**Indonesian Journal of Artificial Intelligence and Data Mining (IJAIDM)**
Vol. 8, No. 2, July 2025, pp. 380 – 386
p-ISSN: 2614-3372 | e-ISSN: 2614-6150


# Pest Detection on Green Mustard Plants Using Convolutional Neural Network Algorithm

**[1*]Nurhikma Arifin, [2]Siti Aulia Rachmini, [3]Juprianus Rusman**
[1,2]Informatics, Engineering Faculty, Universitas Sulawesi Barat, Indonesia
[3]Informatics, Universitas Kristen Indonesia Toraja, Indonesia
Email: [1]nurhikma_arifin@unsulbar.ac.id, [2]sitiaulia.rachmini@unsulbar.ac.id, [3]rusman.jr@ukitoraja.ac.id

**Article Info**
***Article history:***
Received Jun 23th, 2024
Revised May 05th, 2025
Accepted Jul 12th, 2025

***Keyword:***
Adam optimizer
Convolutional Neural Network
Detection
Green Mustard Plants
Pest

**ABSTRACT**
The productivity of mustard greens is vulnerable to pests and diseases that can threaten the yield and quality of the harvest. This study aims to detect pests on green mustard plants using the Convolutional Neural Network (CNN) method. The dataset used in this research consists of 450 images, with 225 images of pest-infested mustard greens and 225 images of healthy mustard greens. These 450 datasets are divided into 400 training data and 50 testing data. The testing was conducted fifteen times using CNN architectures with 2, 3 and 4 convolutional layers, having filter numbers of (64,32) (64, 32, 16) and (64, 32, 16, 8) respectively, and learning rates ranging from 0.1 to 0.00001 with the Adam optimizer. Based on the testing results of the learning rate and the number of layers, it was found that a learning rate of 0.001 provided the best performance with the highest accuracy and the lowest loss, especially in the model with 3 layers (64, 32, 16), which achieved an accuracy of 94% and a loss of 24.92%. A learning rate that is too high (0.1) or too low (0.00001) results in poor performance and instability, with low accuracy and high loss. Therefore, selecting the appropriate learning rate is crucial to achieving optimal results in model training.


***Corresponding Author:***
Nurhikma Arifin,
Informatics, Engineering Faculty,
Universitas Sulawesi Barat,
Jalan Prof. Dr. Baharuddin Lopa, S.H, Talumung, Kabupaten Majene, Sulawesi Barat 91412.
Email: nurhikma_arifin@unsulbar.ac.id
**DOI: http://dx.doi.org/10.24014/ijaidm.v8i2.30953**

## 1. INTRODUCTION

Green mustard plants have become one of the types of leafy vegetables that are easy to cultivate, offering ease in the growth and care processes. The reliability of mustard plants in adapting to various environmental conditions makes them a top choice for farmers to enhance agricultural sustainability. Besides the ease of cultivation, green mustard also has good prospects as a source of income for farmers and a contributor to community nutrition. However, the productivity potential of mustard can be hampered by pest and disease attacks, which threaten crop yields and plant quality [1].

Pest detection in green mustard plants is a critical aspect of agricultural sustainability and food security. Green mustard, as an essential component of food and nutrition, often becomes the target of pests such as aphids, caterpillars, and various plant diseases. These attacks can harm crop yields and plant quality, significantly negatively impacting food security and the local economy. Therefore, efficient solutions are needed to detect pests on plants to support sustainable agriculture towards Smart Agriculture.

Conventional approaches to pest identification on plants involve manual processes that are time-consuming and resource-intensive. With technological advancements, particularly in artificial intelligence, many methods have emerged as promising alternatives to support Smart Agriculture (CNN). One such method is the CNN. Research on pest detection in plants using Convolutional Neural Networks has been widely conducted due to its ability to recognize and classify images with high accuracy [2 - 4]. Sharma et al.

have conducted a study on automatic Pest Detection in Plants using Faster CNN with an average precision accuracy of 98% and 94% for 30% testing data[5]. Hadipour-Rokni et al. implemented a machine vision system and CNN with transfer learning technique to intelligently detect pests on citrus fruits. The results indicate that the VGG-16 and AlexNet models, utilizing the SGDm algorithm, attained the highest accuracies of 98.33% and 99.33% respectively [6]. Erin et al. conducted research on classifying green mustard plant diseases using the CNN method, achieving an accuracy of 99% [7]. She et al developed a method for automatically detecting and counting fruit fly pests in orchards by trap bottles via convolutional neural network with attention mechanism added. The results indicate that by incorporating semantic segmentation U-Net and YOLOv5 algorithm with attention mechanism, achieving an accuracy of 93.5% in counting pests on trap bottle surfaces and 94.3% in counting pests entering trap bottles, facilitating real-time pest monitoring in orchards[8]. Nanny et al Nanny et al. developed a superior ensemble of convolutional neural networks for insect pest image detection. An ensemble of CNNs trained and optimized using various Adam variants for automatic identification of invasive insects achieved the best performance on several datasets, including 95.52% on Deng, 74.11% on IP102, and 99.81% on Xie2, demonstrating significant potential in accelerating the process of recognition and eradication of pests [9]. Sakshi et al. conducted a study on mustard leaf disease detection using Sequential CNN, ResNet-50, VGG, and AlexNet. Their approach showed significant improvements over traditional methods, providing an efficient tool for disease detection in mustard crops and promoting sustainable agriculture for global food security and environmental sustainability[10]. Meanwhile, S. Rana et al. implemented an effective approach to detect pests in jute plants using a Lightweight CNN model, achieving an accuracy of 99.43%. Integration with Gradio enhanced the accessibility and usability of the model for stakeholders [11]. The study by N. U. Noor and F. Solaiman demonstrated that CNN, particularly with the ResNet-101 and DarkNet-19 architectures, is capable of classifying plant diseases with over 99% accuracy and has the potential to enhance real-time early detection in the agricultural sector [12]. S. M. Venkateswara and J. Padmanabhan proposed an automated deep learning-based approach in smart agriculture for pest identification and classification using the IP102 dataset, utilizing autoencoders to address data imbalance, RGB-based object detection and segmentation techniques, and CNN for classification, achieving an accuracy of 84.95% and demonstrating significant potential in enhancing the effectiveness of pest control in the agricultural sector [13] and many researchers still use the CNN method for detecting leaf diseases [14 – 19].

Based on a literature review, this study aims to apply the CNN method in detecting pests on green cabbage plants as a revolution in smart agriculture, with the goal of increasing accuracy and effectiveness of detection. Focusing on specific plant species, this study will test and select optimal parameters such as CNN model architecture, number of kernels or filters, and other parameters like learning rate, number of epochs, and batch size. The selection of these parameters will precisely influence the model's performance in extracting important features from images and enhancing pest detection capabilities on green cabbage plants. CNN is a type of deep learning architecture that has proven successful in pattern recognition on visual data, including image processing. In the context of pest detection, CNN can be trained to recognize visual patterns indicating the presence of pests on green mustard plants. Implementing the CNN method in pest detection on green mustard plants not only provides a practical solution but also can positively impact agricultural efficiency and crop yields. By leveraging artificial intelligence in image processing, the hope is to provide more effective support to farmers in overcoming pest attack challenges, increasing agricultural productivity, and ultimately contributing to global food security.

## 2. RESEARCH METHOD

This study utilizes the CNN method to detect pests on green mustard plants using the Python programming. The CNN method was chosen for its capability to recognize visual patterns indicating the presence of pests on green mustard plants. CNN can effectively learn feature representations from image data, thus enabling accurate pest recognition. The system stages in this study are shown in Figure 1.

### 2.1 Data Collection

The dataset utilized in this study was sourced from Kaggle. It consists of 450 images, with 225 images of pest-infested mustard greens and 225 images of healthy mustard greens. Of these 450 images, 400 are used for training data and 50 for testing data. An example of green mustard plants to be detected is shown in Figure 2.

### 2.2 Labelling Data

In the data labeling process, the dataset is divided into two components: training data and testing data. The training data is utilized to train the machine learning model and consists of 200 samples of pest-infested plants and 200 samples of healthy plants. Each image is labeled accordingly to indicate whether it represents a pest-infested or healthy green mustard plant. On the other hand, the testing data comprises 50 unlabeled samples. This process is conducted to evaluate the model's performance after being trained with the training data. This labeling is crucial for the model to learn and distinguish between the two categories during the training phase and to evaluate its performance during the testing phase.

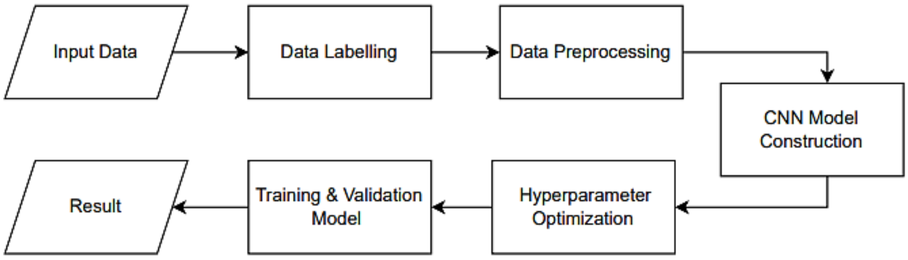


**Figure 1.** The system stages using the CNN method

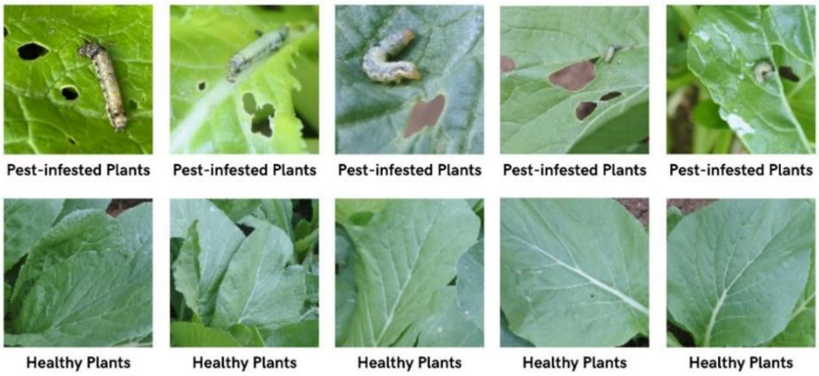


**Figure 2.** Example of healthy and pest-infested green mustard plant data.

### 2.3 Preprocessing Data

In the preprocessing stage, data rescaling is performed by changing the image size. The rescaling process in this study uses the function "rescale=1./255," which transforms the pixel value range of the images to the range of 0-1 with a size of 128 x 128. Rescaling is used to reduce the influence of varying pixel values on the model's results and to enhance the model's ability to detect patterns.

### 2.4 Construction of CNN Model Architecture

In this stage, the architecture of the CNN model is constructed. This process involves determining the number of convolutional layers, the number of filters, the kernel size, the activation function, and the pooling layers, as well as setting up the connection layers and the output layer. The model architecture is designed to extract important features from the input images and learn patterns present in the training data. The CNN architecture in this study consists of three convolutional layers followed by max pooling to extract features from the input images using a 3x3 kernel and ReLU (Rectified Linear Unit) activation. This architecture is then followed by a dropout layer to reduce overfitting, and finally, a fully connected layer for classification purposes using the sigmoid activation function. The specific configuration of these layers is outlined in Table 1.

Based on Table 1, the CNN architecture used in this study consists of several layers specifically arranged to achieve the best performance in detecting pests on green mustard plants. Initially, various experiments were conducted with different configurations of convolutional and pooling layers. For instance, the first convolutional layer produces an output of 126x126x64 using 64 filters, followed by a max-pooling layer that reduces the output size to 63x63x64. Subsequent experiments involved adding convolutional layers and adjusting the number of filters. In the second convolutional layer, 32 filters were used to produce an output of 61x61x32, followed by pooling that reduces the size to 30x30x32. The third convolutional layer used 16 filters to produce an output of 28x28x16, and after pooling, the output size became 14x14x16.

**Table 1.** The specific configuration of these layers

| Layer Type | Output Shape |
|---|---|
| Conv2d_1 | 126, 126, 64 |

| Layer Type | Output Shape |
|---|---|
| Max_pooling2d_1 | 63, 63, 64 |
| conv2d_2 | 61, 61, 32 |
| Max_pooling2d_2 | 30, 30, 32 |
| conv2d_3 | 28, 28, 16 |
| Max_pooling2d_3 | 14, 14, 16 |
| Dropout | 14, 14, 16 |
| flatten | 3136 |
| Dense | 2 |

To mitigate overfitting, a dropout layer was added after the final convolutional layer, setting 50% of neurons randomly to zero during training. The output from this layer was then flattened into a 1D vector with a length of 3136, which was subsequently passed to a fully connected layer with 2 neurons. Throughout the experiments, the number of training epochs was assumed to be 10 to ensure adequate training. Based on the experimental results, this architecture was chosen because it provided the best accuracy with an efficient number of parameters. The process of selecting this architecture involved evaluating various combinations of filters, kernel sizes, and pooling layers, as well as using dropout techniques to reduce overfitting. The final architecture is designed to maximize pest detection performance on green mustard plants while maintaining model efficiency.

### 2.5 Hyperparameters Optimization

The training process involves using several hyperparameters. Hyperparameters are TensorFlow parameters used to regulate the behavior of the model and optimize its performance, such as epoch, batch size, learning rate, and optimization [12]. The hyperparameters used in this study are shown in the following Table 2.

**Table 2.** Values of Hyperparameters

| Parameter | Value/Type |
|---|---|
| Epoch | 100 |
| Learning Rate | 0.1-0.00001 |
| Batch sizes | 64 |
| Optimizer | Adam |
| Callback Function | Validation Loss |
| Loss Function | Categorical Cross-Entropy |

The hyperparameter testing in Table 2 covers several crucial aspects of the model training process. The number of epochs used is 100, allowing the model to thoroughly learn patterns from the data by passing through the entire training set 100 times. The learning rate is tested in a range from 0.1 to 0.00001 to find the most optimal value for controlling weight changes in the model during each training iteration. The batch size is set at 64, meaning the training data will be divided into small batches of 64 samples each, helping to reduce memory usage and speed up training. The optimizer used is Adam, which combines the advantages of RMSProp and Momentum algorithms to accelerate convergence and enhance model performance. The callback function employed is validation loss, allowing the tracking of model performance on validation data during training and enabling actions like early stopping if validation loss does not improve. This hyperparameter testing aims to find the best combination that provides optimal performance, ensuring the model is trained effectively and efficiently.

## 3. RESULTS AND ANALYSIS

In detecting pests on green mustard plants using CNN method, 15 trials were conducted to evaluate both the number of layers and learning rate. The tested layer configurations included (64, 32), (64, 32, 16), and (64, 32, 16, 8). This was done to determine the most optimal layer configuration for achieving high accuracy in pest detection. Additionally, learning rates were tested within the range of 0.1 to 0.00001 for each layer configuration. The goal was to identify the most effective learning rate values that control weight adjustments during training, ensuring fast convergence and optimal accuracy. This testing process is crucial for ensuring the CNN model efficiently detects pests on green mustard plants with high precision. The results of the experiments are shown in the following Table 3 and 4.

**Table 3.** Accuracy Values in Various Layer Configurations and Learning Rate Tests

| Layer | Learning Rate | | | | |
|---|---|---|---|---|---|
| | 0.1 | 0.01 | 0.001 | 0.0001 | 0.00001 |
| 64, 32 | 50% | 50 % | 88% | 86% | 58% |
| 64, 32, 16 | 50% | 50 % | 94% | 88% | 56% |
| 64, 32, 16, 8 | 50 % | 56 % | 92% | 56% | 54% |

**Table 4.** Loss Values in Various Layer Configurations and Learning Rate Tests

| Layer | Learning Rate | | | | |
|---|---|---|---|---|---|
| | 0.1 | 0.01 | 0.001 | 0.0001 | 0.00001 |
| 64, 32 | 69.32% | 69.32% | 34.07% | 41.65% | 68.44% |
| 64, 32, 16 | 69.32% | 69.32% | 24.92% | 48.74% | 69.21% |
| 64, 32, 16, 8 | 69.32% | 68.46% | 31.55% | 66.41% | 69.01% |

The results of the tests in Tables 3 and 4 show that adjustments in the learning rate and the number of layers have a significant impact on the model's performance. Testing with a learning rate of 0.1 resulted in low accuracy (50%) and high loss (69%) across all layer configurations tested. This indicates that the learning rate was too high, preventing the model from learning effectively. In the tests with a learning rate of 0.01, accuracy slightly improved for the 4-layer configuration (56%) but remained low for the 2 and 3-layer configurations (50%). This learning rate was still too high for most scenarios, indicating overfitting or underfitting. The tests with a learning rate of 0.001 yielded the best results, with high accuracy (88%, 94%, and 92%) and low loss (34.07%, 24.92%, and 31.55%). This learning rate was optimal for effective model learning, showing a good balance between convergence and stability. Testing with a learning rate of 0.0001 showed a decrease in accuracy (86%, 88%, and 56%) and an increase in loss (41.65%, 48.74%, and 66.41%) compared to the $10^{-3}$ learning rate. The final tests with a learning rate of 0.00001 resulted in very low accuracy (58%, 56%, and 54%) and high loss (68.44%, 69.21%, and 69.01%). This learning rate was too low, causing the model to struggle with convergence and effective learning.

The impact of the number of layers on model performance is evident in the results. For the configuration with 2 layers (filters: 64, 32), the optimal learning rate of 0.001 resulted in an accuracy of 88% and a loss of 34.07%. For the configuration with 3 layers (filters: 64, 32, 16), the same learning rate yielded the highest accuracy of 94% and the lowest loss of 24.92%, indicating that this configuration is the most optimal. For the configuration with 4 layers (filters: 64, 32, 16, 8), the learning rate of 0.001 resulted in an accuracy of 92% and a loss of 31.55%, which is slightly lower than the 3-layer configuration.

The results of testing various numbers of layers and learning rates in this study show that a learning rate of 0.001 with the layer configuration (64, 32, 16) is the most optimal for this model, achieving the best accuracy of 94% and the lowest loss of 24.92%. Adding more layers can provide a slight increase in accuracy up to a certain point, but the most significant impact comes from adjusting the learning rate. Therefore, determining the correct learning rate is more critical than increasing the number of layers in the model configuration to achieve optimal performance.

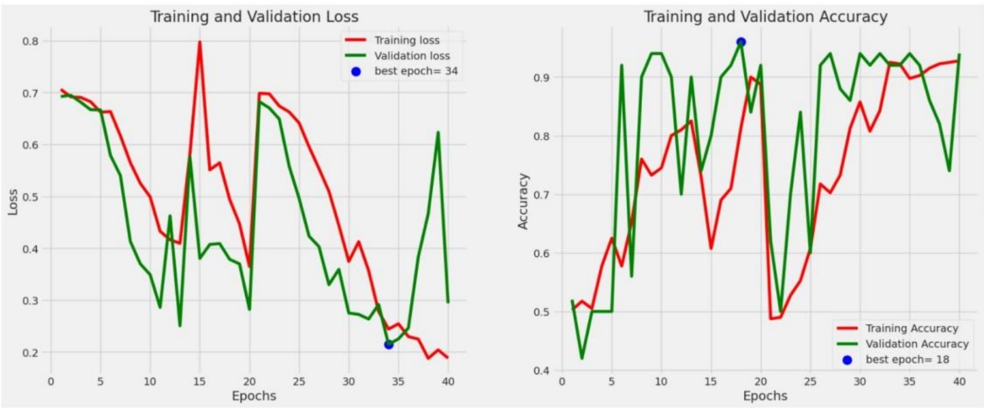


**Figure 3.** Visualization of training and validation for loss and accuracy

Based on the graph in Figure 3, the best results for training and validation loss were achieved at epoch 30, with a value of 0.2941, while the optimal training and validation accuracy was reached at epoch 18 with a value of 0.9400. This indicates that although the loss continues to decrease until epoch 30, the optimal accuracy is achieved earlier at epoch 18. Therefore, it is recommended to stop training the model at epoch 18 to prevent overfitting and ensure good and consistent model performance.

The following table 5 shows the results of the model testing using a confusion matrix to evaluate its performance in detecting pest-infested and healthy images. Out of 25 pest-infested images, the model correctly detected 24 images, indicating an accuracy of 96% for this class. However, 1 image was incorrectly detected as healthy. For healthy images, out of a total of 25 images, the model correctly detected 23 images, resulting in an accuracy of 92% for the healthy class, although 2 images were incorrectly detected as pest-infested. The overall accuracy of the model, calculated by summing the number of correct detections (True

Positives and True Negatives) divided by the total number of data points (50 images), is 94%. This indicates that the model performs well in detecting both types of images, although there are still some detection errors.

**Table 3.** Testing results using the confusion matrix

| Actual | Predicted | | Accuracy |
|---|---|---|---|
| | Pest Infested | Healthy | |
| Pest Infested | 24 | 1 | 96 % |
| Healthy | 2 | 23 | 92% |
| Total Accuracy | | | 94% |

Detection errors in the model can be caused by various factors, such as low data quality, high variability in images, overfitting or underfitting, and biases in the training dataset. Blurry images or poor lighting conditions, variations in angles and backgrounds, as well as the diversity in pest forms, can challenge the model in accurately recognizing patterns.

## 4. CONCLUSION

This study demonstrates the development and evaluation of a CNN model for pest detection on green mustard plants. The dataset was divided into training and testing sets, each consisting of 200 samples of pest-infested and healthy plants for training, and 50 unlabeled test samples. The use of a dropout layer in the CNN architecture helped mitigate overfitting risks, while hyperparameter optimization such as epoch and learning rate significantly contributed to the model's accuracy. Test results showed that the configuration with three convolutional layers (64, 32, 16) and a learning rate of 0.001 achieved the highest accuracy of 94%, despite challenges in detection due to variations in image quality and dataset biases. Overall, this model shows promise for pest detection applications in agriculture with good effectiveness, though further refinement is needed to improve generalization and resilience to diverse image conditions.

For future research, it is recommended to broaden the diversity of the dataset by including various images of pest-infested and healthy plants under different environmental conditions and stages of infestation. Researchers could also explore advanced image preprocessing techniques to address challenges such as varying lighting conditions, image blurriness, and background noise. Investigating the integration of data from multi-sensor sources like infrared or hyperspectral imaging alongside visual images could provide additional insights and enhance pest detection accuracy. Developing real-time monitoring systems using trained CNN models for pest detection could be a next step, incorporating IoT devices and automated alert systems to support timely and informed agricultural decision-making.

## REFERENCES


[1] Puspaningrum AS, Susanto ER, Sucipto A. The Implementation of Forward Chaining Method for Diagnosing Diseases in Mustard Plants. *Inform J.* 2020 Dec 29;5(3):113.

[2] D L Shanthi, K Vinutha, N Ashwini, Vashistha S. *Tomato Leaf Disease Detection Using CNN*. Procedia Comput Sci. 2024;235:2975–84.

[3] Falaschetti L, Manoni L, Di Leo D, Pau D, Tomaselli V, Turchetti C. A CNN-based image detector for plant leaf diseases classification. HardwareX. 2022 Oct;12:e00363.

[4] Fu Y, Guo L, Huang F. A lightweight CNN model for pepper leaf disease recognition in a human palm background. Heliyon. 2024 Jun;10(12):e33447.

[5] Sharma S. Pest Detection in Plants Using Convolutional Neural Network. *Int J Res Appl Sci Eng* Technol. 2021 Nov 30;9(11):1583–92.

[6] Hadipour-Rokni R, Askari Asli-Ardeh E, Jahanbakhshi A, Esmaili paeen-Afrakoti I, Sabzi S. Intelligent detection of citrus fruit pests using machine vision system and convolutional neural network through transfer learning technique. Comput Biol Med. 2023 Mar 1;155:106611.

[7] Ratnasari ED, Rudira DA, Buana AS. *Klasifikasi Penyakit Daun Sawi Hijau dengan Metode CNN*. Pros Semin Nas Teknol dan Sains. Kediri. 2024;3:388–393.

[8] She J, Zhan W, Hong S, Min C, Dong T, Huang H, et al. A method for automatic real-time detection and counting of fruit fly pests in orchards by trap bottles via convolutional neural network with attention mechanism added. Ecol Inform. 2022 Sep 1;70:101690.

[9] Nanni L, Manfè A, Maguolo G, Lumini A, Brahnam S. High performing ensemble of convolutional neural networks for insect pest image detection. Ecol Inform. 2022 Mar 1;67:101515.

[10] Sakshi, C. Sharma, S. Sharma, T. Sharma, S. Gochhait. Green Intelligence: A Sequential CNN Odyssey in Mustard Leaf Disease Detection. In: 2024 ASU International Conference in Emerging Technologies for Sustainability and Intelligent Systems (ICETSIS). 2024. p. 877–82.

[11] S. Rana, M. F. Samad. End-to-End Jute-Pest Detection By Explainable Lightweight CNN. In: 2024 6th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT). 2024. p. 230–5.

[12] N. U. Noor dan F. Solaiman. CNN-Based crop disease and pest detection systems: Enhancing accuracy and efficiency in agricultural sustainability. J. Bangladesh Acad. Sci., vol. 49, no. 1, hlm. 73–85, Jun 2025, doi: 10.3329/jbas.v49i1.78209.
[13] S. M. Venkateswara dan J. Padmanabhan. Deep learning based agricultural pest monitoring and classification. Sci. Rep., vol. 15, no. 1, hlm. 8684, Mar 2025, doi: 10.1038/s41598-025-92659-5.
[14] X. Yue, K. Qi, X. Na, Y. Liu, F. Yang, dan W. Wang. Deep learning for recognition and detection of plant diseases and pests. Neural Comput. Appl., vol. 37, no. 17, hlm. 11265–11310, Jun 2025, doi: 10.1007/s00521-025-11125-5.
[15] S. A. Kumar, F. Shaik, B. Yashwitha, P. Vidyavathi, A. U. Maheswari and P. Spurthi. Intelligence Pest Detection and Control in Agriculture using Computer Vision and Deep Learning. 2025 International Conference on Intelligent Computing and Control Systems (ICICCS), Erode, India, 2025, pp. 1147-1150, doi: 10.1109/ICICCS65191.2025.10985611.
[16] J. Xie dkk., “Intelligent Detection and Control of Crop Pests and Diseases: Current Status and Future Prospects,” Agronomy, vol. 15, no. 6, hlm. 1416, Jun 2025, doi: 10.3390/agronomy15061416.
[17] P. Parameswari, M. Manikantan and P. A. Surya. Enhanced CNN Model Integrated with Generative AI for Real-Time Pest and Disease Detection in Rice Farming. 2025 3rd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India, 2025, pp. 1-4, doi: 10.1109/ICAECA63854.2025.11012178.
[18] L. Zhang, S. Sun, H. Zhao, Z. Li, dan D. Li, “Multi-species insect recognition method based on computer visions: Sustainable agricultural development,” Ecol. Inform., vol. 88, hlm. 103125, Sep 2025, doi: 10.1016/j.ecoinf.2025.103125.
[19] R. Sankar, M. Anubala, M. Hemalatha, U. Lavanya and S. Narmadha, "Twin-Layer Deep Learning Model with IoT Integration for Intelligent Pest Detection in Smart Farming," 2025 Fourth International Conference on Smart Technologies, Communication and Robotics (STCR), Sathyamangalam, India, 2025, pp. 1-6, doi: 10.1109/STCR62650.2025.11020488.
[20] Gandadipoera FHM, Andrian R, Safe’i R. Tree Damage Type Classification Based on Forest Health Monitoring Using Mobile-Based Convolutional Neural Network. *Indonesia J Artif Intell Data Min IJAIDM*. 2024 Sep;7(2):299–307.

**BIBLIOGRAPHY OF AUTHORS**

Nurhikma Arifin, born in Majene in 1993, is a lecturer in the Informatics Study Program, Faculty of Engineering, Universitas Sulawesi Barat. She earned her Bachelor’s degree in Informatics Engineering from Universitas Islam Negeri Alauddin Makassar and her Master’s from Universitas Hasanuddin. Her research interests include artificial intelligence, machine learning, deep learning, data mining, and image processing. Committed to education and research, she actively contributes to advancing knowledge in Informatics and supporting the development of scientific innovation in Indonesia.

Siti Aulia Rachmini, born in Jakarta in 1982, holds a Bachelor’s degree in Informatics Engineering (Intelligent Systems) from Trisakti University (2005) and a Master’s in Electrical Engineering and Informatics from Hasanuddin University (2016). Her research interests include Artificial Intelligence, software engineering, and multimedia processing. She is actively engaged in teaching, applied research, and institutional development, with a focus on integrating AI technologies into real-world solutions and preparing students to address the challenges of the digital era.

Juprianus Rusman, born in Rantepao in 1990, earned his Bachelor’s degree in Informatics Engineering (Intelligent Systems) from STMIK Dipanegara Makassar (2012) and a Master’s in Electrical Engineering and Informatics from Hasanuddin University (2019). His expertise includes microcontroller systems, embedded systems, IoT, and robotics. As an academic, he focuses on developing and applying intelligent systems and automation technologies, while fostering innovation and advancing computer science through education, research, and community contributions in Indonesia.