

© Jurnal Nasional Teknik Elektro dan Teknologi Informasi
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License
Translation of article 10.22146/nteti.v14i3.18791

Classification of Rice Diseases Using Leaf Image-Based Convolutional Neural Network (CNN)

Moh. Heri Susanto¹, Irwan Budi Santoso¹, Suhartono¹, Ahmad Fahmi Karami¹

¹ Informatics Engineering, Faculty of Science and Technology, State Islamic University of Maulana Malik Ibrahim, Malang, Jawa Timur 65144, Indonesia

[Submitted: 30 January 2025, Revised: 3 April 2025, Accepted: 14 July 2025]

Corresponding Author: Moh. Heri Susanto (email: 200605110087@student.uin-malang.ac.id)

ABSTRACT — Rice diseases significantly impact agricultural productivity, making classification models essential for accurately distinguishing rice leaf diseases. Various classification models have been proposed for image-based rice disease classification; however, further performance improvement is still required. This study proposes the use of a convolutional neural network (CNN) to classify rice diseases based on leaf images. The dataset used in this study included leaf images categorized into four conditions: leaf blight, blast, tungro, and healthy. In the initial stage, data preprocessing was conducted, including resizing, augmentation, and normalization. Following preprocessing, a custom CNN architecture was developed, consisting of four convolutional layers, four pooling layers, and three fully connected layers. Each convolutional layer employed a 3×3 kernel with a stride of 1 and ReLU activation, while the pooling layers used max pooling with a 3×3 kernel and a stride of 2. Using a batch size of 32 and the Adam optimizer, the best test performance was achieved with 100 training epochs and a learning rate of 0.0002, resulting in a training accuracy of 0.9930, a loss of 0.0221, and a test accuracy of 0.9647. Model evaluation demonstrated a balanced performance across precision, recall, and F1 score, each achieving 0.9647, indicating highly effective classification without bias toward any specific class. These findings suggest that the simplified CNN model can deliver competitive classification performance without the need for complex architectures or additional enhancement techniques. The proposed CNN model outperformed existing CNN architectures, such as Inception-ResNet-V2, VGG-16, VGG-19, and Xception.

KEYWORDS — Rice Disease, Custom CNN Architecture, Leaf Image, Adam Optimization, Model Evaluation.

I. INTRODUCTION

Rice diseases pose a significant threat to agricultural productivity, necessitating accurate classification methods to distinguish between different types of rice diseases. Several major rice diseases that affect crop yields include leaf blight, blast, and tungro. Leaf blight is caused by the bacterium *Xanthomonas oryzae* pv. *oryzae* (Xoo) and is characterized by elongated lesions that vary in color from yellow to brown. These lesions typically appear on newly developed leaves and may merge over time, resulting in a scorched appearance [1]. Blast, caused by the fungus *Pyricularia grisea*, leads to shriveled panicles and partially filled or empty rice grains. Symptoms of this disease include grayish-white leaf spots with slightly orange-brown margins, shaped like diamonds with pointed edges [2]. Tungro, transmitted by the green planthopper, causes rice plants to turn yellow to orange, exhibit stunted growth, and suffer from reduced grain production [3]. In contrast, healthy rice plants show no signs of disease. Figure 1 illustrates the visual characteristics of the various rice disease types.

Several previous studies have employed classification methods such as decision tree, artificial neural network (ANN), convolutional neural network (CNN), naïve Bayes, k-nearest neighbors (KNN), and support vector machine (SVM). Although each method has contributed significantly to the field, limitations remain, which may stem from various factors such as the nature of the data or the superiority of the model, or its architecture. Therefore, research on rice disease classification still holds potential for further development. Continued refinement is necessary to improve the accuracy and performance of classification methods in order to ensure more reliable disease categorization.

Previous studies on rice disease classification have mainly focused on image data of healthy leaves and brown spots [4]. The proposed methods included gray-level co-occurrence matrix (GLCM) and intensity-level based multi-fractal dimension (ILMFD), followed by classification using SVM, ANN, and neuro-genetic algorithm (Neuro-GA) for comparison. The results indicated that Neuro-GA outperformed ANN; however, SVM proved superior to Neuro-GA, achieving an accuracy of 96.68% [4].

Another study utilized a variation of a dataset available from the UCI Machine Learning Repository, consisting of bacterial leaf blight, brown spots, and leaf smut [5]. ANN was used in the classification process with 100 epochs, achieving an accuracy of 83%, precision of 84%, recall of 84%, and an F1 score of 83% [5]. This study, however, had certain limitations, including a relatively small dataset of only 120 samples and the absence of data augmentation processes.

Using the same dataset, another study reported that the proposed method—transfer learning using a pretrained ResNet101—achieved a validation accuracy of 100% with a loss value of 5.61% [6]. This validation performance indicates that the model was highly effective in learning patterns from the validation dataset. However, since the study was conducted only up to the validation stage and did not include testing, it cannot yet be confirmed whether the model can generalize effectively to unseen data. The testing stage is essential in research to properly evaluate a model's generalization capability.

Another study also focused on classifying rice diseases, specifically leaf blight, brown spots, and leaf smut [7]. Using GLCM for feature extraction and KNN as the classification model, the study reported a testing accuracy of 65.83% [7].



Figure 1. Sample images of rice leaf diseases, (a) leaf blight, (b) blast, and (c) tungro.

This result demonstrates that the KNN method still lacks the capability to achieve high classification performance.

In addition, several researchers have also conducted studies using the decision tree method [8]. In this study, features were extracted through segmentation and morphological operations. The testing accuracy achieved was 90% [8]. However, the dataset used consisted of only 120 samples, and the decision tree method is not effective in handling complex data.

Another study employed the naïve Bayes model to classify rice diseases such as bacterial leaf blight, blast, and tungro. The model achieved an accuracy of 76% and a confidence level of 100% for the bacterial leaf blight class [9]. The results showed that the model performance was suboptimal and that augmentation techniques that were not carried out limited its ability in dealing with data diversity.

Furthermore, another study utilized a dataset consisting of tungro, blast, leaf blight, and brown spot classes, employing a CNN approach. With a total dataset of 320 images, an accuracy of 91.4% was achieved [10]. However, the relatively small dataset and the absence of overfitting mitigation techniques such as data augmentation or cross-validation presented significant limitations in this study. In addition, CNN itself offers vast potential for architectural improvements, including the number of hidden layers, number of epochs, learning rate, batch size, kernel size, loss function, optimization methods, and dataset expansion.

Relevant studies have also demonstrated that increasing dataset diversity and size contributes to improved model performance [11]. With a training dataset of 4,000 images and a testing dataset of 300 images, an accuracy of 95.67% was achieved. This study utilized the InceptionResNetV2 architecture and transfer learning methods [11]. Despite the

high accuracy, the dataset used exhibited imbalanced background characteristics. The blast and bacterial leaf blight classes had overlapping backgrounds, while the brown spot and healthy classes consisted of single rice leaves with plain white backgrounds. This condition may lead to model overfitting, where the model focuses more on background features than on the actual leaf characteristics.

In response to these challenges, this study proposes the use of CNN to classify rice diseases based on leaf images. Leaf imagery was chosen as rice disease symptoms typically manifest initially on the leaves in the form of discoloration and spots before spreading to other parts, such as the stem or panicle. CNN is selected due to its ability to effectively learn and extract key features automatically, eliminating the need for manual feature extraction, as the convolutional layers inherently perform feature extraction. It has also been shown that in image classification tasks, CNN outperforms traditional machine learning methods such as SVM [12].

The proven success of CNN in image-based object classification and its superiority over other machine learning techniques make it an ideal choice for achieving robust and reliable performance in rice disease diagnosis. It is important to clarify that this study focuses on classification within the domain of intelligent systems, without addressing real-time implementation or automation in agricultural environments. This research contributes to the fundamental aspects of intelligent systems, which can later be applied in broader smart agriculture applications. The objective of this study is to develop a CNN architecture specifically designed for rice disease classification, ensuring that the model achieves competitive performance compared to existing classification models.

Although previous studies may have achieved promising levels of accuracy, there is always room for improvement. This capacity positions CNN as a powerful tool for classifying rice leaf diseases based on visual characteristics.

Based on the above discussion, this study presents several key contributions. First, it optimizes a simple CNN architecture for rice disease classification, aiming to enhance accuracy and performance without increasing model complexity. Second, the study provides a detailed evaluation of performance metrics, including accuracy, precision, recall, and F1 score to assess the model's ability to distinguish between different rice diseases. Third, a comparative performance analysis is conducted between the proposed CNN model and other CNN-based models, offering insights into the effectiveness of the proposed approach.

II. METHODOLOGY





A. DATA COLLECTION

In this study, a rice disease dataset based on leaf images was collected from Kaggle. This dataset was selected because the data have been verified and validated by Kaggle, making them suitable for further analysis, such as classification. The dataset comprised 220 images of rice leaves affected by bacterial leaf blight, 200 images of blast, 80 images of tungro, and an additional 210 images of healthy leaves, resulting in a total of 710 images. Table I presents several sample images used in this study.

B. DATA TRAINING AND DATA TESTING SCHEME

The rice disease dataset was divided into two primary groups: training data and testing data. A total of 80% of the

TABLE I
EXAMPLE IMAGES FROM THE DATASET USED

No.	Rice Leaf Condition	Images
1.	Blast	
2.	Leaf blight	
3.	Tungro	
4.	Healthy	

dataset was allocated for model training. This training stage enables the model to learn and recognize relevant patterns or features in the data. The training process involved resizing, augmentation, and normalization. Subsequently, a CNN model was constructed to obtain new weights and biases. These learned weights and biases were then stored for use during the testing stage.

After the model training was completed, the remaining 20% of the dataset was used as testing data. The testing process also included resizing and normalization, similar to the training stage. The preprocessed testing data were then fed into the CNN using the previously learned weights and biases to classify the input and produce the final classification results, indicating the type of rice condition.

C. PREPROCESSING

Image preprocessing is a crucial step in preparing data for the CNN model. It involves techniques such as resizing and augmentation. Since color features play an important role in classification, no color conversion is applied.

The first step was image resizing to ensure uniformity across the dataset. This is necessary because CNNs require input images of consistent dimensions. If image sizes vary, the CNN model is not able to process them properly. In this study, all images were resized to 64×64 pixels. This resolution was selected because it is relatively small, thereby reducing computational burden.

Next, data augmentation was applied to enhance training data diversity and improve model generalization [13]. This was done through rotation, enabling the model to learn object features from different orientations. Additional augmentation techniques such as flipping, shifting, shearing, and zooming were also implemented. These augmentations allow the model to recognize objects from various perspectives, enrich the data

representation, and reduce the risk of overfitting. Such techniques support the model's robustness during testing and contribute to improved accuracy through the availability of abundant data variations.

Finally, normalization was applied to both training and testing data by scaling pixel values to the range of 0 to 1. This step ensures consistency and enhances the efficiency of model training. Normalization also helps to prevent large gradients, which can slow down the training process.

D. PROPOSED CNN ARCHITECTURE

Figure 2 illustrates the proposed CNN architecture for classifying rice diseases. This architecture is designed to optimize feature detection in 64×64 -pixel images and consists of four hidden layers. The first hidden layer contains a convolutional layer (denoted as K_a) utilizing eight 3×3 filters, with three input channels corresponding to the RGB color channels. The output from this convolutional layer has dimensions of $62 \times 62 \times 8$, based on a 'no padding' convolution calculation. Following convolution, the data are processed using a rectified linear unit (ReLU) activation function to address nonlinearity. Subsequently, a max pooling layer (M_{pa}) with a 2×2 filter is applied, resulting in an output size of $31 \times 31 \times 8$.

The second hidden layer contains a convolutional layer (K_b) that employs sixteen 3×3 filters. The input consists of eight channels obtained from the output of the previous hidden layer. The output dimensions of this convolutional layer are $29 \times 29 \times 16$, also calculated using 'no padding'. ReLU is applied for activation. A 2×2 max pooling layer follows, producing an output of $14 \times 14 \times 16$.

The third hidden layer includes a convolutional layer (K_c) with thirty-two 3×3 filters, and sixteen input channels derived from the previous layer's output. Using 'no padding', the convolution output has dimensions of $12 \times 12 \times 32$. ReLU is used as the activation function. This is followed by a max pooling layer with a 2×2 filter, resulting in an output of $6 \times 6 \times 32$.

The final hidden layer contains a convolutional layer (K_d) with sixty-four 3×3 filters and thirty-two input channels from the preceding layer. Using the 'no padding' rule, the resulting output is $4 \times 4 \times 64$. The ReLU activation function is again employed. A subsequent max pooling operation with a 2×2 filter yields the final output of $2 \times 2 \times 64$.

Following the completion of the feature extraction process, the resulting multidimensional data are flattened into a one-dimensional array using a flatten layer. This transformation enables the data to be processed by the fully connected layer. The output of this stage is in the form of raw data or logits. Subsequently, a dropout technique with a rate of 0.5 is applied to mitigate overfitting. Finally, the softmax layer converts the raw data into classification probabilities, allowing the model to determine the type of rice disease by selecting the class with the highest probability based on input images of low resolution. The details of each step are explained as follows.

1) CONVOLUTIONAL LAYER

In the proposed architecture, the convolutional layer is responsible for extracting features from the input data, which consists of images of rice leaves [14]. This layer operates by applying a set of filters to the input image to perform feature extraction. The extracted features include texture, color, and pattern shapes present on the rice leaf images. The computation

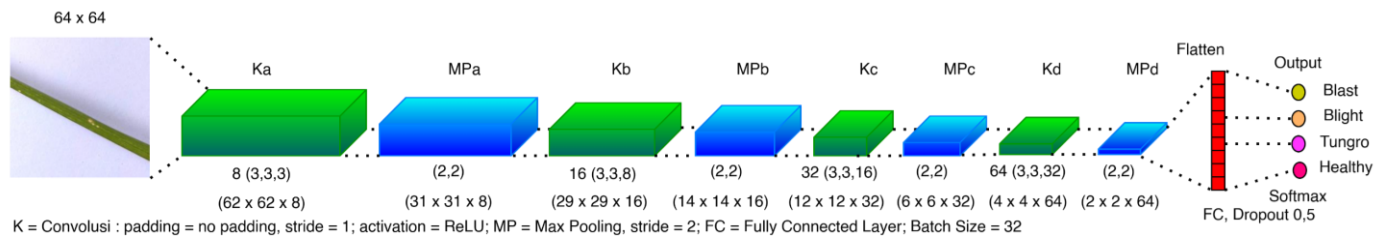


Figure 2. Proposed CNN architecture.

process in the convolutional layer is mathematically represented by (1) [15].

$$Y(p, q) = \sum_i \sum_j X(p, q) * F(p - i, q - j) \quad (1)$$

where $Y(p, q)$ denotes the output of the convolution operation at coordinate (p, q) , $X(p, q)$ is the input image at coordinate (p, q) , and $F(i, j)$ is the convolutional filter or kernel at position (i, j) . The variables p and q represent output coordinates, while i and j represent the spatial indices of the filter. Each filter generates one feature map upon convolving with the input. Therefore, if a convolutional layer contains eight filters, it will produce eight corresponding feature maps.

During the convolutional layer operation, stride and padding play critical roles in determining the feature extraction process from rice leaf images. In this study, a stride value of 1 was used, ensuring that the convolutional filter shifts by one pixel at each step to the next position. Meanwhile, no padding was applied, as the purpose is to reduce the input dimensions (i.e., height and width). By omitting padding, the model decreases computational load while retaining features relevant to rice disease classification. Equations (2) and (3) present the formulas used to determine the output dimensions of the convolution operation without padding, as applied in this study according to the standard convolution rule [16]:

$$W_2 = \frac{W_1 - F_1}{S} + 1 \quad (2)$$

$$H_2 = \frac{H_1 - F_2}{S} + 1 \quad (3)$$

where W_1 and H_1 are the width and height of the input matrix, F_1 and F_2 are the width and height of the filter/kernel, and S is the stride or kernel steps applied to the input data.

2) RECTIFIED LINEAR UNIT (ReLU)

The ReLU activation function was employed to effectively handle the nonlinearity present in rice disease image data. The ReLU function transforms negative values into zero and retains positive values, as defined in (4) [17].

$$\hat{y}_a(p, q) = \begin{cases} 0 & Y(p, q) < 0 \\ Y(p, q) & Y(p, q) \geq 0 \end{cases} \quad (4)$$

where $Y(p, q)$ is the output from the convolutional layer and $\hat{y}_a(p, q)$ is the output from the ReLU layer. By discarding negative values, ReLU ensures that the model retains only relevant disease features on the rice leaves. This selective activation enhances the model's ability to focus on distinguishing features between healthy and diseased leaves.

3) POOLING LAYER

After processing through the ReLU activation, the data proceeded to the pooling layer, which reduced the spatial dimensions of the feature maps. This step lowers the computational burden and decreases the parameter complexity

of the network [18]. As illustrated in Figure 2, the type of pooling used in this study is max pooling, which selects the maximum value from each feature map region. This method preserves the most significant and dominant features of diseased rice leaves while eliminating less informative data. A stride value of 2 was applied, meaning the filter moves two pixels at a time, effectively reducing the feature map size. The output dimensions are computed using (5) and (6) [19].

$$W_2 = \frac{W_1 - F_1}{S} \quad (5)$$

$$H_2 = \frac{H_1 - F_2}{S} \quad (6)$$

4) FLATTEN LAYER

Following the pooling operation in the final hidden layer, the data were processed by a flatten layer, which reshaped the multidimensional data into a one-dimensional vector. This step is crucial to ensure that the extracted features can be passed into the fully connected layer for further processing.

5) FULLY CONNECTED LAYER

The fully connected layer formed a critical component of the model architecture in this study. At this stage, the model began to identify specific disease patterns based on the extracted features, which were then processed and combined to produce raw outputs (logits). After each fully connected layer, a ReLU activation and a dropout with a rate of 50% (0.5) were applied. The dropout technique serves to prevent overfitting [6] by randomly deactivating 50% of neurons during training. This regularization method helped avoid dependency on specific neurons and enhanced the model's generalization ability to new rice leaf images, ultimately improving classification accuracy.

6) SOFTMAX AND NEGATIVE LOG-LIKELIHOOD (NLL) LOSS

At this stage, the softmax function was employed to transform the raw data (logits) from the fully connected layer into a probability distribution, ensuring that all values are nonnegative and the total probability sums to one. Equation (7) defines the softmax output [20].

$$S(\hat{y}_{ak}) = \frac{e^{\hat{y}_{ak}}}{\sum_{i=1}^K e^{\hat{y}_{ik}}} \quad (7)$$

where S represents the softmax output for the k th class, \hat{y}_{ak} denotes the output value from the fully connected layer, e is Euler's number $\cong 2.718$, and K is the total number of classes.

Following the probability distribution computation, the loss is calculated using negative log likelihood (NLL) loss. NLL loss is conceptually similar to cross-entropy loss, with the key difference being that cross-entropy computes the softmax and the loss separately, while NLL implicitly includes the softmax operation. NLL penalizes incorrect predictions [20], as defined in (8).

$$E = -\hat{y} \log \hat{y}_{ak} \quad (8)$$

where \hat{y} is the ground-truth label (1 for correct, 0 for incorrect), and \hat{y}_{ak} is the predicted probability for the k th class. A lower NLL output indicates a better prediction, signifying that the model has effectively learned to identify disease patterns in the rice leaf images.

7) BACKPROPAGATION AND GRADIENT CALCULATION

The computed loss was then used in the backpropagation process to update the model's weights and biases. This study employed the Adam optimizer for this purpose. Adam optimization is a method used for updating weights and biases [21]. It is an adaptive learning rate optimization algorithm specifically designed for training deep neural networks [22], combining the advantages of RMSprop and momentum. The formulas for updating the weights are given in (9) to (12) [23].

$$m_w = \beta_1 * m_{w-1} + (1 - \beta_1) * \left(\frac{\partial E}{\partial w}\right) \quad (9)$$

$$\frac{\partial E}{\partial w} = \delta \cdot x \quad (10)$$

$$v_w = \beta_2 * v_{w-1} + (1 - \beta_2) * \left(\frac{\partial E}{\partial w}\right)^2 \quad (11)$$

$$w = w - \left(\frac{\alpha * m_w}{\sqrt{(v_w + \epsilon)}}\right). \quad (12)$$

The bias terms are updated using (13) to (16).

$$m_b = \beta_1 * m_{b-1} + (1 - \beta_1) * \left(\frac{\partial E}{\partial b}\right) \quad (13)$$

$$\frac{\partial E}{\partial b} = \delta \quad (14)$$

$$v_b = \beta_2 * v_{b-1} + (1 - \beta_2) * \left(\frac{\partial E}{\partial b}\right)^2 \quad (15)$$

$$b = b - \left(\frac{\alpha * m_b}{\sqrt{(v_b + \epsilon)}}\right). \quad (16)$$

In these equations, m_w and m_b represent the momentum terms for the weights and biases, respectively, while v_w and v_b represent the velocity terms. β_1 is the parameter for the first moment (typically set to 0.9), and β_2 is the parameter for the second moment (typically set to 0.999). $\frac{\partial E}{\partial w}$ and $\frac{\partial E}{\partial b}$ denote the gradients of the loss function concerning the weights and biases. m_{w-1} and m_{b-1} are the previous momentum values, while v_{w-1} and v_{b-1} are the previous velocity values. The variables w and b refer to the updated weights and biases, respectively. α is the learning rate, ϵ is a small constant (typically 10^{-7}) to prevent division by zero, δ is the error signal (loss) at the output neuron, and x is the input associated with the corresponding weight in the neural network.

The backpropagation process continued until the specified number of epochs was reached. Upon completion, the updated weights and biases were saved and used in the testing phase.

8) EVALUATION

The model's performance on the test dataset was assessed through evaluation metrics. The evaluation included accuracy, precision, recall, and F1 score, shown in (17) until (20), respectively. The accuracy measures how frequently the model correctly predicts the test data as a whole. A value closer to 1 indicates better performance, while a value closer to 0 suggests suboptimal performance. Precision measures the proportion of correctly predicted positive samples out of all predicted positives. Recall measures the proportion of actual positives

correctly predicted by the model. F1 score represents the harmonic mean of precision and recall.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (17)$$

$$Precision = \frac{TP}{TP+FP} \quad (18)$$

$$Recall = \frac{TP}{TP+FN} \quad (19)$$

$$F1 \text{ score} = 2 \frac{Recall \times Precision}{Recall + Precision} \quad (20)$$

To provide a more detailed overview of model performance, a confusion matrix is also utilized. Table II summarizes the classification performance in terms of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN).

For example, in the case of leaf blight prediction, a TP occurs when the model correctly predicts leaf blight. FN includes all entries in the leaf blight row that are not TP. FP includes entries in the leaf blight column that are not TP. TN consists of all entries outside the leaf blight row and column. The same principles apply to predictions in other classes.

III. RESULTS AND DISCUSSION

A. TESTING RESULTS

The initial stage of this study involved a series of experimental schemes to identify the most optimal outcome. These trials were conducted by varying the number of epochs and learning rates. The variation in epochs refers to the number of complete passes through the training dataset, while the learning rate variation determines the step size in updating the weights during the training process. The purpose of these variations was to identify the most effective parameter combination to enhance model performance.

The experiments were conducted in stages, from A1 to B4. Models A1 to A4 were evaluated using 30 epochs and learning rates of 0.001, 0.0001, 0.0002, and 0.0003, respectively. A learning rate of 0.001 was chosen for its relatively low value, while the others served as comparative benchmarks. Models B1 to B4 were trained using 100 epochs, with the same range of learning rates. The increase in epochs was intended to assess the effect of this parameter on model performance.

Upon completion of these experiments based on the designed schemes, the results from trial A1 (30 epochs, learning rate 0.001) showed a training accuracy of 0.9018 and a relatively low loss of 0.2258. The training duration was approximately $112^m0.8^s$. The testing accuracy reached 0.9295, which was higher than the training accuracy, indicating potential underfitting. This suggests that the model might not have captured the complex features in the training data but still managed to generalize well on the test data.

In trial A2 (30 epochs, learning rate 0.0001), the training accuracy was 0.8305 with a loss of 0.3880, and a training duration of $128^m0.8^s$. The testing accuracy was 0.8802. These results indicate that reducing the learning rate leads to a longer training time. This is because smaller step sizes cause the model to take longer to reach an optimal point. The model's training and testing performance in this experiment were suboptimal compared to trial A1.

Trial A3 (30 epochs, learning rate 0.0002) yielded an improved training accuracy of 0.9234, a reduced loss of 0.1921, and a training time of $112^m6.4^s$. The testing accuracy also

TABLE II
CONFUSION MATRIX

Actual	Prediction Class			
	<i>Blight (A)</i>	<i>Blast (B)</i>	<i>Tungro (C)</i>	<i>Healthy (D)</i>
Blight (A)	AA	AB	AC	AD
Blast (B)	BA	BB	BC	BD
Tungro (C)	CA	CB	CC	CD
Healthy (D)	DA	DB	DC	DD

improved to 0.9436. These results indicate that a slightly increased learning rate enabled better model performance. The findings suggest that the model benefits from a learning rate higher than in A2 but lower than in A1, allowing for faster and more effective convergence without increasing the loss. However, underfitting was still observed in this trial.

In trial A4 (30 epochs, learning rate 0.0003), overall model performance declined. The training accuracy was 0.9179, with a loss of 0.2069 and a training duration of $112^m3.7^s$. The testing accuracy dropped to 0.9084. Despite the relatively high training accuracy, the decreased testing accuracy compared to A1 and A3 suggests overfitting. The model performed well during training, as shown by the small gap between training and testing accuracy.

Trial B1 (100 epochs, learning rate 0.001) achieved a training accuracy of 0.9850 and a loss of 0.0393, which was lower than in previous trials. The training duration increased significantly to $364^m8.7^s$ due to the extended number of epochs. The testing accuracy was 0.9154, indicating overfitting, as the model may have memorized the training data excessively. This trial demonstrates that increasing the number of epochs, which results in longer training time, does not necessarily enhance model performance.

In trial B2 (100 epochs, learning rate 0.0001), the training accuracy decreased to 0.9421, and the loss increased to 0.1366. The training duration remained at $364^m8.7^s$. The testing accuracy improved slightly to 0.9225, showing that overfitting persisted but was less severe than in B1. The model was better able to generalize to the testing data compared to B1.

Trial B3 (100 epochs, learning rate 0.0002) resulted in the highest training accuracy observed, at 0.9930. The loss decreased significantly to 0.0221, and the testing accuracy increased markedly to 0.9647, indicating excellent generalization capability. The training duration was $363^m10.8^s$, shorter than that of B1 and B2.

Finally, in trial B4 (100 epochs, learning rate 0.0003), the model achieved a training accuracy of 0.9910 and a loss of 0.0246, with a training time of approximately $359^m23.2^s$. The testing accuracy was 0.9225, indicating overfitting. This parameter combination was found to be suboptimal, as the model failed to achieve effective convergence and lacked the ability to learn properly from the training data.

Table III presents the complete results of all experiments for easier analysis. As shown in Table III, models trained with 100 epochs generally produced better accuracy and lower loss compared to those trained with 30 epochs. This is because more epochs provide the model with extended time to learn from the data, allowing it to achieve lower loss.

Across experiments A1 to A4 and B1 to B4, the highest accuracies were consistently achieved with a learning rate of 0.0002. This indicates that a moderately small learning rate supports more stable learning and results in lower loss. However, a learning rate lower than 0.0002, such as 0.0001, did

TABLE III
RESULTS OF OVERALL EXPERIMENTS

Experiment	Training Accuracy	Loss	Training Duration	Testing Accuracy
A1	0.9018	0.2258	$112^m0.8^s$	0.9295
A2	0.8305	0.3880	$128^m7.8^s$	0.8802
A3	0.9234	0.1921	$112^m6.4^s$	0.9436
A4	0.9179	0.2069	$112^m3.7^s$	0.9084
B1	0.9850	0.0393	$377^m24.3^s$	0.9154
B2	0.9421	0.1366	$364^m8.7^s$	0.9225
B3	0.9930	0.0221	$363^m10.8^s$	0.9647
B4	0.9910	0.0246	$359^m23.2^s$	0.9225

not lead to better performance. This may be due to the learning rate being too small, thereby slowing down the learning process and hindering the model's ability to converge effectively.

The best performance was achieved in trial B3, which used 100 epochs and a learning rate of 0.0002. This configuration resulted in a training accuracy of 0.9930, a loss value of 0.0221, and a testing accuracy of 0.9647. The variations in both epoch and learning rate significantly influenced the model's performance, highlighting the importance of parameter tuning in optimizing the CNN model for rice disease classification.

Figure 3(a) illustrates the training accuracy graph for trial B3. It can be observed that at the beginning of the training, accuracy increased rapidly, indicating that the model was able to learn the patterns in the training data effectively. However, the model exhibited instability at several points during the process. This instability suggests that the model encountered difficulties in adapting to complex data, possibly due to the emergence of unseen features. Despite these fluctuations, the model ultimately achieved its highest accuracy by the final epoch.

Figure 3(b) presents the training loss graph for trial B3. Initially, the loss value was close to 1, which can be attributed to the random initialization of weights, causing the model to be poorly adjusted to the data at the start. As training progressed, the loss decreased steadily and approached zero, indicating that the model successfully adjusted its weights to fit the training data. This suggests that prediction errors were minimized as training continued.

B. DISCUSSION

In addition to accuracy, several other evaluation metrics were used to assess model performance, namely precision, recall, and F1 score. Figure 4 presents the evaluation metrics corresponding to the best-performing model (B3).

In the context of this study, recall is considered more critical than precision and F1 score, as it measures the proportion of actual positive cases that are correctly identified by the model. This implies that recall is essential to ensure the model successfully detects all diseased rice leaves.

In agricultural applications, failing to identify infected rice leaves poses a greater threat than misclassifying healthy leaves. Therefore, recall becomes the primary metric to minimize the risk of undetected diseased leaves.

Figure 4 shows that precision, recall, and F1 score all achieved the same value. This indicates that the model is capable of classifying data effectively without bias toward any particular class. High precision signifies that the model rarely misclassifies healthy leaves as diseased. High recall demonstrates that the model is largely successful in identifying

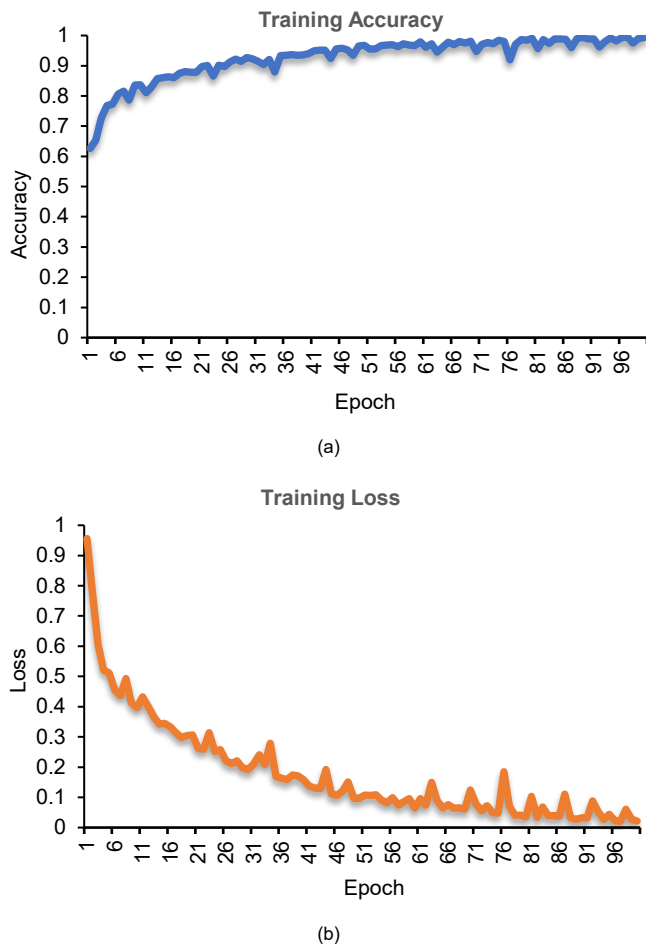


Figure 3. Graphs showing the best-performing trial (B3), (a) training accuracy, (b) training loss.

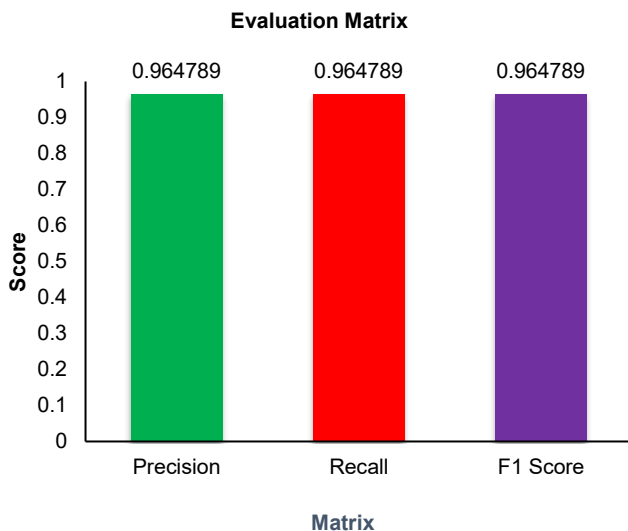


Figure 4. Evaluation matrix for the B3.

infected leaves. Meanwhile, a high F1 score reflects the model's strong performance in avoiding prediction errors for both healthy and diseased leaves. The similarity of accuracy, precision, recall, and F1 score values further confirms that the model performs optimally across all classes.

In addition, to obtain a detailed understanding of the prediction distribution, a confusion matrix was used, as shown in Figure 5. The diagonal values indicate correct classification,

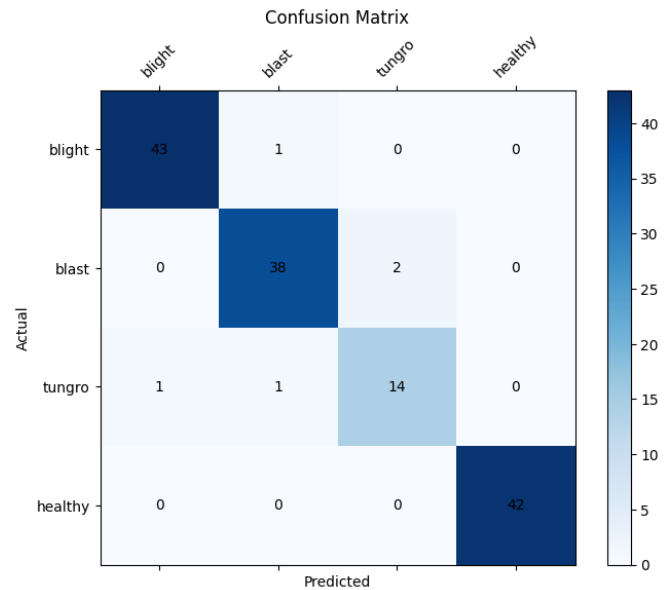


Figure 5. Confusion matrix results for B3.

with 43 leaf blight, 38 blast, 14 tungro, and 42 healthy samples correctly predicted. However, some errors occurred, such as 1 leaf blight sample being classified as blast, 2 blast samples as tungro, 1 tungro sample as leaf blight, and 1 tungro sample as blast. The color intensity in the matrix visually differentiates between high and low values, helping to assess the model's performance in distinguishing between various rice diseases and healthy conditions.

Several previous studies serve as benchmarks for this research, utilizing different CNN variants, as summarized in Figure 6. The Inception-ResNet-V2 model with a transfer learning approach employed a pretrained model with adjustments made to the final layers, resulting in a short training duration and a test accuracy of 92.68% [24]. VGG-16 with transfer learning utilizing pretrained weights from ImageNet achieved an accuracy of 92.46% [25]. VGG-19, an improved version of VGG-16, demonstrated better performance with an accuracy of 95.24% [26]. The Xception model, utilizing depthwise separable convolution, provided efficient computation and good accuracy at 89% [27]. In comparison, the proposed method, achieving an accuracy of 96.47%, offers competitive performance while maintaining architectural simplicity.

Overall, the proposed CNN model exhibits an accuracy improvement of approximately 1.23%. One contributing factor to this improvement is the alignment between the model architecture and the characteristics of the dataset. The custom CNN architecture proposed in this study was designed with careful consideration of several aspects, including the number of hidden layers, the use of padding in convolutional layers, the selection of kernel size and quantity, activation function choice, pooling layer technique, optimization method, loss function, overfitting mitigation strategies, number of epochs, learning rate, and batch size. By employing a well-suited architecture, the model can more effectively learn complex features from the data, overcome overfitting or underfitting issues, and enhance accuracy. Additionally, preprocessing played a vital role in improving model accuracy, resulting in better performance compared to previous CNN-based models that may not have been optimally tuned.

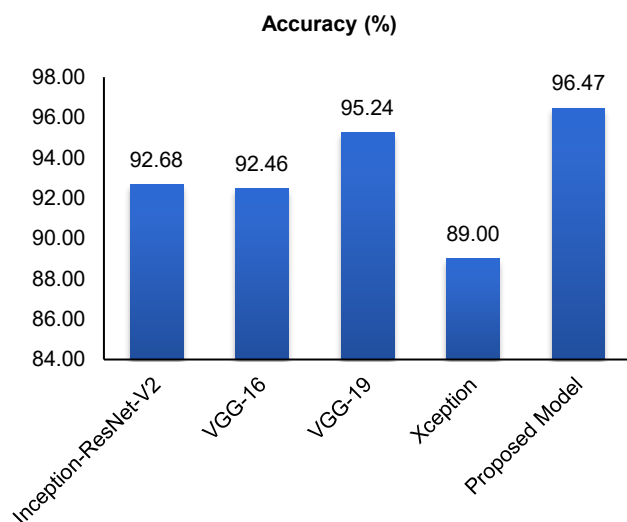


Figure 6. Comparison with Existing CNN Models.

IV. CONCLUSION

In this study, the implemented CNN model has successfully classified infected and noninfected leaves with high accuracy. The test accuracy, precision, recall, and F1 score all achieved the same value of 0.96. Prediction errors were minimal, with only five misclassifications recorded. Notably, all samples in the healthy class were correctly predicted. These results demonstrate that the CNN architecture employed in this approach is capable of achieving improved accuracy and competitive performance compared to existing CNN classification models, despite its relatively simple structure.

As this study primarily focused on the fundamental construction of a CNN model, several enhancements can be explored in future research for further optimization. First, it is recommended to investigate the integration of additional techniques such as transfer learning, fine-tuning, or other methods that may improve model performance. A limitation of this study is the exclusive use of a simple CNN architecture without the incorporation of such auxiliary techniques. This approach was intentionally chosen to evaluate the basic capability of CNNs in recognizing visual patterns from rice leaf images directly, allowing the results to reflect the intrinsic effectiveness of the CNN architecture itself without external influence.

Moreover, the study utilized a limited dataset obtained from publicly available sources. Expanding the dataset with more diverse and high-quality images could enhance the model's generalization capability. Additionally, exploring hybrid approaches that combine CNNs with other classification methods may offer improved feature extraction and performance outcomes.

CONFLICTS OF INTEREST

The authors affirm that they have no conflicts of interest to declare regarding this study.

AUTHORS' CONTRIBUTIONS

Conceptualization, Moh. Heri Susanto and Irwan Budi Santoso; methodology, Moh. Heri Susanto; software, Moh. Heri Susanto; validation, Moh. Heri Susanto, Irwan Budi Santoso, Suhartono, and Ahmad Fahmi Karami; formal analysis, Moh. Heri Susanto, Irwan Budi Santoso, and

Suhartono; investigation, Moh. Heri Susanto; data curation, Moh. Heri Susanto; writing-original drafting, Moh. Heri Susanto, and Irwan Budi Santoso; writing-reviewing and editing, Suhartono and Ahmad Fahmi Karami.

ACKNOWLEDGMENTS

The authors would like to thank the Department of Informatics Engineering, State Islamic University of Maulana Malik Ibrahim, for the support and guidance provided throughout this research.

REFERENCES

- [1] P.M. Giopany, "Penyakit hawar daun bakteri pada tanaman padi." Distan. Access date: 15-Mar-2025. [Online]. Available: https://distan.bulelengkab.go.id/informasi/detail/artikel/55_penyakit-hawar-daun-bakteri-pada-tanaman-padi?
- [2] V. Silvana, "Mengenal penyakit blas pada tanaman padi." Distan. Access date: 15-Mar-2025. [Online]. Available: https://distan.bulelengkab.go.id/informasi/detail/artikel/57_mengenal-penyakit-blas-pada-tanaman-padi?
- [3] S. Sipi and Subiadi, "Penyakit tungro dan keracunan Fe pada tanaman padi." Access date: 15-Mar-2025. [Online]. Available: <https://repository.pertanian.go.id/server/api/core/bitstreams/6eb9a1de-0c26-4d49-9534-203a469b8f24/content>
- [4] S. Chaudhary and U. Kumar, "Automated detection and classification of rice crop diseases using advanced image processing and machine learning techniques," *Trait. Signal*, vol. 41, no. 2, pp. 739–752, Apr. 2024, doi: 10.18280/ts.410216.
- [5] N. Sunandar and J. Sutopo, "Utilization of artificial neural network in rice plant disease classification using leaf image," *Int. J. Res. Sci. Eng.*, vol. 4, no. 2, pp. 1–10, Feb./Mar. 2024, doi: 10.55529/ijrise.42.1.10.
- [6] U.N. Oktaviana, R. Hendrawan, A.D.K. Annas, and G.W. Wicaksono, "Klasifikasi penyakit padi berdasarkan citra daun menggunakan model terlatih Resnet101," *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 5, no. 6, pp. 1216–1222, Dec. 2021, doi: 10.29207/resti.v5i6.3607.
- [7] R.A. Saputra *et al.*, "Rice leaf disease image classifications using KNN based on GLCM feature extraction," in *Int. Conf. Adv. Inf. Sci. Dev. (ICAISD)*, 2020, pp. 1–6, doi: 10.1088/1742-6596/1641/1/012080.
- [8] A.D.P. Alwy, M.S.N. Wahid, B.N.N. Ag, and M.M. Fakhri, "Klasifikasi penyakit padi dengan ekstraksi fitur LBP dan GLCM," *J. Deep Learn. Comput. Vis. Digit. Image Process.*, vol. 1, no. 1, pp. 1–10, Mar. 2023, doi: 10.61255/decoding.v1i1.51.
- [9] Rahmi and R.A. Saputra, "Klasifikasi Penyakit padi melalui citra daun menggunakan metode naive Bayes," *J. Inform. Tek. Elektro Terap.*, vol. 12, no. 2, pp. 885–892, Apr. 2024, doi: 10.23960/jitet.v12i2.4012.
- [10] A.K. Abasi *et al.*, "Enhancing rice leaf disease classification: A customized convolutional neural network approach," *Sustainability*, vol. 15, no. 20, Oct. 2023, Art. no 15039, doi: 10.3390/su152015039.
- [11] Krishnamoorthy N *et al.*, "Rice leaf diseases prediction using deep neural networks with transfer learning," *Environ. Res.*, vol. 198, pp. 1–8, Jul. 2021, doi: 10.1016/j.envres.2021.111275.
- [12] I.W.S.E. Putra, A.Y. Wijaya, and R. Soelaiman, "Klasifikasi citra menggunakan convolutional neural network (CNN) pada Caltech 101," *J. Tek. ITS*, vol. 5, no. 1, pp. 65–69, 2016, doi: 10.12962/j23373539.v5i1.15696.
- [13] R. Jain *et al.*, "Pneumonia detection in chest X-ray images using convolutional neural networks and transfer learning," *Measurement*, vol. 165, pp. 1–10, Dec. 2020, doi: 10.1016/j.measurement.2020.108046.
- [14] K. Azmi, S. Defit, and Sumijan, "Implementasi convolutional neural network (CNN) untuk klasifikasi batik tanah liat Sumatera Barat," *J. Unitek*, vol. 16, no. 1, pp. 28–40, Jan.-Jun. 2023.
- [15] A. Batool and Y.-C. Byun, "A lightweight multi-path convolutional neural network architecture using optimal features selection for multiclass classification of brain tumor using magnetic resonance images," *Results Eng.*, vol. 25, pp. 1–15, Mar. 2025, doi: 10.1016/j.rineng.2025.104327.
- [16] S.S. Bhat, A. Ananth, and P.S. Venugopala, "Design and evolution of deep convolutional neural networks in image classification - A review," *Int. J. Integr. Eng.*, vol. 15, no. 1, pp. 213–225, May 2023, doi: 10.30880/ijie.2023.15.01.019.
- [17] I.B. Santoso and I.K.E. Purnama, "Epileptic EEG signal classification using convolutional neural networks based on optimum window length

- and FFT's length," in *ICCCM '20, Proc. 8th Int. Conf. Comput. Commun. Manag.*, 2020, pp. 87–91, doi: 10.1145/3411174.3411179.
- [18] M.A. Rahman, M.R. Islam, M.A.H. Rafath, and S. Mhejabin, "CNN based COVID-19 detection from image processing," *J. ICT Res. Appl.*, vol. 17, no. 1, pp. 99–113, Apr. 2023, doi: 10.5614/itbj.ict.res.appl.2023.17.1.7.
- [19] A. Ghosh *et al.*, "Fundamental concepts of convolutional neural network," in *Recent Trends and Advances in Artificial Intelligence and Internet of Things*, Cham, Switzerland: Springer, 2019, pp. 519–567.
- [20] S.B. Shrestha, L. Zhu, and P. Sun, "Spikemax: Spike-based loss methods for classification," 2020, *arXiv: 2205.09845*.
- [21] C.R. Rahman *et al.*, "Identification and recognition of rice diseases and pests using convolutional neural networks," *Biosyst. Eng.*, vol. 194, pp. 112–120, Jun. 2020, doi: 10.1016/j.biosystemseng.2020.03.020.
- [22] Wardianto, Farikhin, and D.M.K. Nugraheni, "Analisis sentimen berbasis aspek ulasan pelanggan restoran menggunakan LSTM dengan ADAM optimizer," *JOINTECS (J. Inf. Technol. Comput. Sci.)*, vol. 8, no. 2, pp. 67–76, May 2023, doi: 10.31328/jointecs.v8i2.4737.
- [23] A. Bhattacharjee, A.A. Popov, A. Sarshar, and A. Sandu, "Improving the adaptive moment estimation (ADAM) stochastic optimizer through an implicit-explicit (IMEX) time-stepping approach," 2024, *arXiv: 2403.13704*.
- [24] M.A. Islam *et al.*, "An automated convolutional neural network based approach for paddy leaf disease detection," *Int. J. Adv. Comput. Sci. Appl. (IJACSA)*, vol. 12, no. 1, pp. 280–288, Jan. 2021, doi: 10.14569/IJACSA.2021.0120134.
- [25] S. Ghosal and K. Sarkar, "Rice leaf diseases classification using CNN with transfer learning," in *IEEE Calcutta Conf. (CALCON)*, 2020, pp. 230–236, doi: 10.1109/CALCON49167.2020.9106423.
- [26] A.A.J.V. Priyanka and I.M.S. Kumara, "Classification of rice plant diseases using the convolutional neural network method," *Lontar Komput. J. Ilm. Teknol. Inf.*, vol. 12, no. 2, pp. 123–129, Aug. 2021, doi: 10.24843/lkjiti.2021.v12.i02.p06.
- [27] A.R. Muslikh, D.R.I.M. Setiadi, and A.A. Ojugo, "Rice disease recognition using transfer learning Xception convolutional neural network," *J. Tek. Inform. (Jutif)*, vol. 4, no. 6, pp. 1535–1540, Dec. 2023, doi: 10.52436/1.jutif.2023.4.6.1529.