecture/Methods	Dataset/Context	Test Accuracy/Validation	Main Notes
This research (2025)	VGG16 + Adam	5 classes (7,460 images + augmentation), real-time test	97.17% (testing) 100% (real-time)	Stable performance, no overfitting, highest accuracy
ResNetNet-50 Research [9]	ResNet-50	6 classes, limited datasets	79.52% (testing)	Low performance, difficult to generalize
MobileNet Research [10]	MobileNet	5 classes, 20 epoches	90% (validation)	There is overfitting, high loss
Pandrachus (15) [IJEEEMI]	VGG16 (transfer learning)	5 classes, small UCI datasets	84% (validation)	The performance is limited, with no real-time

Table 7 shows that this study can achieve a higher level of accuracy than previous studies. MobileNet has a validation accuracy of 90% but is overfitting, while ResNet-50 only produces a test accuracy of 79.52%. The study that also used VGG16 by Paneru et al. (2024) only achieved a validation accuracy of 84%, but this study achieved a test accuracy of 97.17% and a real-time accuracy of 100%. These differences show that Adam's optimizer choice, application of data augmentation, and real-time validation are important factors in improving the performance of VGG16 for detecting rice leaf diseases.

6. CONCLUSION

This study reveals that the CNN algorithm with the design of VGG16 can be applied effectively to detect diseases in rice leaves. From the results of the experiment, it was revealed that the best combination was achieved with the application of the Adam optimizer for 50 epochs, which provided a validation accuracy of 98.6% and a test accuracy of 97.17%. More in-depth analysis using confusion matrix and classification reports showed excellent performance across all categories, with high levels of precision, recall, and F1-score, even reaching 100% in direct testing. This confirms that the developed model is not only useful in the scientific field, but also offers significant practical benefits for farmers in detecting diseases early, thereby supporting increased agricultural production and maintaining food security.

The innovation of this study lies in the incorporation of comparative analysis of the performance of various optimizers with validation based on direct testing, which provides a deeper understanding of the effectiveness of CNN VGG16 in agriculture. These findings confirm CNN's position as one of the leading methods in computer vision for applications in the agricultural sector.

For future development, it is recommended that future research focus on the use of larger and more diverse datasets to improve the model's generalization capabilities. In addition, integration with technologies such as drones or IoT can be a solution to automatically monitor in the field. Research for lighter architectures such as MobileNet or EfficientNet is also important to do so that the system can operate efficiently on edge devices.

BIBLIOGRAPHY

[1] A. Jinan, B. H. Hayadi, and U. P. Utama, "Classification of Rice Plant Diseases Using Convolutional Neural Network Method through Leaf Imagery (Multilayer Perceptron)," *J. Comput. Eng. Sci.*, vol. 1, no. 2, pp. 37–44, 2022.

[2] S. R. Shah *et al.*, "50 : A Case Study on Early Detection of a Rice Disease," pp. 1–13, 2023.

[3] M. Khoiruddin, A. Junaidi, and W. A. Saputra, "Classification of Rice Leaf Disease Using Convolutional Neural Network," *J. Dinda Data Sci. Inf. Technol. Anal Data.*, vol. 2, no. 1, pp. 37–45, 2022, doi: 10.20895/dinda.v2i1.341.

[4] D. MARGARITA, "Classification of Rice Leaf Diseases Using Support Vector Machine Based on Deep Feature," vol. 9, no. 4, pp. 1–21, 2024.

[5] S. Agustiani, Y. T. Arifin, A. Junaidi, S. K. Wildah, and A. Mustopa, "Classification of Rice Leaf Disease using Random Forest and Color Histogram," *J. Computing*, vol. 10, no. 1, pp. 65–74, 2022, doi: 10.23960/computing.v10i1.2961.

[6] A. C. Milano, "Classification of Rice Leaf Disease Using the Efficientnet-B6 Deep Learning Model," *J. Inform. and Tek. Electro Therapy.*, vol. 12, no. 1, 2024, doi: 10.23960/jitet.v12i1.3855.

[7] Dimas Mulya Saputra, Erwin Hermawan, and Sahid Agustian2, "Health Classification in Rice Plants Using Unmanned Aerial Vehicle (UAV) Imagery Using Convolutional Neural Networks (CNN) Method," *J. Ilm. Technology. Therapeutic Information.*, vol. 9, no. 3, pp. 308–318, 2023, doi: 10.33197/jitter.vol9.iss3.2023.1044.

[8] P. Fine *et al.*, "Deep learning-based automatic diagnosis of rice leaf diseases using ensemble CNN models," *Sci. Rep.*, vol. 15, no. 1, pp. 1–13, 2025, doi: 10.1038/s41598-025-13079-z.

[9] P. Pai, S. Amutha, M. Basthikodi, B. M. Ahamed Shafeeq, K. M. Chaitra, and A. P. Gurpur, "A twin CNN-based framework for optimized rice leaf disease classification with feature fusion," *J. Big Data*, vol. 12, no. 1, 2025, doi: 10.1186/s40537-025-01148-z.

[10] S. Suhendar, A. Purnama, and E. Fauzi, "Disease Detection in Sweet Potato Leaves Using Convolutional Neural Network Method," *J. Ilm. Inform. Glob.*, vol. 14, no. 3, pp. 62–67, 2023, doi: 10.36982/jiig.v14i3.3478.

[11] R. Cipta, Khoerunisa, N. M. S. Saraswati, Rizki Noor Prasetyono, and M. Zidan Alfariki, "Detection of Brown, Brown Narrow, and Blight Disease through the Color Spectrum of Digital Rice Leaf Images Using the Convolutional Neural Network Method," *Zo. J. Sist. Inf.*, vol. 5, no. 2, pp. 334–346, 2023, doi: 10.31849/zn.v5i2.13245.

[12] M. Baihaqy and S. Rohman, "BASED ON LEAVES USING CONVOLUTIONAL NEURAL NETWORK (CNN) ALGORITHMS WITH ARCHITECTURE," vol. 2, no. 2, pp. 63–71, 2025.

[13] M. Rifqi Almadani, P. Prajoko, and D. Indriyana, "Classification of Rice Plant Health Using Convolutional Neural Network Algorithm," *JATI (Journal of Mhs. Tek. Inform.*, vol. 8, no. 5, pp. 10177–10182, 2024, doi: 10.36040/jati.v8i5.10844.

[14] E. M. Dogo, O. J. Afolabi, and B. Twala, "On the Relative Impact of Optimizers on Convolutional Neural Networks with Varying Depth and Width for Image Classification," *Appl. Sci.*, vol. 12, no. 23, 2022, doi: 10.3390/app122311976.

[15] B. Paneru, B. Paneru, and K. B. Shah, "i," vol. 6, no. 3, pp. 174–187, 2024.

[16] Y. A. Suwitono and F. J. Kaunang, "Implementation of Convolutional Neural Network (CNN) Algorithm for Leaf Classification Using SEMMA Data Mining Method," *J. Komtika (Computing and Inform.*, vol. 6, no. 2, pp. 109–121, 2022, doi: 10.31603/komtika.v6i2.8054.

[17] M. Rijal, A. M. Yani, and A. Rahman, "Detection of Leaf Imagery for Rice Disease Classification using a Deep Learning Approach with CNN Model," *J. Techno. Integrated*, vol. 10, no. 1, pp. 56–62, 2024, doi: 10.54914/jtt.v10i1.1224.

[18] M. N. Fawaiq, E. Utami, and D. Ariatmanto, "Rice Plant Disease Detection with Data Augmentation Using Transfer Learning," *Int. J. Res. Publ. Rev.*, vol. 4, no. 4, pp. 2195–2199, 2023, doi: 10.55248/gengpi.2023.4.4.35530.

[19] M. T. Roseno, S. Oktarina, Y. Nearti, H. Syaputra, and N. Jayanti, "Comparing CNN Models for Rice Disease Detection: ResNet50, VGG16, and MobileNetV3-Small," *J. Inf. Syst. Informatics*, vol. 6, no. 3, pp. 2099–2109, 2024, doi: 10.51519/journalisi.v6i3.865.
[20] M. L. Alfarisi, M. Herisda, S. A. Kalamsyah, Y. A. Nugraha, S. Salsabila, and M. D. Prasetio, "Evaluation of ANN, CNN, and SVM Models for CIFAR-10 Image Classification," *J. Anal. Digit Computing.*, vol. 9, no. 1, pp. 28–33, 2025, [Online]. Available: https://oaj.jurnalhst.com/index.php/jakd/article/view/9084

Halaman Ini Dikosongkan