

Sustainable Agriculture: Real-time Detection of Rice Leaf Diseases on Mobile Platforms using Lightweight Deep Learning Models

Fitri Damayanti ^{1,*}, Iwan Santosa ², Achmad Imam Sudianto ³, Imamah ¹, and Sri Herawati ¹

¹ Department of Information System, Faculty of Engineering, University of Trunodjoyo Madura, Bangkalan, Indonesia

² Department of Informatics, Faculty of Engineering, University of Trunodjoyo Madura, Bangkalan, Indonesia

³ Department of Mechatronics, Faculty of Engineering, University of Trunodjoyo Madura, Bangkalan, Indonesia

Email: fitrid@trunodjoyo.ac.id (F.D.); iwan.santosa@trunodjoyo.ac.id (I.S.); aimam.sudianto@trunodjoyo.ac.id (A.I.S.); i2m@trunodjoyo.ac.id (I.); sriherawati@trunodjoyo.ac.id (S.H.)

*Corresponding author

Abstract—Rice diseases pose a serious threat to agricultural productivity and food security, especially in Indonesia. This study proposes a rice disease detection system using deep learning-based rice leaf images implemented on mobile devices to provide fast, accurate, and easily accessible diagnoses. Two main architectures were used, namely You Only Look Once version 8 (YOLOv8) for real-time detection and EfficientNet for high-efficiency classification. The dataset consisted of an open-source dataset that had undergone annotation, pre-processing, and augmentation. The models implemented on mobile devices are YOLO and EfficientNet. This research demonstrates the utilization of both models in mobile applications for the real-time detection of illnesses in rice leaves. The outcomes from the EfficientNet model indicate a value of 1 for all classes, except for brown spot and leaf blast, which both yield a value of 0.99. In contrast, YOLOv8 shows an average accuracy of 0.85–0.92 but excels in inference speed, making it suitable for real-time applications. The best model was then deployed using TensorFlow Lite on the RiceLeaf Detection application, which can provide disease diagnoses directly from rice leaf images. This research demonstrates the potential of lightweight Convolutional Neural Networks (CNNs) to improve early detection of rice diseases, support smart farming practices, and contribute to efforts to improve food security.

Keywords—food security, rice disease detection, Convolutional Neural Network (CNN), EfficientNet, YOLOv8, image processing, TensorFlow Lite, mobile application

I. INTRODUCTION

Diseases that attack plants generally damage important parts such as leaves, stems, roots, and fruits. This damage prevents plants from growing optimally, thereby reducing crop quality and production. This condition ultimately leads to food insecurity due to a decrease in the supply of quality food [1]. Globally, plant diseases cause an average

yield loss of 20%–40% each year. This figure is a major factor contributing to food shortages in various regions, which ultimately increases agricultural production costs due to disease control efforts [2]. The Food and Agriculture Organization (FAO) estimates that the world population will reach approximately 9.1 billion by 2050. To meet the food needs of such a large population, global food production needs to be increased by about 70% compared to current production [3]. Therefore, plant diseases pose a very serious threat that must be addressed immediately to maintain the stability of the world's food supply in the future.

Plant diseases can be caused by two main factors. First, biotic factors, which are diseases that arise due to attacks by organisms. Second, abiotic factors, which are disorders caused by unsuitable environmental conditions or nutrient deficiencies in plants [4]. These plant disorders are common, so innovation through technology is needed to eliminate delays in disease diagnosis caused by limited access and knowledge, especially in 3T areas (underdeveloped, frontier, and outermost regions). Through real-time disease detection using mobile phones, it is hoped that farmers can be assisted in resolving these issues immediately. This is important to address because plant diseases can have a serious impact on daily life, particularly in terms of food security. This is especially true for rice, which is the staple food of the Indonesian people [5].

In identifying damage caused by plant diseases, farmers will generally diagnose the problem, but the diagnostic steps involved raise several crucial questions, such as:

- (1) Inconsistency and unreliability: Differences in skill levels and observation can lead to misdiagnosis in the early stages of plant observation.
- (2) Economic impact: Failure to identify infected plant diseases at an early stage will greatly contribute to

reducing crop yields and quality. This will result in economic losses for farmers.

- (3) Food security risks: The inability to control plant diseases will damage the food supply chain, ultimately leading to poverty and stunting. Controlling plant diseases is key to meeting the ever-increasing demand for food.
- (4) Environmental issues: If there are errors or delays in identifying plant diseases (misdiagnosis), farmers tend to rely on excessive use of chemicals to overcome them. This causes an increase in chemical residues in the environment, which ultimately damages soil quality and pollutes the water around agricultural land.

Due to these various problems, an automated system is needed that can detect plant diseases quickly and accurately on a large scale. This system requires technology that helps farmers immediately recognize plant diseases so that appropriate action can be taken immediately. To solve this problem, image processing technology has proven to be effective in several ways, such as feature extraction in images [6]. This process will assist in identification [7] and image classification [8]. This will help in detecting diseases from processed plant images [9]. In deep learning, Convolutional Neural Networks (CNN) have good accuracy and can capture complex image patterns [10]. Through the CNN approach, it is hoped that this technology will be able to detect plant diseases more effectively, thereby reducing economic losses for farmers and preventing environmental damage caused by excessive use of chemicals.

II. LITERATURE REVIEW

The use of CNN in plant disease detection is one of the applications of deep learning. Plant disease detection using the ResNet and VGG-16 Visual Geometry Group models produces high detection accuracy, but it will be less than optimal when applied to mobile devices because these models require high computation [11]. With the development of deep learning, Tensorflow Lite has become a way to optimize the efficiency of models that will be applied on mobile devices. The use of Tensorflow Lite in mobile applications shows a 30% increase in inference time compared to other conventional models [12]. Other studies have implemented MobileNet on Android devices, showing low latency results during real-time detection [13]. Furthermore, subsequent studies emphasize the importance of model conversion to reduce computing power on mobile devices [14]. Although the results of using Lightweight CNN have great potential, several limitations, such as model adjustments to dynamic agricultural conditions and environments, must be addressed [15].

Prior studies utilized EfficientNet for real-time decision-making tasks in a digital environment, emphasizing its little computing expense and appropriateness for latency-sensitive edge inference [16]. A comparative study was conducted that yielded practical model complexity metrics (parameters, runtime behavior) when evaluating residual networks against classical

models; this study demonstrated that residual/Visual Geometry Group (VGG)-style networks necessitate considerably greater computational power and memory than scaled lightweight networks like EfficientNet for analogous tasks [17]. This research utilizes EfficientNet as a model implemented on edge devices in the findings chapter, comparing it to You Only Look Once (YOLO), MobileNet, and ResNet models, since the training outcomes demonstrate that EfficientNet outperforms the other models.

This study aims to develop real-time rice plant disease detection with the help of a Lightweight CNN optimized with TensorFlow Lite to improve inference time and reduce computing power. The application and results of this research are as follows:

- (1) Implementation of Lightweight CNN: Lightweight CNN architectures, such as MobileNet, EfficientNet, or simplified CNN architectures that can be optimally applied to mobile devices, differ from previous ones that used complex models such as VGG or ResNet.
- (2) Real-Time on Mobile Devices: The focus is on real-time inference directly from mobile devices (Android/iOS) without relying on a cloud connection, thereby overcoming the issue of limited internet connectivity in rural areas.
- (3) Model Inference Optimization with TensorFlow Lite: TensorFlow Lite reduces model size and improves efficiency so that applications run smoothly on all devices.

The combination of novelty makes this research significant and relevant, particularly in the application of practical, affordable, and effective Artificial Intelligence (AI) technology to support agriculture in underserved areas.

To better position the proposed approach within the current state-of-the-art, it is important to compare the obtained results with existing mobile-based rice disease detection studies. In comparison, the EfficientNet-based model in this study achieves competitive performance, particularly in controlled dataset conditions. Meanwhile, the YOLOv8 model provides additional capability in localizing disease regions, which is not addressed in pure classification-based approaches. From a deployment perspective, lightweight models are typically evaluated not only in terms of accuracy but also in terms of inference speed and model size. Although this study focuses primarily on accuracy and functional deployment, the use of TensorFlow Lite enables efficient on-device inference, making the proposed system comparable to other mobile-based solutions in practical scenarios.

III. MATERIALS AND METHODS

The methods used in this study are YOLOv8 [18] and EfficientNet [19]. YOLOv8 was used because it has high accuracy and speed [20], low Floating-point Operations Per Second (FLOPs) for mobile deployment, and advantages over previous versions of YOLO with 96.4% precision and 96.1% recall [21]. Meanwhile, EfficientNet and YOLOv8 are implemented as two independent models

within the same application rather than being combined. Each model is executed separately under different usage scenarios to support classification [22] and real-time detection tasks [23]. EfficientNet is applied as an image-level classifier, while YOLOv8 performs object detection to localize and classify disease regions within the image. To explain in detail the methods and steps in this study, Fig. 1 illustrates the workflow for developing and applying the rice plant disease detection model. It starts

with data acquisition, data set annotation, and data preprocessing before exporting in YOLOv8 and Pascal VOC formats [24]. The model is then built and trained using YOLOv8, with performance evaluated and hyperparameter tuned [25]. Models that meet certain performance values will be applied to mobile phone applications, enabling real-time detection of rice plant diseases from the camera. The stages shown in Fig. 1 are explained as follows.

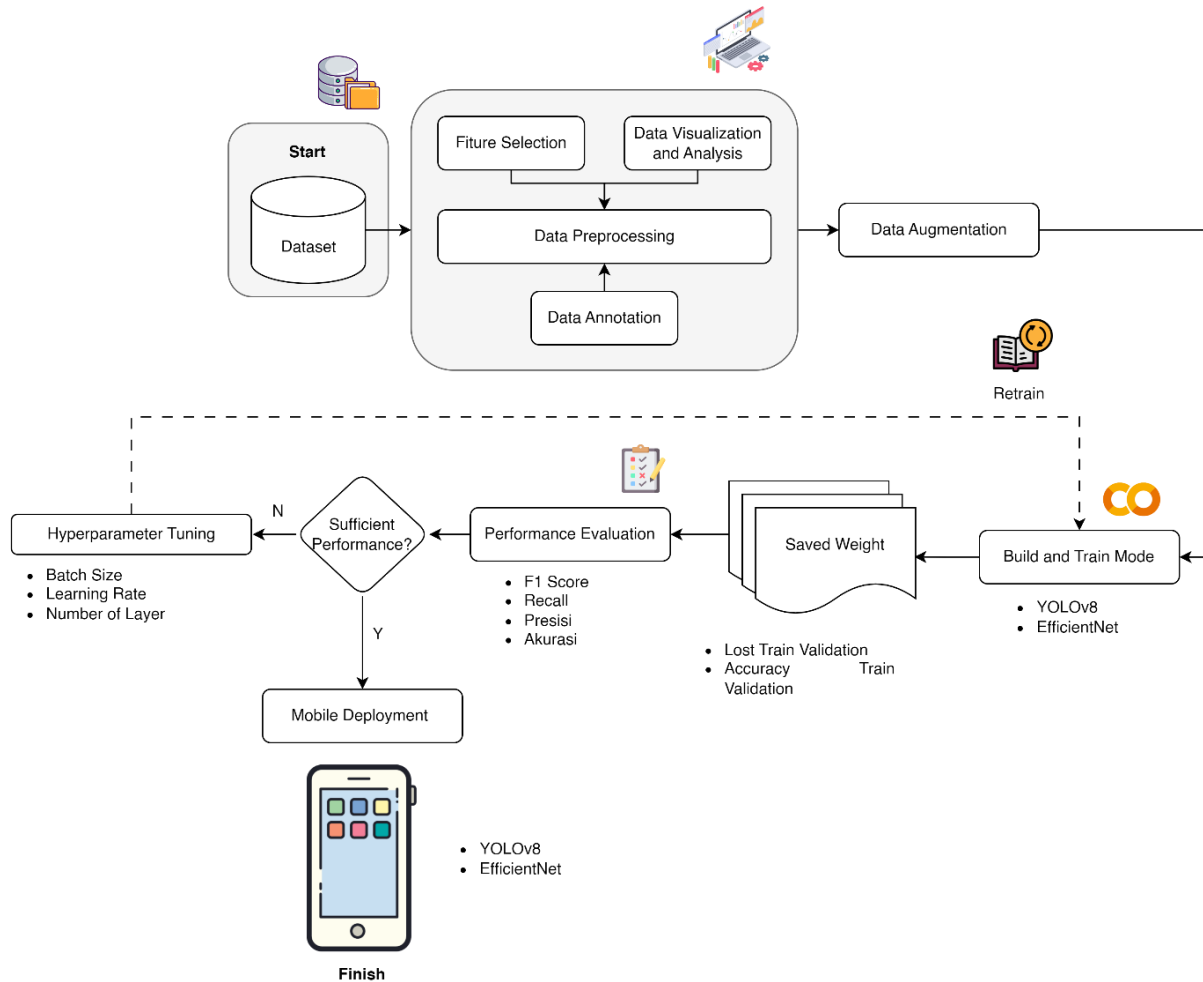


Fig. 1. Research stages.

A. Dataset Collection

Contains a collection of image data used to classify and build models for detecting rice plant diseases. The dataset used is the General dataset accessible through Kaggle (<https://www.kaggle.com/datasets/loki4514/rice-leaf-diseases-detection>). The dataset was partitioned into three subsets: training, validation, and testing, with a ratio of 70:15:15. This division was executed at the outset to prevent data leakage between the subsets. This data has been classified according to the condition of rice leaves. Detailed explanations are provided in Table I. However, specific metadata such as rice cultivars and exact acquisition devices are not explicitly provided in the dataset source. All images are in RGB format with varying resolutions. These resolutions were standardized to

224×224 pixels prior to model training to ensure consistency across experiments. The total number of samples corresponds consistently to the dataset after preprocessing, ensuring that no overlap exists between subsets.

TABLE I. RICE LEAF DATASET

Class	Train Data	Test Data
Bacterial Leaf Blight	2583	376
Brown Spot	3026	380
Healthy Rice Leaf	2576	391
Leaf Blast	3549	363
Leaf Scald	3002	386
Narrow Brown Leaf Spot	2370	382
Neck Blast	2000	322
Rice Hispa	2760	225
Sheath Blight	3207	288
Tungro	1740	310

B. Data Annotation and Feature Selection

Feature selection is a systematic process for analyzing data. Feature selection involves selecting and transforming features from the original data, while data annotation involves labeling data to facilitate the classification of diseases in the image data to be trained [26].

C. Data Pre-Processing

Resizing images and normalizing pixel values. Resizing photos to the same size to standardize image dimensions across the data set [27]. In particular, uniform block sizes are essential for achieving the best results when training CNNs on images.

D. Data Augmentation

Artificially multiplying data to vary existing images. Techniques used include zooming, cropping, changing exposure values, or adding noise [28]. Used to increase model performance effectiveness. The augmentation techniques include:

- (1) Zooming with a range of 0.8–1.2;
- (2) Random cropping with up to 10% area variation;
- (3) Brightness/exposure adjustment within a range of $\pm 20\%$;
- (4) Gaussian noise addition with a variance of 0.01.

Data augmentation was exclusively applied to the training set following the data splitting process, employing techniques such as zooming, cropping, exposure adjustment, and noise addition. The validation and test sets were maintained in their original state to ensure an unbiased evaluation.

E. Model Building

Each CNN has several steps. First, it takes image input from the dataset. After using the convolution matrix, several layers can be added for the next steps, such as the max pooling layer, the flatten layer, and dense layer [29]. Following these layers, we can construct a CNN model for the purpose of detecting plant leaf diseases. This study does not employ cross-validation; instead, an independent hold-out test set is utilized to assess the final model's performance.

For mobile deployment, the trained models were converted into TensorFlow Lite format without applying any quantization technique. The models retain their original 32-bit Floating-Point (FP32) precision to preserve maximum accuracy during inference. This decision was made to avoid potential accuracy degradation that may arise from quantization, especially considering the high classification performance achieved by the model during training.

F. Deep Learning Model

1) YOLO

The first choice is favored due to its high accuracy and performance. As a single-stage model, YOLOv8 outperforms other alternatives in terms of inference speed [30]. Furthermore, YOLOv8 demonstrates exceptional capabilities in classification, image segmentation, and pose estimation.

2) EfficientNet

Second is EfficientNet, a CNN architecture that achieves high accuracy while maintaining computational efficiency. This model was developed by scaling up the basic architecture using a compound scaling method that optimizes the depth, width, and resolution of the network uniformly [31]. Without sacrificing accuracy, previous research has shown that this architecture works well on desktop computers and mobile devices such as tablets and smartphones. The performance of this system is influenced by a number of scaling concepts, such as width scaling, depth scaling, resolution scaling, and complexity scaling. Through scaling techniques, a total of eight models were created: EffectiveNet-B0 to EffectiveNet-B7. The other models (B1 to B7) are scaled variants of the base model, EfficientNet-B0, which serves as the foundation. The number of parameters and the complexity of the model architecture both increase as we move from B0 to B7. Previous research has shown that EfficientNet outperforms ResNet and Inception, which have accuracies below 80%. Our research chose EfficientNet-B1 because of the balance between its size and architectural complexity. EfficientNet-B1 consistently produces the best accuracy and efficiency, based on extensive testing. EfficientNet-B1 was selected as the intermediate model due to its preservation of the excellent classification accuracy inherent to the EfficientNet family, while simultaneously minimizing the number of parameters and FLOPs for efficient on-device inference [32]. Preliminary analysis indicates that EfficientNet variants attain markedly superior accuracy per FLOP in comparison to traditional models like VGG or ResNet (e.g., EfficientNet-B0 ≈ 0.4 Giga Floating-point Operations Per Seconds (GFLOPs) versus ResNet-50 ≈ 4.1 GFLOPs), and subsequent studies validate the appropriateness of EfficientNet for latency-critical edge applications [33]. Consequently, B1 provides an ideal equilibrium between precision and simplicity of implementation on mobile devices. This architecture proposes a combined scaling method, which uses the coefficient ϕ to systematically scale the network width,

depth, and resolution evenly: depth: $d = \alpha^\phi$, width: $w = \beta^\phi$, resolution: $r = \gamma^\phi$, s.t. $\alpha, \beta^2, \gamma^2 \approx 2$, $\alpha \geq 1$, $\beta \geq 1$, $\gamma \geq 1$. Where fast grid search can identify the constants α , β , and γ . The user-defined parameter ϕ intuitively determines how many additional resources are available for model scaling, while α , β , and γ indicate how these additional resources should be allocated to network width, depth, and resolution, respectively. It should be noted that the FLOPs of a typical convolution operation is proportional to d^2, w^2, r^2 ; that is, doubling the network depth will double the FLOPs, while doubling the network width or resolution will quadruple the FLOPs. The scalability of ConvNets using Eq. (3) will roughly increase the total FLOPs by $(\alpha, \beta^2, \gamma^2)^\phi$ because convolution operations generally account for the majority of the computational cost in ConvNets. In the EfficientNet architecture, $\alpha, \beta^2, \gamma^2 \approx 2$ is limited so that the total

FLOPS will increase by approximately 2^ϕ for each addition of ϕ .

G. Deployment

The core feature of this application uses TensorFlow Lite, which integrates real-time plant disease detection models. This research uses the TensorFlow Lite Task Library for a high-level Application Programming Interface (API) that simplifies the process of detecting diseases from camera images and displays annotated results [34]. It is implemented on mobile phones because it can be an easily accessible diagnostic tool, especially in remote areas with limited resources, thereby contributing to improved plant disease detection. The three main parts of YOLOv8 are the backbone, neck, and head. The main innovation of this model is the presence of Spatial Pyramid Pooling Fast (SPPF), which is capable of extracting features at different image scales. Compared to using the traditional Feature Pyramid Network (FPN) module, this architectural model uses the C2f module [35]. This module enhances features in several layers, making it possible to obtain spatial and contextual information from the images of leaves to be detected and classified [36]. After the image is captured, the convolution stage is performed as in a typical CNN. However, there is a difference in the next stage, namely the use of Mobile inverted Bottleneck Convolution (MBConv) equipped with Squeeze-and-Excitation (SE). This combination is used because it can improve accuracy while remaining computationally light [37]. This is important because the final implementation of this research uses mobile devices to detect images of rice leaves directly in the field. MBConv works with a special structure called an inverted bottleneck (narrow-wide-narrow approach) [38], making it more efficient in processing information. The SE module itself forms a network to adaptively select important features based on the relationship between channels.

H. Performance Matrix

The accuracy, precision, recall, and F1-Scores were used to evaluate the performance of every model, including YOLO, EfficientNet, and other architectural models. The degree to which the measurement findings closely resemble the real values is known as accuracy [39]. Accuracy measures, or the percentage of accurate predictions from all data according to Eq. (1), demonstrate how closely measurement findings match actual values [40]. The accuracy of repeated measurements is referred to as precision. Precision, as expressed in Eq. (2), is the percentage of accurate positive predictions among all positive predictions (TP and FP). The ability of the model to identify every positive case that currently exists is referred to as recall. The number of positive cases found by the model using Eq. (3) is listed in recall [41]. Model performance in classification is assessed using the F1-Score, which combines precision and recall. According to Eq. (4), a high F1-Score value shows that the model has great accuracy and recall [42].

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

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

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

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

where, TP = True Positive, TN = True Negative, FP = False Positive, FN = False Negative.

IV. RESULT AND DISCUSSION

The test results show significant differences in the performance of deep learning models used to detect rice leaf diseases. YOLOv8 is evaluated using detection metrics such as mAP, while classification metrics are reported only after mapping detection outputs into class predictions. All class results for each architecture are listed in Table II. Meanwhile, Table III shows the amounts of resources and time required to train the model, along with the confusion matrix results. Table II shows that EfficientNet consistently records precision, recall, F1-Score, and accuracy values close to 1.0 for almost all disease classes. This confirms that this architecture has strong generalization capabilities in identifying leaf image variations, as well as better computational efficiency due to its relatively small weight size compared to other complex models. The following section additionally demonstrates the outcomes of real-work implementation using a smartphone application named RiceLeaf. This program comprises three functionalities for inputting things for detection: Gallery (importing objects through picture files), Camera (photographing objects prior to detection), and Live (detecting photos in real-time without prior capture). This findings section presents Table III, which elucidates the application of two types: macro and weighted, while addressing fairness across classes and realism in relation to dataset frequency.

In contrast, YOLOv8 showed relatively lower performance with an average accuracy score of around 0.85–0.92. However, YOLOv8 still excels in terms of inference speed, making it more suitable for real-time detection on devices with limited computing power. MobileNet provides more stable results than YOLOv8, with metric values close to 1.0 in most classes, confirming its potential as a lightweight model for mobile applications.

Meanwhile, ResNet50 maintains high performance with accuracy in the range of 0.95–0.99. However, its large size makes it less than optimal for deployment on mobile

devices. GTM shows fairly good performance with an accuracy of 0.84–0.94 but still lags behind EfficientNet and MobileNet. Overall, these results confirm that

EfficientNet is the most superior model for accuracy and reliability of detection.

TABLE II. OVERALL ARCHITECTURE RESULTS FOR EACH DISEASE CLASSIFICATION

Class	EfficientNet				YOLOv8				MobileNet				ResNet50				GTM			
	P	R	F	A	P	R	F	A	P	R	F	A	P	R	F	A	P	R	F	A
Bacterial																				
Leaf Blight	1	1	1		0.86	0.9	0.88		1	1	1		0.99	0.98	0.99		0.99	0.99	0.99	
Brown Spot	0.99	0.99	0.99		0.87	0.88	0.87		0.96	0.97	0.97		0.86	0.96	0.91		0.73	0.97	0.833	
Healthy	1	1	1		0.92	0.85	0.88		0.99	0.99	0.99		0.99	0.86	0.92		0.98	0.7	0.82	
Leaf Blast	0.99	0.99	0.99		0.87	0.95	0.91		0.96	0.95	0.95		0.85	0.9	0.87		0.83	0.84	0.84	
Leaf Scald	1	1	1		0.87	0.93	0.90		0.99	0.99	0.90		0.97	0.99	0.98		0.99	0.96	0.98	
Narrow Brown Spot	1	1	1	0.9968	0.89	0.89	0.89	0.8568	0.98	0.98	0.98	0.9868	0.97	0.91	0.94	0.9492	0.95	0.94	0.94	0.915
Neck Blast	1	1	1		0.87	0.9	0.88		1	1	1		1	1	1		1	1	1	
Rice Hispa	1	1	1		0.85	0.94	0.89		1	1	1		0.91	0.99	0.95		0.79	0.98	0.88	
Sheath Blight	1	1	1		0.85	0.86	0.85		1	0.99	1		0.99	0.93	0.96		0.99	0.83	0.9	
Tungro	1	1	1		0.85	0.9	0.87		1	1	1		1	1	1		0.99	0.97	0.98	
Macro avg	1	1	1		0.87	0.9	0.9		0.99	0.99	0.99		0.95	0.95	0.95		0.93	0.92	0.92	
Weighted avg	1	1	1		0.88	0.9	0.91		0.99	0.99	0.99		0.95	0.95	0.95		0.93	0.91	0.92	

P = Precision, R = Recall, F = F1-Score, A = Accuracy, WS = Weight Size (MB).

TABLE III. PERFORMANCE MATRIX AND RESOURCES USED

Metric	ResNet50	Google Teachable Machine	MobileNet V2	EfficientNet B1	YOLOv8
Accuracy	1.00	0.95	0.99	1.00	0.97
Precision (Macro)	1.00	0.96	0.99	1.00	0.87
Recall (Macro)	1.00	0.95	0.99	1.00	0.94
F1-Score (Macro)	1.00	0.95	0.99	1.00	0.90
mAP@0.5	-	-	-	-	0.62
mAP@0.5:0.95	-	-	-	-	0.55
Trainable Parameters	19,716,490	N/A	2,026,698	6,601,470	68,163,201
Training Time (Seconds)	5700	1440	4200	4800	7300
Model Size (MB)	93.6	2.3	25.1	77.3	130.40
Quantized Model Size (KB)	23614.12	2612.83	2610.83	7286.04	26335.40

The observed performance disparity between EfficientNet and YOLOv8 can be attributed to their fundamentally divergent learning objectives. EfficientNet is specifically engineered for image-level classification, enabling the model to exclusively concentrate on discriminative features across the entirety of the leaf image. Conversely, YOLOv8 operates as a one-stage object detection model that concurrently executes localization and classification. This dual-task optimization entails a trade-off, wherein the model must strike a balance between bounding box regression and class prediction. Consequently, a portion of the model's capacity is allocated to spatial localization, which may result in a reduction in classification accuracy compared to a dedicated classifier. Furthermore, small or diffuse disease regions may pose greater challenges for YOLOv8 to localize with precision, thereby further impacting its classification performance.

To conduct this research, specific components are necessary, the results of which are presented in Tables II and III. The hardware and software specifications are outlined in Table IV.

TABLE IV. HARDWARE SPECIFICATION

Name	Details
Central Processing Unit	Intel Xeon
Random Access Memory	13 GB
Graphic Processing Unit	GPU NVIDIA Tesla
Operating System	Windows 11
Python	3.11

A. EfficientNet

To explain how EfficientNet performed during the training and validation processes using the dataset, see Figs. 2 and 3. In the accuracy graph in Fig. 2, the training accuracy value increases sharply from the beginning of the epoch and reaches almost 1.0 at the end of the iteration,

while the validation accuracy increases more steadily to approach 0.9. This indicates that the model can learn effectively from the training data, while maintaining its generalization ability on the validation data.

The loss graph in Fig. 3 reinforces these results. The training loss value decreased significantly from the beginning to the end of training, reaching a value of around 0.1. Meanwhile, the validation loss decreased consistently from 1.2 at the beginning to around 0.3 at the end of the epoch. Although there was a slight increase in loss when fine-tuning began (around the 20th epoch), the overall trend showed a steady decline in both metrics. This phenomenon can be interpreted as an increase in model complexity due to fine-tuning, which temporarily causes fluctuations but ultimately improves detection performance.

Overall, these results show that EfficientNet successfully achieves a balance between high accuracy and validation stability, with no indication of significant overfitting. The combination of high accuracy (nearly 1.0) and consistent loss reduction reinforces the results of the previous table, where EfficientNet proved to be superior to other models in detecting rice leaf diseases with better precision and reliability. Figs. 2 and 3 illustrate the training flow. The graphs in both images illustrate the outcomes achieved during the training of the EfficientNet model, characterized by robust learning and effective generalization, exhibiting results devoid of overfitting or underfitting.

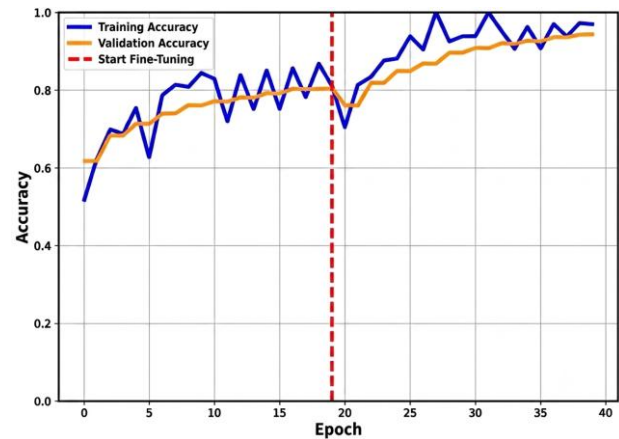


Fig. 2. Accuracy graph during training.

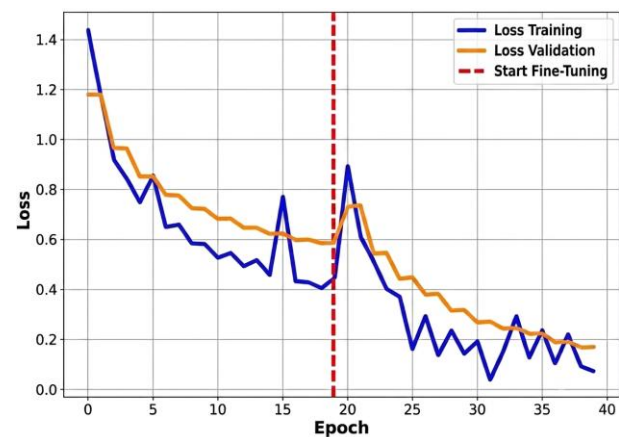


Fig. 3. Loss graph during training.

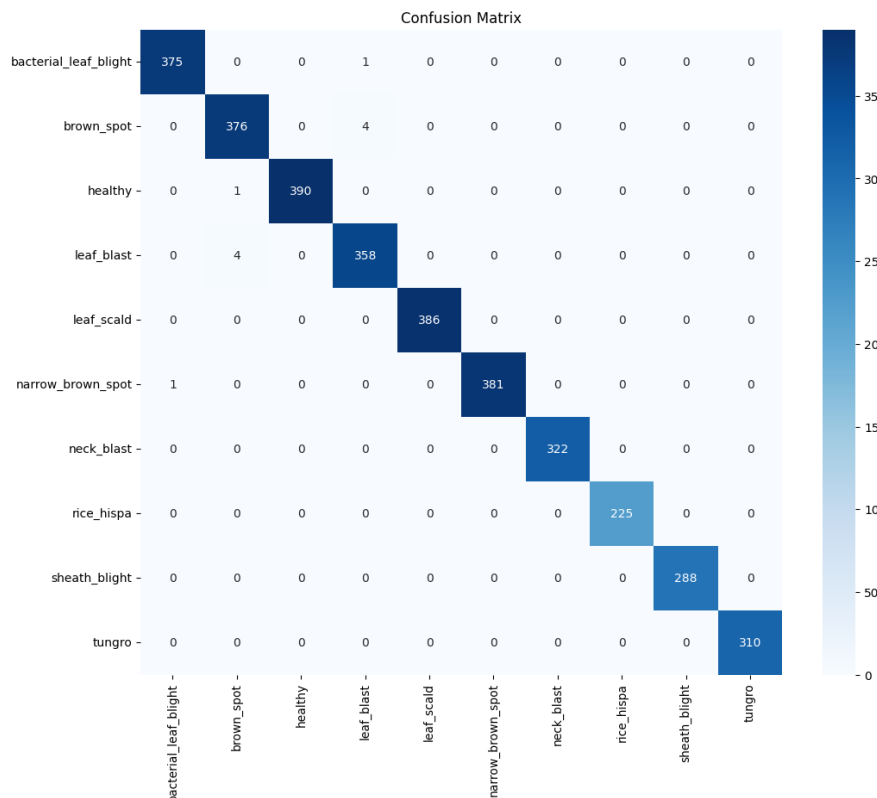


Fig. 4. Confusion matrix results on EfficientNet.

The results of the confusion matrix evaluation on the EfficientNet architecture shown in Fig. 4 indicate the classification performance for ten classes of rice diseases. The dominant diagonal pattern indicates high accuracy, with several classes such as leaf scald, narrow brown spot, tungro, and sheath blight being perfectly identified without any prediction errors. The slightly lower performance observed for Brown Spot and Leaf Blast classes can be attributed to the visual similarity between these disease types. Both exhibit overlapping characteristics such as brownish lesions, irregular shapes, and similar spatial distribution patterns on the leaf surface. In early stages, these diseases may present subtle and low-contrast symptoms, making them difficult to distinguish even for human observers. Specifically, approximately 0.013% of Brown Spot samples are misclassified as Leaf Blast, and 0.013% of Leaf Blast samples are incorrectly predicted as Brown Spot. These errors highlight the difficulty in distinguishing between visually similar disease patterns. This challenge is further amplified under varying lighting conditions and image quality, leading to increased misclassification between these classes.

These findings show that EfficientNet has reliable accuracy for detecting diseases, supported by the results in Table III, so that deployment can be continued in the RiceLeaf Detection mobile application.

EfficientNet's near-perfect macro performance (approximately 1.00) can be attributed to the relatively clean and well-structured dataset, where inter-class variations are sufficiently distinguishable. However, it is crucial to note that this performance is computed on the test set, which differs from the validation set utilized during training.

The validation accuracy depicted in the training curves is slightly lower (approximately 0.9) because it reflects the model's performance during training iterations, where the model is still adapting and regularization effects (e.g., dropout, data augmentation) are active. In contrast, the

final evaluation metrics are obtained after training convergence using the optimized model on the test set, which may result in higher reported performance.

Furthermore, the discrepancy may also indicate that the dataset is relatively less complex, leading to high classification confidence once the model has fully converged.

B. YOLO

To explain the precision and recall results of the YOLOv8 architecture used in the Ultralytics application in detail, the visual results are shown in Figs. 5 and 6. The YOLOv8 training precision graph in Fig. 5 shows a positive relationship between the prediction confidence level and the precision value in rice disease classification. In general, increased confidence results in better precision in most classes. The overall average reaches 0.869, reflecting adequate performance, although it is still below that of EfficientNet.

Meanwhile, the Recall graph of the YOLOv8 training results in Fig. 6 shows that high recall values are achieved at low confidence levels, indicating that the model can detect most positive cases. However, as the confidence threshold increases, recall decreases sharply, resulting in more positive samples being missed. Recall drastically declines when the confidence threshold rises, as Fig. 6 illustrates. Elevated thresholds exclude detections with moderate confidence levels, many of which are still accurate, thereby converting true positives into false negatives. The persistent drop observed across all categories underscores the intrinsic trade-off between precision and recall: elevating precision at higher thresholds inadvertently diminishes recall. The overall average value reaches 0.948, demonstrating the ability of this architecture to capture the majority of positive cases despite a trade-off with a decrease in precision at high thresholds.

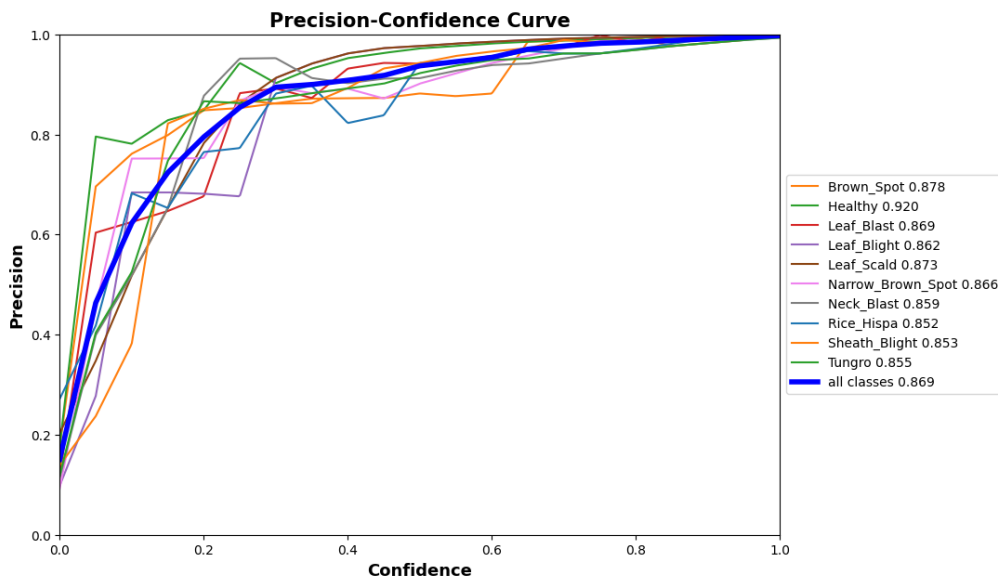


Fig. 5. Precision results on YOLOv8.

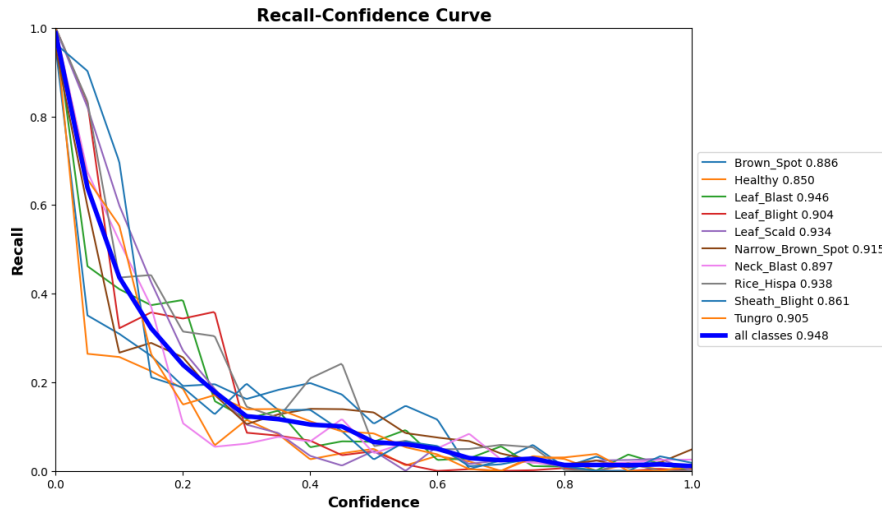


Fig. 6. Recall results on YOLOv8.

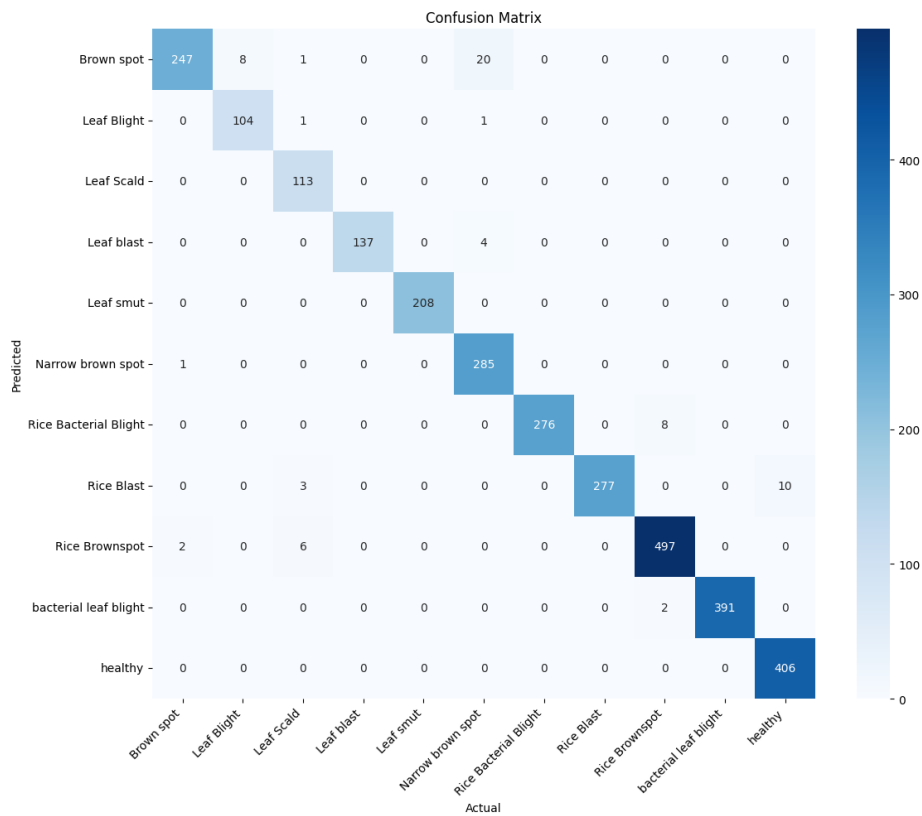


Fig. 7. Confusion matrix results on YOLOv8.

The confusion matrix results from training the YOLOv8 architecture in Fig. 7 show that most rice disease classes can be identified well, as indicated by the dominance of high numbers on the main diagonal. Classes such as Rice Brown Spot (497), Healthy (406), and Bacterial Leaf Blight (391) have very high prediction accuracy. However, there are minor classification errors, for example, Brown Spot is partially misidentified as Leaf Blight and Leaf Scald. In addition, Rice Blast shows some errors in the Healthy class. Overall, YOLOv8 is capable of detecting diseases with good performance, although there is still overlap in classes with similar visual symptoms.

C. Deployment on RiceLeaf Detection App

After training all architectures and determining the model to be used, namely EfficientNet, we deployed the model on an application called RiceLeaf Detection shown in Fig. 8, a smartphone application that diagnoses diseases using only images of rice leaves. Kotlin was used for Android development and Flutter for cross-platform functionality to create an intuitive user experience. Table V provides a detailed list of the specifications of the mobile phone employed in the research and the memory utilized during the application's execution.

TABLE V. MOBILE PHONE SPECIFICATIONS

Chipset	MediaTek Helio G99
RAM	8 GB
Storage	256 GB
Android Version	16
RAM (used by the application)	171 MB

The main feature of this application is supported by TensorFlow Lite, which allows the integration of our object detection model for real-time analysis. With the

help of the TensorFlow Lite Task Library and the high-level programming interface (API) that we use, identifying diseases in rice photos is easy, and the results are displayed with annotations. The feature of this mobile implementation is to provide an easily accessible diagnostic tool, which is particularly useful in remote areas with limited access to resources to improve crop yields. Compared to traditional methods without mobile implementation, this methodology offers significant benefits in practical agricultural applications.

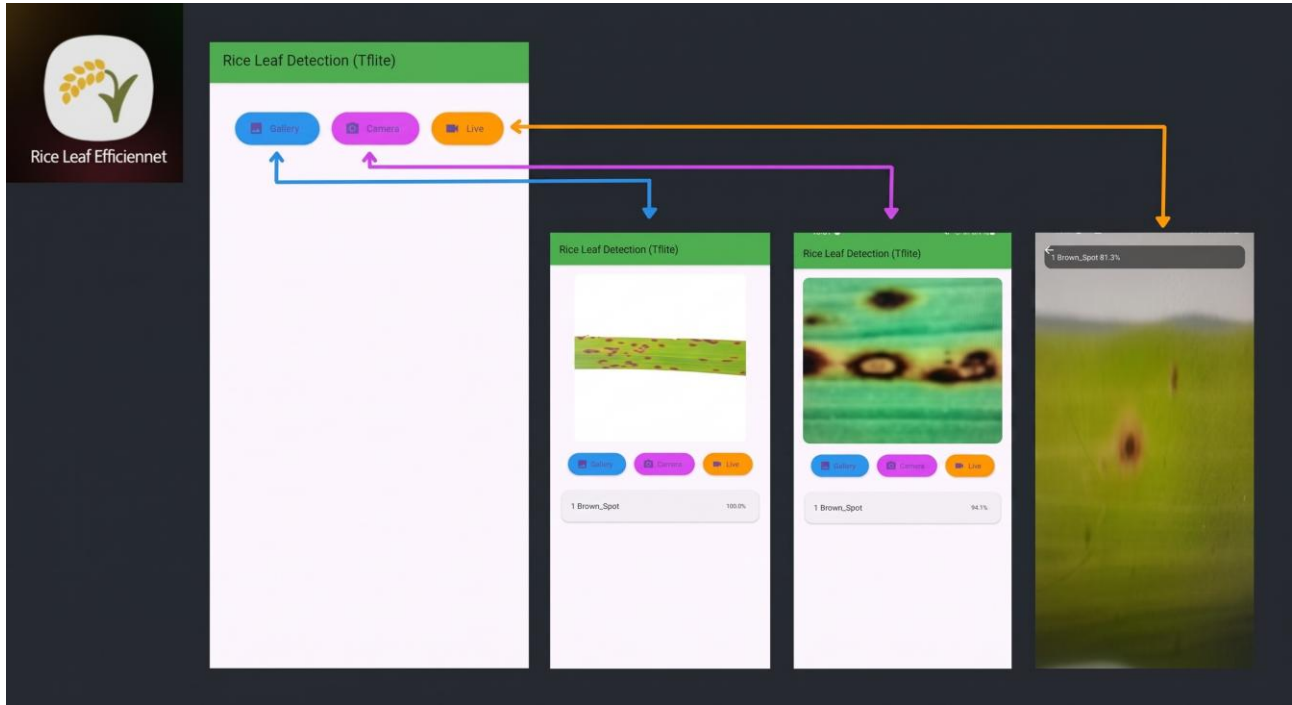


Fig. 8. Rice leaf detection application.

Despite the absence of quantization, the TensorFlow Lite models demonstrated efficient execution on mobile devices. However, it is recognized that employing quantization techniques, such as dynamic or integer quantization, could further enhance inference speed and mitigate memory usage. This presents a promising avenue for future research endeavors.

Although the proposed model exhibits high accuracy in controlled dataset conditions, its direct impact on food security and pesticide reduction cannot yet be fully validated. The current evaluation is limited to laboratory-scale datasets without real-field testing under varying environmental conditions such as lighting, occlusion, and background complexity. Consequently, while the model demonstrates strong potential to support early disease detection, further field-based validation is required to confirm its effectiveness in reducing pesticide usage and improving food security outcomes in practical agricultural settings.

Despite the promising results, several limitations should be acknowledged. Firstly, the dataset utilized in this study originates from an open-source repository, which lacks comprehensive metadata such as rice cultivars, acquisition devices, and environmental conditions. This limitation

restricts the ability to fully evaluate the model's generalization capabilities across diverse agricultural environments. Secondly, the evaluation is primarily conducted on dataset-level testing rather than real-field deployment. In practical settings, factors such as varying illumination, occlusion, complex backgrounds, and different growth stages of the disease may significantly impact model performance. Thirdly, while EfficientNet achieves high classification accuracy, its performance may be influenced by the relatively well-separated dataset classes, which may not fully represent the complexity of real-world scenarios.

V. CONCLUSION

This study proves that the use of deep learning architecture can improve the accuracy and efficiency of rice leaf disease detection. The training results indicate that EfficientNet achieves superior classification accuracy for rice leaf photos compared to YOLOv8. The effective implementation of the TensorFlow Lite model via EfficientNet in the RiceLeaf Detection mobile application illustrates the viability of deploying lightweight CNNs on edge devices. The results indicate that EfficientNet provides higher classification accuracy, while YOLOv8

offers additional advantages in localization and real-time detection. Implementation in the TensorFlow Lite-based RiceLeaf Detection application provides a practical, accessible, and efficient solution for farmers, especially in areas with limited infrastructure. Future work will focus on conducting real-field validation to assess the robustness and reliability of the model in practical agricultural environments. This includes testing under varying lighting conditions, different rice growth stages, and diverse field backgrounds. These findings hold promise for early disease detection and may contribute to the enhancement of agricultural practices. Nevertheless, further validation in real-world conditions is imperative to ascertain its impact on pesticide reduction and food security. Future research will investigate the impact of various quantization strategies on the trade-off between model accuracy, computational efficiency, and edge optimized architecture in mobile deployment scenarios.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Fitri Damayanti: Conceptualization, Methodology, Writing-Original Draft. Iwan Santosa: Software, Methodology, Review & Editing. Achmad Imam Sudianto: Review & Editing, Supervision. Imamah: Review & Editing, Supervision. Sri Herawati: Review & Editing, Supervision; all authors had approved the final version.

FUNDING

Thanks to BIMA (Ministry of Education and Science) for funding this research under contract number B/003/UN46.1/PT.01.03/BIMA/PL/2025.

REFERENCES

- [1] H. N. Fones, D. P. Bebbler, T. M. Chaloner, W. T. Kay, G. Steinberg, and S. J. Gurr, "Threats to global food security from emerging fungal and oomycete crop pathogens," *Nature Food*, vol. 1, no. 6, pp. 332–342, 2020. doi: 10.1038/s43016-020-0075-0
- [2] G. Steinberg and S. Gurr, "Fungi, fungicide discovery and global food security," *Fungal Genetics and Biology*, vol. 144, 103476, 2020. doi: 10.1016/j.fgb.2020.103476
- [3] A. Roy *et al.*, "Towards strategic interventions for global food security in 2050," *Science of the Total Environment*, vol. 954, 176811, 2024. doi: 10.1016/j.scitotenv.2024.176811
- [4] A. Praveen, S. Dubey, S. Singh, and V. K. Sharma, "Abiotic stress tolerance in plants: A fascinating action of defense mechanisms," *3 Biotech*, vol. 13, no. 3, 2023. doi: 10.1007/s13205-023-03519-w
- [5] P. E. R. Prahardini *et al.*, "Characteristic of grain and rice quality from five superior varieties support food sustainability," *IOP Conf. Ser.: Earth Environ. Sci.*, vol. 1253, no. 1, 012016, 2023. doi: 10.1088/1755-1315/1253/1/012016
- [6] F. Damayanti *et al.*, "Restoration of Javanese characters based on Wasserstein generative adversarial network-gradient penalty," *Mathematical Modelling of Engineering Problems*, vol. 11, no. 12, 2024. doi: 10.18280/mmep.111223
- [7] F. Damayanti and A. Rachmad, "Recognizing gender through facial image using support vector machine," *Journal of Theoretical and Applied Information Technology*, vol. 30, no. 3, 2016.
- [8] W. Setiawan and F. Damayanti, "Fundus image classification using two dimensional linear discriminant analysis and support vector machine," *Int. J. Adv. Eng., Manag. Sci.*, vol. 2, no. 10, 2016.
- [9] F. Damayanti *et al.*, "Identification of Madura tobacco leaf disease using gray-level co-occurrence matrix, color moments and Naïve Bayes," *J. Phys.: Conf. Ser.*, vol. 1477, no. 5, 052054, 2020. doi: 10.1088/1742-6596/1477/5/052054
- [10] A. I. Sudianto, M. A. Muslim, and M. Rusli, "Sensor fusion using model predictive control for differential dual wheeled robot," *Kinetik*, vol. 8, no. 1, 2023. doi: 10.22219/kinetik.v8i1.1614
- [11] B. Balaji, T. S. Murthy, and R. Kuchipudi, "A comparative study on plant disease detection and classification using deep learning approaches," *Int. J. Image, Graphics Signal Process.*, vol. 15, no. 3, pp. 48–59, 2023. doi: 10.5815/ijigsp.2023.03.04
- [12] S. E. Adi and A. J. Casson, "Design and optimization of a TensorFlow Lite deep learning neural network for human activity recognition on a smartphone," in *Proc. IEEE EMBS*, 2021. doi: 10.1109/EMBC46164.2021.9629549
- [13] C. Nigam *et al.*, "Real-time object detection on edge devices using mobile neural networks," in *Proc. IITCEE*, 2024, pp. 1–4. doi: 10.1109/IITCEE59897.2024.10467220
- [14] S. Naveen and M. R. Kounte, "Compact optimized deep learning model for edge: A review," *Int. J. Electr. Comput. Eng.*, vol. 13, no. 6, pp. 6904–6912, 2023. doi: 10.11591/ijece.v13i6.pp6904-6912
- [15] W. Feng *et al.*, "Lightweight isotropic convolutional neural network for plant disease identification," *Agronomy*, vol. 13, no. 7, 1849, 2023. doi: 10.3390/agronomy13071849
- [16] W. Cai *et al.*, "Comparison of residual network and other classical models for classification of interlayer distresses in pavement," *Applied Sciences*, vol. 14, no. 15, 6568, 2024. doi: 10.3390/app14156568
- [17] Q. Zia *et al.*, "Optimized real-time decision making with EfficientNet in digital twin-based vehicular networks," *Electronics*, vol. 14, no. 6, 1084, 2025. doi: 10.3390/electronics14061084
- [18] J. M. Kimeu *et al.*, "Deep learning-based mobile application for the enhancement of pneumonia medical imaging analysis," *Informatics in Medicine Unlocked*, vol. 50, 101582, 2024. doi: 10.1016/j.imu.2024.101582
- [19] G. G. Yadao, O. M. Y. Julkipli, and C. O. Manlises, "Performance analysis of EfficientNet for rice grain quality control—An evaluation against YOLOv7 and YOLOv8," in *Proc. ICICT*, 2024, pp. 93–98. doi: 10.1109/ICICT62343.2024.00021
- [20] A. Evan *et al.*, "Detection and counting of grape leaves using YOLOv8 via TFLite on mobile applications," in *Proc. IEIT*, 2024, pp. 246–251. doi: 10.1109/IEIT64341.2024.10763328
- [21] L. Wang *et al.*, "Improved YOLOv8n based potato seedling detection in UAV images," *Frontiers in Plant Science*, vol. 15, 2024. doi: 10.3389/fpls.2024.1387350
- [22] R. Wang *et al.*, "EfficientNet-YOLOv5: Improved YOLOv5 based on EfficientNet backbone," in *Proc. UV*, 2022. doi: 10.1109/UV56588.2022.10185489
- [23] D. Bordoloi *et al.*, "YOLOv5 and EfficientNet for oral cancer diagnosis," in *Proc. APCIT*, 2024. doi: 10.1109/APCIT62007.2024.10673457
- [24] D. Cheng, Z. Zhao, and J. Feng, "Rice diseases identification based on improved YOLOv7-Tiny," *Agriculture*, vol. 14, no. 5, 709, 2024. doi: 10.3390/agriculture14050709
- [25] H. Yang *et al.*, "Real-time rice disease spot segmentation using improved YOLOv8," in *Proc. MLISE*, 2024. doi: 10.1109/MLISE62164.2024.10674034
- [26] L. Zhang *et al.*, "CrossConvPyramid: Deep multimodal fusion for epileptic MEG spike detection," *IEEE J. Biomed. Health Inform.*, 2025. doi: 10.1109/JBHI.2025.3538582
- [27] F. Alshomrani, "Unified pipeline for brain tumor classification and segmentation," *Life*, vol. 14, no. 9, 1143, 2024. doi: 10.3390/life14091143
- [28] F. J. Moreno-Barea *et al.*, "Application of data augmentation techniques towards metabolomics," *Computers in Biology and Medicine*, vol. 148, 105916, 2022. doi: 10.1016/j.combiomed.2022.105916
- [29] J. A. Pandian *et al.*, "Plant disease detection using deep convolutional neural network," *Applied Sciences*, vol. 12, no. 14, 6982, 2022. doi: 10.3390/app12146982
- [30] A. M. Moraes *et al.*, "Effectiveness of YOLO architectures in tree detection," *Standards*, vol. 5, no. 1, 9, 2025. doi: 10.3390/standards5010009
- [31] N. Mandapaka *et al.*, "Classification of paintings using EfficientNet," in *Proc. ODICON*, 2024. doi: 10.1109/ODICON62106.2024.10797572

- [32] Q. Zia *et al.*, “Optimized real-time decision making with EfficientNet in vehicular networks,” *Electronics*, vol. 14, no. 6, 1084, 2025.
- [33] M. Tan and Q. Le, “EfficientNet: Rethinking model scaling for CNNs,” in *Proc. Neural Networks. International Conference on Machine Learning*, 2020.
- [34] M. Reda *et al.*, “AgroAid: Mobile app for plant disease classification using TFLite,” *Informatics*, vol. 9, no. 3, 55, 2022. doi: 10.3390/informatics9030055
- [35] W. Xie *et al.*, “Detection algorithm for citrus Huanglongbing disease using YOLOv8n,” *Sensors*, vol. 24, no. 14, 4448, 2024. doi: 10.3390/s24144448
- [36] M. Zhang *et al.*, “Mulberry leaf disease detection using improved YOLOv8,” *Forests*, vol. 15, no. 7, 1188, 2024. doi: 10.3390/f15071188
- [37] T. Ahmad *et al.*, “EfficientNet-based recognition of peach plant diseases,” *Computers, Materials & Continua*, vol. 71, no. 1, pp. 2073–2089, 2022. doi: 10.32604/cmc.2022.018961
- [38] Q. Yang, S. Duan, and L. Wang, “Efficient identification of apple leaf diseases,” *Agronomy*, vol. 12, no. 11, 2784, 2022. doi: 10.3390/agronomy12112784
- [39] A. Rachmad *et al.*, “CNN-based classification of corn leaf disease,” *Math. Modelling Eng. Problems*, vol. 10, no. 2, pp. 530–536, 2023. doi: 10.18280/mmep.100220
- [40] S. Rochman *et al.*, “Classification of salt quality using machine learning,” *Math. Modelling Eng. Problems*, vol. 11, no. 4, pp. 1005–1012, 2024. doi: 10.18280/mmep.110417
- [41] S. S. Putro *et al.*, “Comparison of SAW and WP methods for agricultural land selection,” *Commun. Math. Biol. Neurosci.*, 2021. doi: 10.28919/cmbn/5820
- [42] A. Rachmad *et al.*, “Impact of convolutional layer selection in ResNet-50v2,” *Ingénierie des Systèmes d’Information*, vol. 30, no. 6, 2025. doi: 10.18280/isi.300609

Copyright © 2026 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).