Journal of Applied Informatics and Computing (JAIC)
Vol.10, No.2, April 2026, pp. 1952~1961
e-ISSN: 2548-6861

Application of the Yolov8 Algorithm for Detecting Rice Plant Diseases with Web-Based Digital Images

Rakhmat Hidayat [1]*, Noora Qotrun Nada [2]*, Agung Handayanto [3]*
*Informatics Departement, Universitas Persatuan Guru Republik Indonesia Semarang
hidayatrakhamat38@gmail.com [1], noora@upgris.ac.id [2], agunghan@upgris.ac.id [3]

Article Info

Article history:

Received 2026-02-26
Revised 2026-03-16
Accepted 2026-04-10

Keywords:

Rice Leaf Disease,
YOLOv8,
Deep Learning,
Image Classification,
Smart Agriculture.

ABSTRACT

The decline in environmental quality caused by industrial pollution and climate change has weakened the natural resistance of rice plants (Oryza sativa), increasing their susceptibility to various diseases. Conventional disease identification methods that rely on manual observation are often limited by subjectivity and human visual constraints. This study proposes a deep learning–based system for automatic rice leaf disease classification using the You Only Look Once version 8 (YOLOv8) architecture. The model was trained using a publicly available rice leaf image dataset consisting of 6,889 images categorized into eight classes: Bacterial Leaf Blight, Brown Spot, Leaf Blast, Leaf Scald, Sheath Blight, Narrow Brown Leaf Spot, Rice Hispa, and Healthy Rice Leaf. The research methodology includes image pre-processing, data augmentation, dataset splitting, and training using the YOLOv8n-cls model for 50 epochs. Experimental results demonstrate high classification performance with an accuracy of 99.5%, precision of 99%, recall of 98%, and an F1-score of 0.99. The trained model was then deployed into a web-based application that allows users to upload rice leaf images and obtain real-time disease classification results. The proposed system provides a practical tool to support early detection of rice plant diseases and assist farmers in improving crop management in modern agriculture.

I. INTRODUCTION

Rice is a staple food for Indonesians and plays a significant role in ensuring national food security[1]. The success of rice cultivation is greatly influenced by the quality of the growing environment, especially the availability of clean and decent irrigation water[2]. Irrigation water functions as a medium for distributing nutrients and maintaining the physiological balance of rice plants so that they can grow and develop optimally[3]. However, the degradation of environmental quality is currently a serious threat to the productivity of rice fields in various regions of Indonesia[4]. Water pollution in Indonesia is often caused by low public awareness in managing domestic and industrial waste, which has a systemic impact on the availability of clean water for the agricultural sector[5]. As a case study, in Sojomerto Village, splitting stone mining activities without adequate supervision has caused the river water to become cloudy due to pollution of mining materials in the form of mud, sediment, and oil residues[6]. This condition causes irrigation water to no longer meet physical and chemical quality standards, which ultimately makes it difficult for people to utilize water resources for agricultural production needs. The decline in the quality of irrigation water has a direct impact on the physiological health of rice plants[7]. Water contaminated with mining residues can damage soil structure and cause plants to experience prolonged environmental stress[8]. Rice plants that grow under stress due to environmental pollution tend to have weakened immune systems, making them much more susceptible to attacks of harmful pathogens[9]. Plant infections such as bacterial leaf blight, brown spot, and leaf blast show higher intensity on contaminated land, which ultimately causes a significant decline in crop productivity[10]. Therefore, the need for an early and accurate mechanism for detecting rice diseases is very urgent to mitigate farmers' economic losses[11]. So far, the identification of rice diseases in

Sojomerto Village is still carried out conventionally through manual visual observation. This method has a fundamental drawback because it relies heavily on the subjectivity and individual experiences of farmers. The limitations of human vision in recognizing the early symptoms of the disease often lead to misidentification or delay in diagnosis in large areas of rice fields[12]. This delay results in non-optimal medical treatment of plants, so the spread of the disease becomes uncontrollable. In addition, manual monitoring requires a significant amount of time and labor, especially when rice fields cover large agricultural areas. Therefore, a more efficient and automated approach is required to assist farmers in identifying plant diseases more quickly and accurately. The development of computer vision and deep learning technology offers automation solutions in detecting plant diseases based on digital imagery with high precision [13]. Deep learning methods are capable of extracting complex visual patterns from plant leaves and learning discriminative features that distinguish between healthy and diseased plants. Recent studies have shown that convolutional neural networks and modern object detection architectures can significantly improve the performance of plant disease identification systems. Gupta et al. [13] demonstrated that YOLO-based deep vision models provide reliable performance in classifying plant leaf diseases by efficiently learning visual representations from large-scale datasets. Similarly, Krisdianto et al. [14] implemented the YOLO algorithm to detect rice plant diseases and reported promising results in improving the accuracy of disease detection compared to traditional image processing techniques. These findings indicate that deep learning-based approaches have strong potential to support precision agriculture and smart farming systems. The YOLO (You Only Look Once) algorithm is one of the most advanced architectures capable of classifying and detecting objects in real time [14]. Unlike traditional detection methods that perform multi-stage processing, YOLO performs object localization and classification simultaneously within a single neural network architecture, resulting in faster inference speed and improved computational efficiency. YOLOv8 has the advantage of an improved backbone and neck architecture that results in higher detection accuracy and better computing efficiency. This capability allows YOLOv8 to be deployed on devices with limited specifications, making it a highly suitable solution for real-time plant disease detection systems [15]. These advantages make YOLOv8 ideal for integration into plant disease early warning systems and digital agriculture platforms. Based on these problems, this study aims to develop a digital image-based rice crop disease detection system using the web-based integrated YOLOv8 algorithm. The benefit of this research is to provide a quick and accurate diagnostic tool for farmers in Sojomerto Village in facing the threat of plant diseases due to the impact of environmental quality degradation. Through this system, it is hoped that the effectiveness of disease control can be increased to maintain crop stability amid the challenges of irrigation water pollution.

II. METHOD

A. Flow of research methods

This research methodology begins with identifying the problem, followed by collecting a collection of digital image data. Next, a pre-processing stage is carried out to standardize the input data used by the model. The model is then trained using the YOLOv8 architecture. An evaluation process is carried out to ensure that the model achieves high accuracy. Once the evaluation results are considered satisfactory, the model is implemented into a web-based system. The overall research workflow is illustrated in Figure 1: Research Methodology Flow.

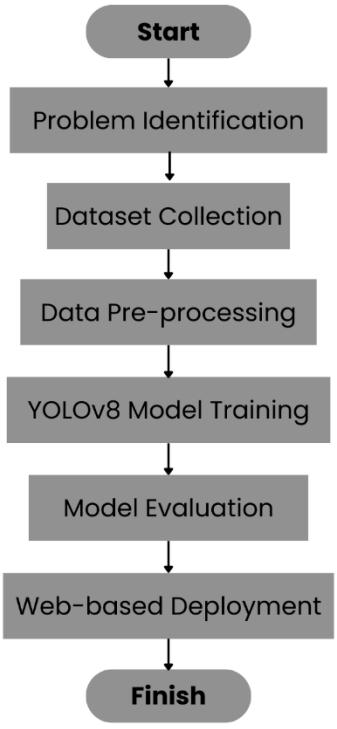

Figure 1 Research Method Flow

Problem identification is conducted through direct field surveys in rice fields, aiming to observe the actual condition of rice plants affected by diseases. This stage involves documenting visible disease symptoms as well as identifying factors that contribute to the occurrence of rice plant diseases.

B. Dataset Collection

The dataset used in this study consists of digital images of rice leaves representing eight different health conditions. The classes include Bacterial Leaf Blight, Brown Spot, Leaf Blast, Leaf Scald, Narrow Brown Leaf Spot, Rice Hispa, Sheath Blight, and Healthy Rice Leaf. In total, the dataset contains 6,889 images, which represent various visual symptoms of

rice leaf diseases. The primary dataset was obtained from a publicly available dataset on Kaggle [16], which provides a comprehensive collection of rice leaf disease images for machine learning research. This dataset has been widely used in previous studies related to plant disease detection and provides labeled images for several rice disease categories. In addition, this study refers to the dataset structure and disease categorization used in previous research by Jatmika and Saputra [17], who applied deep learning methods for identifying rice plant diseases using convolutional neural networks. The dataset categories used in this study follow a similar classification scheme to ensure consistency with previous research in the field of plant disease detection. The images in the dataset were captured under different environmental conditions, including variations in lighting intensity, background complexity, leaf orientation, and image quality. Such variations are important to ensure that the trained model can generalize well when applied to real-world agricultural environments where image conditions are often uncontrolled. The distribution of images for each class used in this study is presented in Table 1.

TABLE 1.
DISTRIBUTION OF RICE LEAF DISEASE DATASET

No	Types of Rice Diseases	Number of Images
1	Leaf Blast	1.234
2	Sheath Blight	1.096
3	Brown Spot	1.077
4	Rice Hispa	879
5	Leaf Scald	749
6	Bacterial Leaf Blight	716
7	Healthy Rice Leaves	668
8	Narrow Brown Leaf Spot	470
Total		**6.889**

C. Data Pre-Processing

Pre-processing of the dataset was carried out to ensure that the rice leaf images were ready for training with the YOLOv8 model. The pre-processing stages include:

1) Image Resizing: The entire image of the rice leaf is resized to 224 × 224 pixels. This resizing process aims to standardize the image dimensions so that they matches the needs of the model input and can speed up the computational process without losing important visual information in the image.
2) Data Labeling: Each image was annotated according to the type of disease present on the leaf. This labeling process assigns the images to their corresponding disease classes, which is essential for supervised learning in the classification task.
3) Data Quality Check: Corrupted or low-quality images were removed to maintain the integrity of the dataset and ensure reliable model training.

Although the dataset obtained from Kaggle [16] was already divided into training, validation, and testing subsets, these pre-processing steps ensure that all images are consistent in size, quality, and labeling, which is crucial for achieving optimal performance in YOLOv8 model training.

D. Data Augmentation

To improve the robustness of the YOLOv8 model and reduce the risk of overfitting, several data augmentation techniques were applied to the training dataset. Data augmentation artificially increases the diversity of the dataset by generating modified versions of existing images, simulating variations commonly encountered in real agricultural environments.

1) Horizontal Flipping: Images were flipped horizontally to simulate different leaf orientations during image capture.
2) Random Rotation: Images were randomly rotated within a certain degree range to account for various angles at which leaves may be photographed in the field.
3) Scaling and Zooming: Random scaling and zooming were applied to simulate variations in distance between the camera and the leaf surface.
4) Brightness and Contrast Adjustment: Variations in lighting conditions were simulated by adjusting the brightness and contrast of the images.
5) Random Cropping: Portions of the leaf images were randomly cropped to help the model recognize disease patterns even when only part of the leaf is visible.

These augmentation steps ensure that the YOLOv8 model learns generalized features and can accurately classify rice leaf diseases under different environmental conditions, such as varying lighting, leaf orientation, background complexity, and image quality.

E. Dataset Splitting

The dataset used in this study was already divided into three subsets provided by the Kaggle source [16]: training, validation, and testing. The training dataset is used to train the YOLOv8 model, the validation dataset is used to monitor the model's performance during training and prevent overfitting, and the testing dataset is used to evaluate the final performance of the trained model. Although the dataset was pre-split, the distribution ensures a sufficient number of images in each subset to enable effective training and evaluation. This structure helps maintain objectivity when assessing the model's performance and ensures that the trained model generalizes well to unseen data.

TABLE 2.
DATASET SPLITTING OVERVIEW

Subset	Percentage	Number of Images
Training	80%	5,511
Validation	10%	689
Testing	10%	689

F. YOLOv8 Model Architecture

YOLOv8 is a deep learning algorithm built on the Convolutional Neural Network (CNN) architecture, specifically designed to perform object detection and classification tasks efficiently within an integrated processing pipeline. In this study, the YOLOv8 architecture accepts rice leaf images of 224 × 224 pixels as input. The processing steps in the architecture include:

1) Backbone Network: Consists of convolutional layers, batch normalization, SiLU activation functions, and C2f blocks, which together extract important visual features from the input images.
2) Feature Enhancement: The extracted features are further processed using the Spatial Pyramid Pooling Fast (SPPF) module, which captures multi-scale contextual information from the images.
3) Classification Head: The enhanced features are passed to the classification head, which includes Global Average Pooling, a Fully Connected layer, and a Softmax activation function to produce the final class prediction.

The overall architecture configuration used in this study is illustrated in Figure 2: YOLOv8 Architecture.

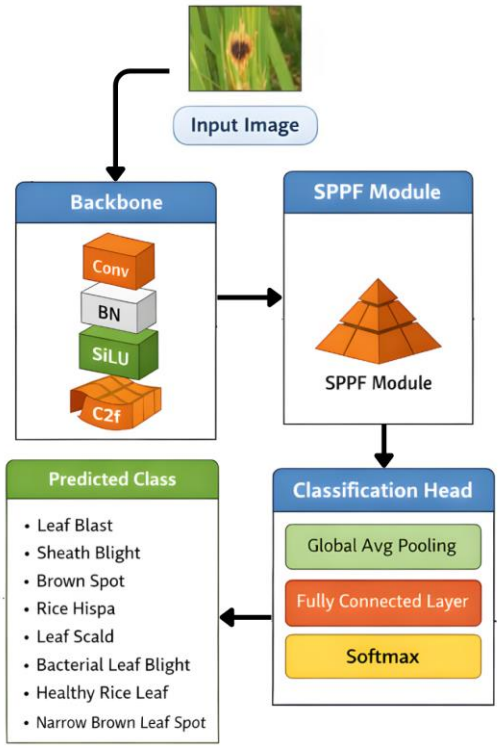

Figure 1 YOLOv8 Architecture

YOLOv8 was selected for this study because it offers a lighter, more stable, and more efficient architecture compared to previous YOLO versions and conventional classification algorithms. This model achieves an optimal balance between superior accuracy and fast inference speed. In addition, the optimized network architecture facilitates better adaptation to variations in data, making it particularly suitable for detecting rice plant diseases based on leaf image analysis, especially when dealing with limited datasets[18].

G. Model Training

The training process of the YOLOv8n-cls model was conducted using Google Colaboratory as a cloud-based computing environment that provides GPU acceleration to speed up the deep learning model training process[19]. The training procedure was implemented using the Ultralytics YOLOv8 framework, which is developed using the Python programming language and the PyTorch deep learning library that supports dynamic computational graphs and efficient GPU processing[20]. The dataset used in this study consisted of 5,171 training images, 738 validation images, and 1,478 testing images, which were categorized into eight classes of rice leaf health conditions, including Bacterial Leaf Blight, Brown Spot, Leaf Blast, Leaf Scald, Narrow Brown Leaf Spot, Rice Hispa, Sheath Blight, and Healthy Rice Leaf. During the training stage, the input images were resized to 224 × 224 pixels to match the input requirements of the YOLOv8 classification model. The training process was performed for 50 epochs with a batch size of 32, allowing the model to iteratively learn representative features from the rice leaf images. To improve training efficiency and model convergence, pretrained weights were utilized as the initial parameters of the network. In addition, the Automatic Mixed Precision (AMP) technique was applied to accelerate the computation process while maintaining numerical stability during model training[21]. The detailed configuration of the training parameters used in this study is presented in Table 3.

TABLE 3.
YOLOV8 TRAINING CONFIGURATION

Parameter	Value
Model	YOLOv8n-cls
Image Size	224 × 224
Epoch	50
Batch Size	32
Platform	Google Colab
Framework	*Ultralytics YOLOv8*
Hardware	GPU T4
Optimization	Automatic Mixed Precision

H. Model evaluation

The performance of the trained model was evaluated using several commonly used classification metrics, namely Accuracy, Precision, Recall, and F1-score, to comprehensively assess the effectiveness of the rice leaf disease classification system. These evaluation metrics are

calculated based on the confusion matrix, which consists of four main components: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Accuracy is used to measure the overall correctness of the model predictions by comparing the number of correct predictions to the total number of predictions. Precision evaluates the model's ability to correctly identify specific disease classes among all predicted instances. Recall measures the model's capability to detect all actual disease instances within a particular class. Meanwhile, the F1-score represents the harmonic mean of Precision and Recall, providing a balanced evaluation when considering both false positives and false negatives[22].

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (3)$$

$$\text{F1-score} = \frac{2TP}{2TP + FP + FN} \quad (4)$$

I. Web-Based System Deployment

Upon completion of the training and evaluation processes, the trained YOLOv8 model is saved in .pt format, which contains the learned parameters of the deep learning model. This trained model is then utilized in the deployment stage to enable automatic classification of rice leaf diseases within a web-based system. The development of the web application begins with the design of the user interface (UI) using Figma, which allows the system interface to be designed visually before implementation. The interface design aims to create a simple, responsive, and user-friendly layout, enabling users to easily interact with the system and upload rice leaf images for disease detection[23]. After the interface design stage, the web interface is implemented using HTML as the primary structure of the web page[24]. The system uses a native web framework approach, which keeps the application lightweight and efficient while reducing system complexity. This approach allows the website to load faster and ensures a responsive user interface that can adapt to different screen sizes. The backend system is developed using the Flask framework, which acts as the server responsible for processing user requests and running the trained deep learning model. Flask was chosen because it is lightweight, flexible, and easily integrated with Python-based deep learning models. In this architecture, when a user uploads an image of a rice leaf through the website interface, the image is sent to the Flask server, where the stored YOLOv8 model performs inference. The model then analyzes the uploaded image and produces a classification result, which is returned to the web interface and displayed to the user in real time[25], [26]. Furthermore, the web-based implementation allows the system to be accessed through both desktop and mobile devices without requiring additional application installation. Since the system is implemented as a web application, farmers can simply access the website through a browser on their smartphones, tablets, or personal computers, making the disease detection system more accessible and practical for real agricultural use.

III. Results and Discussion

A. Model Training Performance

The training process of the YOLOv8n-cls model was conducted using a rice leaf disease dataset consisting of eight disease classes. The model was trained for 50 epochs using a Tesla T4 GPU through the Google Colaboratory environment. The YOLOv8n-cls architecture used in this study contains 30 layers with approximately 1.45 million parameters, making it lightweight and computationally efficient for image classification tasks. The dataset used in the training process consisted of 5,171 training images, 738 validation images, and 1,478 testing images representing eight types of rice plant diseases. During the training process, the model gradually learned the discriminative visual features of rice leaf diseases, which can be observed through the decreasing loss values and increasing validation accuracy. At the early stage of training (Epoch 1), the model produced a relatively high loss value of 1.635 with a validation accuracy of 71.5%, indicating that the model had not yet fully learned the visual patterns of the dataset. As training progressed, the model performance improved significantly. By Epoch 10, the loss value decreased to 0.1917, while the validation accuracy increased to 97.2%. The model continued to converge, reaching a loss value of 0.1011 and validation accuracy of 99.2% at Epoch 20. At the final stage (Epoch 50), the loss further decreased to 0.02436, while the validation accuracy stabilized at approximately 99.2%, indicating stable learning without significant overfitting. Overall, the training results demonstrated that the model achieved Top-1 accuracy of 99.46% and Top-5 accuracy of 100%, with a fast inference time of approximately 0.20 ms per image, making the model suitable for real-time web-based rice disease detection systems[27]. The training performance curves for loss and validation accuracy across the 50 epochs are illustrated in Figure 3.

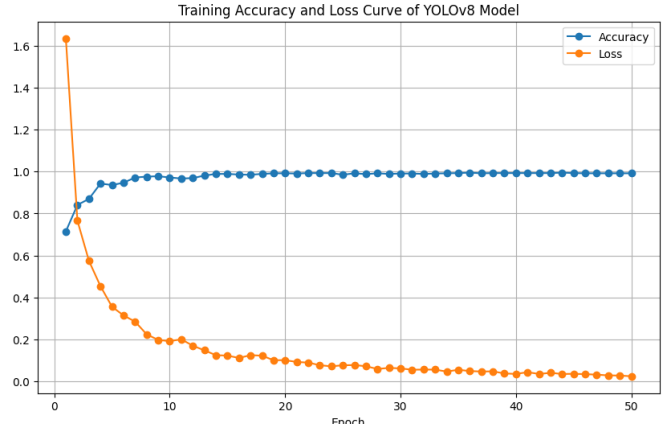

Figure 2 Training Accuracy and Loss Curve of YOLOv8 Modelzz

B. *Example of Classification Result*

ITo illustrate how the model performs classification, an example prediction from a single rice leaf image is presented in Table 3.

TABLE 3.
PROBABILITY OUTPUT FOR ONE IMAGE

Classes	Probability
Bacterial Leaf Blight	0.02
Brown Spot	0.03
Healthy Rice Leaf	0.01
Leaf Blast	0.87
Leaf Scald	0.02
Narrow Brown Leaf Spot	0.01
Rice Hispa	0.02
Sheath Blight	0.02

Based on the probability values generated by the model, the final classification result is determined using the argmax function, which selects the class with the highest probability value.

$$\hat{y} = \arg\max_{i}(p_i) \qquad (5)$$

Where pi denotes the probability of the $i-$ th predicted class.

Since the highest probability value is 0.87, the model predicts the rice leaf image as Leaf Blast.

C. *Model Evaluation Results*

To evaluate the performance of the trained model, a classification report was generated based on the testing dataset containing 1,478 rice leaf images. The evaluation results are presented in Table 4.

TABLE 4
CLASSIFICATION REPORT

Classes	Accuracy	Recall	F1 Score	Support
Bacterial Leaf Blight	0.99	0.98	0.98	166
Brown Spot	0.99	0.99	0.99	221
Healthy Rice Leaf	0.98	0.99	0.99	150
Leaf Blast	0.98	0.98	0.98	250
Leaf Scald	0.98	0.98	0.98	187
Narrow Brown Leaf Spot	0.98	0.98	0.98	128
Rice Hispa	0.99	0.99	0.99	173
Sheath Blight	1.00	0.99	0.99	203
Accuracy			0.99	1478

The evaluation results indicate that the proposed YOLOv8 model achieved excellent classification performance, with precision, recall, and F1-score values exceeding 0.98 for almost all disease classes. The Sheath Blight class achieved the highest precision value of 1.00, demonstrating perfect classification performance for that class. The similarity between macro average and weighted average values (0.99) indicates that the model does not exhibit bias toward certain classes despite variations in dataset size.

D. *Confusion Matrix Analysis*

In addition to the classification report, the model performance is further analyzed using a confusion matrix, which visualizes the relationship between predicted labels and actual labels.

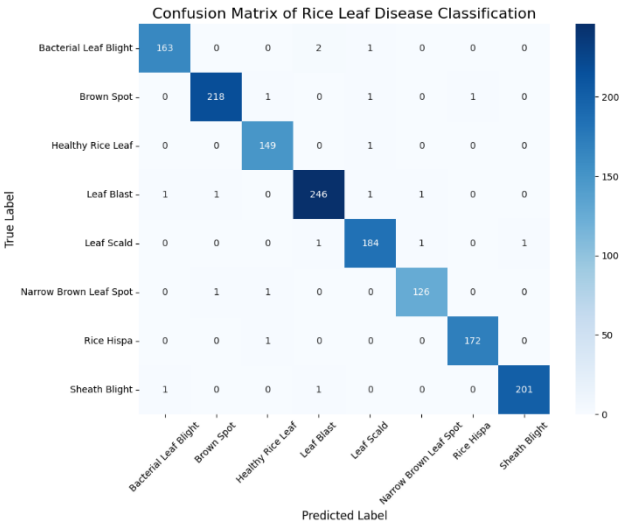

Figure 3 Confusion Matrix of Rice Le Disease Classification

The confusion matrix results indicate that most prediction values are concentrated along the main diagonal, which represents correctly classified samples. For example, the Leaf

Blast class was correctly classified in 246 instances, while Brown Spot was correctly identified in 218 samples. Only a small number of misclassifications were observed between classes with visually similar characteristics. For instance, two samples of Bacterial Leaf Blight were misclassified as Leaf Blast, which may occur due to similarities in lesion patterns and color distribution on rice leaves. Overall, the dominance of values along the diagonal of the confusion matrix indicates that the YOLOv8 model successfully captures distinctive visual features of rice plant diseases.

E. ROC Curve Analysis for Rice Disease Detection

To further evaluate the classification performance of the proposed model, Receiver Operating Characteristic (ROC) curves were generated for each disease class. The ROC curve illustrates the relationship between the True Positive Rate (TPR) and False Positive Rate (FPR), providing insight into the model's ability to distinguish between classes.

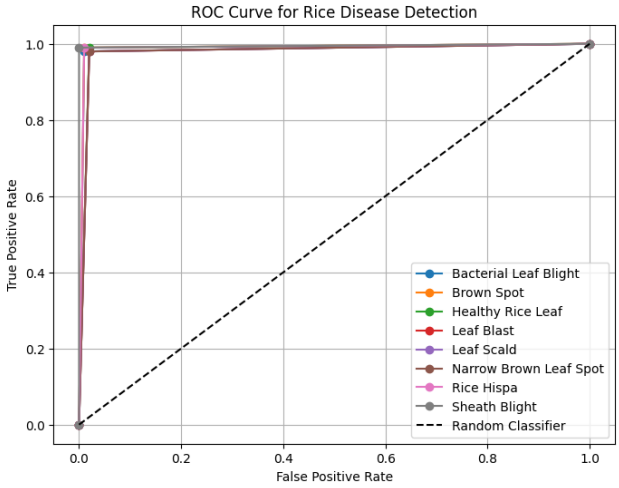

Figure 4 ROC Curve for Rice Disease Detection

The ROC curve results show that most classes achieve Area Under Curve (AUC) values close to 1.0, indicating excellent classification capability. High AUC values suggest that the model can effectively differentiate between disease classes with minimal false positive and false negative rates. This result further confirms the robustness of the YOLOv8 architecture in learning complex visual patterns from rice leaf images.

F. Web-Based System Deployment

After completing the training and evaluation stages, the trained YOLOv8 model was saved in .pt format, which stores the learned weights and parameters of the classification model. The trained model was then integrated into a web-based application to enable automatic classification of rice plant diseases from leaf images. Prior to system implementation, the user interface (UI) was first designed using Figma to produce a simple, responsive, and user-friendly interface. The finalized interface design was then implemented using HTML as the primary structure of the web page, ensuring a lightweight and efficient web application. The backend system was developed using the Flask framework, which serves as the bridge between the web interface and the trained YOLOv8 model. When a user uploads a rice leaf image through the website, the image is sent to the Flask server, where the YOLOv8 model processes the input and performs disease classification. The prediction results are then returned and displayed directly on the web page in real time . The system was developed using a native web framework approach, which was intentionally chosen due to its lightweight architecture and efficient performance. In addition, the website was designed to be responsive, allowing it to be accessed through multiple devices such as desktop computers, laptops, and smartphones without requiring users to install additional applications. Through this deployment approach, the developed system provides not only accurate disease classification but also offers a practical decision-support tool that can assist farmers in identifying rice plant diseases quickly and easily in real agricultural environments. This practical implementation demonstrates that deep learning-based rice disease detection can be effectively integrated into a lightweight web platform, enabling accessible and real-time disease identification for farmers.

G. System Interface Implementation

The web-based rice disease detection system provides several main interface pages designed to facilitate user interaction with the application. These pages were developed with a simple and intuitive layout to ensure ease of use, particularly for farmers as the primary users of the system.

1) Dashboard Page: The dashboard page functions as the main entry point of the system and provides general information about the application. This page presents a brief overview of the system objectives, instructions for using the website, and introductory information related to rice plant disease detection. The interface is designed to be simple and easy to understand so that users can quickly access the available system features. The appearance of the dashboard interface is presented in Figure 5.

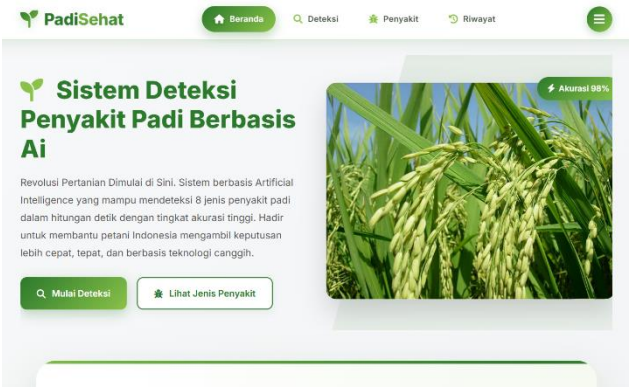

Figure 5 Dashboard Page of the Web-Based Rice Disease Detection System

2) Disease Detection Page: The disease detection page represents the core functionality of the system, where users can perform rice disease classification. On this page, users can upload rice leaf images from their device or capture photos directly using the device camera. The uploaded images are then processed by the trained classification model integrated into the web system, and the prediction results are displayed automatically in real time. The interface of the detection feature is shown in Figure 6.

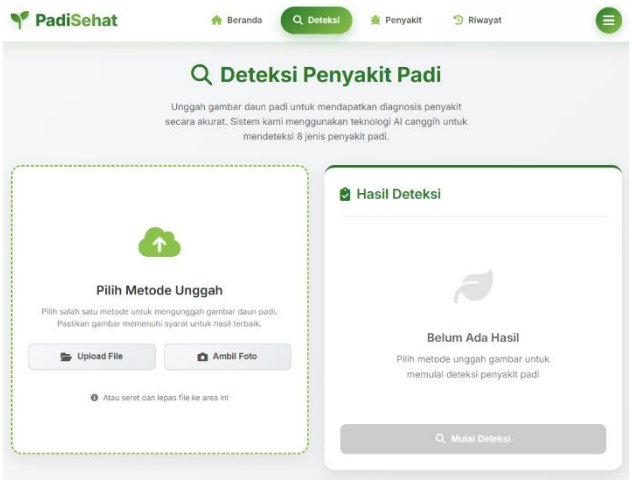

Figure 6 Disease Detection Interface

3) Disease Information Page: The disease information page provides detailed descriptions of the rice plant diseases included in the system dataset. The information presented includes the disease name, visible symptoms on rice leaves, possible causes of the disease, and additional explanations that may help farmers better understand each disease type and take appropriate preventive actions. The appearance of this information page is illustrated in Figure 7.

Figure 7 Disease Information Page

4) Detection History Page: The detection history page stores and displays the results of previous disease detection activities. This page records important information such as the predicted disease class and the detection timestamp, allowing users to review past analyses and monitor rice plant conditions over time. The interface of the detection history feature is shown in Figure 8.

H. System Workflow

The developed system integrates the trained rice disease classification model into a web application environment to enable automatic disease identification through rice leaf image analysis. The system is designed to assist farmers in detecting rice plant diseases quickly and efficiently using a web-based platform that can be accessed through various devices without requiring additional software installation.

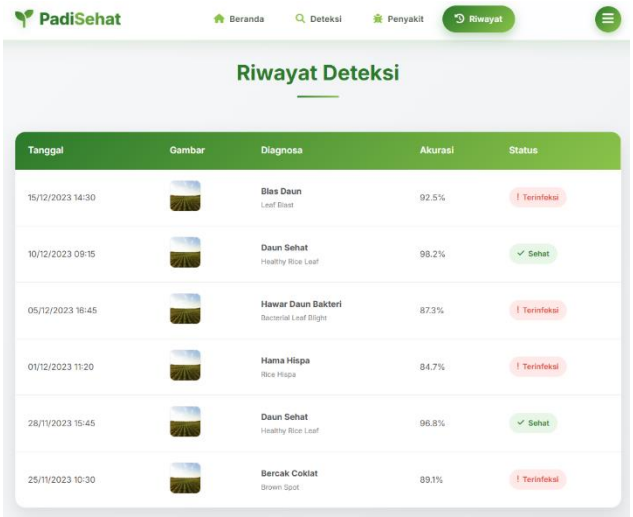

Figure 7 Detection History Page

The detection process begins when users access the website and navigate to the disease detection page. Users can upload a rice leaf image from their device or capture an image directly using the device camera. The uploaded image then undergoes a pre-processing stage, including image resizing to match the input dimensions required by the classification model. After pre-processing, the image is analyzed by the trained model, which extracts visual features from the rice leaf image and classifies the disease based on previously learned patterns. The classification results, including the predicted disease class and confidence score, are then displayed directly on the web interface in real time. Finally, the prediction results are automatically stored in the system database and can be accessed through the detection history page. This feature allows users to review previous detection results and monitor the condition of rice plants over time. Through this workflow, the system provides a practical and accessible tool that can support farmers in conducting early detection and management of rice plant diseases.

IV. CONCLUSION

This study successfully developed a web-based rice plant disease detection system using the YOLOv8n-cls deep learning model. The research began with problem identification through field observations and discussions with farmers to understand the symptoms and impacts of rice plant diseases. The dataset used in this study consisted of rice leaf images from eight disease categories obtained from the Kaggle dataset. The data were pre-processed through image resizing, labeling, and dataset splitting into training, validation, and testing sets to ensure data consistency and improve model performance during the training process. The YOLOv8n-cls model was trained using a GPU-based environment in Google Colaboratory and demonstrated excellent classification performance. The evaluation results showed high values of accuracy, precision, recall, and F1-score, indicating that the model is capable of accurately classifying different types of rice plant diseases. In addition, the model achieved a fast inference speed, making it suitable for real-time applications. To provide practical benefits, the trained model was deployed into a web-based application using the Flask framework as the backend system. The user interface was designed using Figma and implemented using HTML to create a simple, responsive, and user-friendly website. The system provides several main features, including a dashboard page, disease detection page, disease information page, and detection history page, which allow users to detect rice plant diseases and monitor detection results easily through a web browser on various devices.

Overall, the proposed system demonstrates that deep learning technology can be effectively integrated into a lightweight web platform to support early detection of rice plant diseases. This implementation provides a practical decision-support tool that can assist farmers in identifying rice diseases quickly and accurately, thereby supporting better crop management and potentially improving agricultural productivity.

REFERENCES

[1] S. Herliana, S. Ratnaningtyas, S. Nur Arifin, and N. Lawiyah, "Analysis of Indonesia's Food Security Strategy: Rice Price Volatility," *Global Conference on Business and Social Sciences Proceeding*, vol. 14, no. 2, pp. 1–1, Dec. 2022, doi: 10.35609/gcbssproceeding.2022.2(35).

[2] H. Agustina, B. Setiawan, M. Solahuddin, and Sugiyanta, "SRI (Rice Intensification System) water management of rice productivity," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 542, no. 1, p. 012051, Jul. 2020, doi: 10.1088/1755-1315/542/1/012051.

[3] O. Aziz *et al.*, "Irrigation methods affect water productivity, grain yield, and growth responses of rice at different levels of nitrogen," *J. Soil Water Conserv.*, vol. 73, no. 3, pp. 329–336, May 2018, doi: 10.2489/jswc.73.3.329.

[4] M. Barchia, H. Mareja, A. Susatya, and E. Putri, "Sustainability Of Rice Cultivation In Irrigated Paddy Fields In The Highlands Of Bengkulu, Indonesia," in *Proceedings of the 3rd Sriwijaya International Conference on Environmental Issues, SRICOENV 2022, October 5th, 2022, Palembang, South Sumatera, Indonesia*, EAI, 2023. doi: 10.4108/eai.5-10-2022.2328294.

[5] R. Ardinata and W. M. Manurung, "Disaster from water pollution in Indonesia: Unsustainable human interaction with the environment and its social impacts," *ASEAN Natural Disaster Mitigation and Education Journal*, vol. 2, no. 2, Jan. 2025, doi: 10.61511/andmej.v2i2.2025.1478.

[6] D. Desrizal, N. Carlo, and N. Syah, "The Impacts of PETI on the Batang Hari River to the Decline of Water Quality, Land Transfer Function, Socio-Cultural Life and the Community Economy," *Sumatra Journal of Disaster, Geography and Geography Education*, vol. 3, no. 1, pp. 54–61, Jun. 2019, doi: 10.24036/sjdgge.v3i1.182.

[7] D. Kumar *et al.*, "Impact of saline water irrigation on growth, yield and quality of rice (Oryza sativa L.) varieties," *International Journal of Research in Agronomy*, vol. 7, no. 10, pp. 117–122, Oct. 2024, doi: 10.33545/2618060X.2024.v7.i10b.1709.

[8] A. Punia and S. K. Singh, "Contamination of water resources in the mining region," in *Contamination of Water*, Elsevier, 2021, pp. 3–17. doi: 10.1016/B978-0-12-824058-8.00015-3.

[9] P. Prasher and M. Sharma, "An Introduction to Rice Diseases," in *Cereal Diseases: Nanobiotechnological Approaches for Diagnosis and Management*, Singapore: Springer Nature Singapore, 2022, pp. 3–15. doi: 10.1007/978-981-19-3120-8_1.

[10] T. G. Devi, P. Neelamegam, and A. Srinivasan, "Rice plant disease, crop stages endemic and control measures - a survey," *International Journal of Advanced Intelligence Paradigms*, vol. 23, no. 3/4, p. 399, 2022, doi: 10.1504/IJAIP.2022.126698.

[11] Hussain. A and B. Srikaanth. P, "Disease Classification and Detection Techniques in Rice Plant using Deep Learning," in *2022 8th International Conference on Smart Structures and Systems (ICSSS)*, IEEE, Apr. 2022, pp. 1–7. doi: 10.1109/ICSSS54381.2022.9782162.

[12] P. Seelwal and A. Sharma, "Machine Vision Systems for Rice Diseases Detection: A Review," in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, IEEE, Apr. 2022, pp. 1686–1689. doi: 10.1109/ICACITE53722.2022.9823713.

[13] C. Gupta *et al.*, "Deep vision in agriculture: assessing the function of YOLO in the classification of plant leaf diseases (PLDs)," *BioData Min.*, vol. 18, no. 1, p. 91, Nov. 2025, doi: 10.1186/s13040-025-00497-y.

[14] K. Krisdianto, Elta Sonalitha, and Yandhika Surya Akbar Gumilang, "Deteksi penyakit padi menggunakan YOLO," *Uranus : Jurnal Ilmiah Teknik Elektro, Sains dan Informatika*, vol. 2, no. 3, pp. 125–134, Jul. 2024, doi: 10.61132/uranus.v2i3.259.

[15] I. V. S. L. Haritha, M. Harshini, S. Patil, and J. Philip, "Real Time Object Detection using YOLO Algorithm," in *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, IEEE, Dec. 2022, pp. 1465–1468. doi: 10.1109/ICECA55336.2022.10009184.

[16] Md. M. Hasan, "Rice Leaf Disease Dataset," https://www.kaggle.com. [Online]. Available: https://www.kaggle.com/datasets/mdmehedihasan19/rice-leaf-disease-dataset/code/data

[17] S. Jatmika and D. E. Saputra, "Rice Plants Disease Identification Using Deep Learning with Convolutional Neural Network Method," *SinkrOn*, vol. 7, no. 3, pp. 2008–2016, Aug. 2022, doi: 10.33395/sinkron.v7i3.11540.

[18] Ultralytics, "YOLOv8: State-of-the-Art YOLO Models," https://docs.ultralytics.com. Accessed: Dec. 24, 2025. [Online]. Available: https://docs.ultralytics.com/models/yolov8/

[19] P. G. J. and N. K. V., "Google Colaboratory : Tool for Deep Learning and Machine Learning Applications," *Indian Journal of Computer Science*, vol. 6, no. 3–4, p. 23, Aug. 2021, doi: 10.17010/ijcs/2021/v6/i3-4/165408.

[20] A. G. S. M. F. L. A. B. J. C. G. et al. Paszke, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems (NeurIPS)*, Red Hook, NY, USA: Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html

[21] X. Jia *et al.*, "Highly Scalable Deep Learning Training System with Mixed-Precision: Training ImageNet in Four Minutes," Jul. 2018, [Online]. Available: http://arxiv.org/abs/1807.11205

[22] M. Praveena, V. B. Dubisetty, K. V. Varaprasad, M. Rama, P. S. Vadana, and T. S. R. Sai, "An In-Depth Analysis of Deep Learning and Machine Learning Methods for Identifying Rice Leaf Diseases," in *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*, IEEE, Sep. 2023, pp. 951–955. doi: 10.1109/ICOSEC58147.2023.10276335.

[23] M. Okty Dea Pratama and S. Suwarni, "Pengembangan Prototipe Desain User Interface & User Experience (UI/UX) Pada Aplikasi OSS URINDO Menggunakan FIGMA," *Jurnal Teknologi Informasi*, vol. 8, no. 2, pp. 155–166, Dec. 2022, doi: 10.52643/jti.v8i2.2772.

[24] M. Simon, "Manipulating HTML Elements," in *JavaScript for Web Developers*, Berkeley, CA: Apress, 2023, pp. 93–120. doi: 10.1007/978-1-4842-9774-2_3.

[25] C. Patil, "Crop Analyzer Using Deep Learning and Yolo," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 12, no. 3, pp. 3416–3421, Mar. 2024, doi: 10.22214/ijraset.2024.59651.

[26] H. L. Walingkas and P. O. N. Saian, "Penerapan Framework Flask pada Pembangunan Sistem Informasi Pemasok Barang," *Jurnal JTIK (Jurnal Teknologi Informasi dan Komunikasi)*, vol. 7, no. 2, pp. 227–234, Apr. 2023, doi: 10.35870/jtik.v7i2.729.

[27] H. Saputra, K. Muchtar, N. Chitraningrum, A. Andria, and A. Febriana, "Performance evaluation of hyper-parameter tuning automation in YOLOV8 and YOLO-NAS for corn leaf disease detection," *SINERGI*, vol. 29, no. 1, p. 197, Jan. 2025, doi: 10.22441/sinergi.2025.1.018.