JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

Implementation of YOLO11 for Disease Detection in Strawberry Plants Based on Android Application

Yosea Mervandy Sugiarto[1], Aditya Dwi Putro W[1,*], Abednego Dwi Septiadi[2]

[1] Faculty of Informatics, Informatics Engineering, Telkom University, Purwokerto, Indonesia
[2] Faculty of Informatics, Software Engineering, Telkom University, Purwokerto, Indonesia
Email: [1]yoseamervandy@student.telkomuniversity.ac.id, [2*]adityaw@telkomuniversity.ac.id, [3]abednego@telkomuniversity.ac.id
Corresponding Author Email: adityaw@telkomuniversity.ac.id
Submitted **27-01-2026;** Accepted **02-06-2026;** Published **30-06-2026**

Abstract

A Diseases in strawberry plants, particularly leaf spot and powdery mildew, represent major challenges that can diminish fruit quality and production quantity. Manual diagnosis by farmers is often subjective, time-consuming, and prone to error. This research aims to develop an automated strawberry disease detection system by implementing and comparing three variants of the latest deep learning architecture, YOLO11n (Nano), YOLO11s (Small), and YOLO11m (Medium), into an Android-based application. The results indicate that the YOLO11n (Nano) variant, as the baseline, provides the most optimal performance for mobile use, achieving a mean Average Precision (mAP@50) of 91.7%, a precision of 0.888, and a recall of 0.841. After integration into Android devices using the TensorFlow Lite format, the model recorded a real-time inference time ranging from 104-125 ms at a speed of 8 FPS. This study contributes an empirical framework for deploying cutting-edge deep learning models on resource-constrained edge devices, establishing that YOLO11n effectively bridges the gap between state-of-the-art detection accuracy and mobile operational efficiency. Furthermore, it provides a practical roadmap for the digital transformation of early-stage crop monitoring, enabling farmers to perform reliable, real-time diagnostics directly in the field.

Keywords: Deep Learning; Android; Object Detection; Strawberry Diseases; Tensorflow Lite; YOLO11

1. INTRODUCTION

Indonesia is a tropical country with vast potential for developing various horticultural commodities, one of which is the strawberry (*Fragaria x ananassa*). Strawberry plants are high-value horticultural crops extensively cultivated in highland regions. One prominent area known for strawberry production is Serang Village, Karangreja District, Purbalingga Regency, situated at an altitude of approximately 650–1,300 meters above sea level. Serang Village possesses fertile soil conditions and a supportive climate, making it a primary producer of vegetables and fruits in Central Java. Unfortunately, various types of diseases have caused a decline in strawberry production in this region. According to the Central Bureau of Statistics (*Badan Pusat Statistik*), strawberry production in Central Java has decreased significantly in recent years. In 2020, production reached 1,053 tons, but it plummeted to 608 tons by 2023, representing a 42.2% decline over a three-year period[1].

Diseases infecting strawberry plants have become a crucial issue in the agricultural industry, leading to substantial productivity losses. Several major diseases frequently attack strawberry plants, such as leaf spot and powdery mildew [2]. Leaf spot, caused by the fungus *Mycosphaerella fragariae*, often appears as small purple spots on the upper surface of the leaves, which then develop into centers colored gray or white. If left untreated, this infection will massively disrupt the photosynthesis process. This disease is a common problem occurring at Mulia Farm, Serang Village, causing farmers to experience crop failure. This directly impacts their income and the sustainability of their agricultural businesses . In addressing this issue, farmers generally still rely on manual techniques to identify the diseases infecting their strawberry plants[3]. However, manual methods have significant limitations in terms of objectivity; a diagnosis can vary from one farmer to another depending on their respective field experience.

Regrettably, manual methods are often inaccurate and time-consuming, leading to late and ineffective disease management. A delay of just a few days can allow fungal spores to spread across the entire field via wind or water splashes. Therefore, early-stage disease detection is essential to avoid losses in quality, quantity, and finances. Image processing technology can serve as an early detection solution based on the visual parameters of leaf images, with the hope that strawberry plants are not treated too late if the disease is detected at an early age [4]. Digital image processing methods enable disease detection based on visual characteristics present on the leaf, such as pigmentation changes, shape anomalies, and texture degradation. By utilizing algorithms trained to recognize specific patterns of various disease types, the identification process can be carried out automatically, quickly, and accurately [5].

One of the most revolutionary approaches in this field is the use of *deep learning* technology. As a branch of artificial intelligence, *deep learning* utilizes layered artificial neural network structures to learn complex features from image data. This technology is capable of extracting deep information from digital images more effectively than conventional algorithms that rely on manual feature extraction. An architecture highly suitable for this task is You Only Look Once (YOLO) [6]. The latest version of the YOLO model, YOLO11, is known for its superior performance due to optimizations in the network backbone and neck structures[7]. YOLO11 is capable of classifying and localizing objects within an image with *real-time* speed, making it an ideal tool for agricultural applications such as detecting strawberry plant diseases in the field, which often feature complex visual backgrounds [8].

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

The technical advantages of YOLO11 lie in the use of C3k2 blocks and a *sequential attention* mechanism, which allow the model to focus on very small infection areas on the leaf without being distracted by soil textures or surrounding weeds . This addresses the weaknesses found in previous YOLO versions, which often experienced a decrease in accuracy during *small object detection*. Furthermore, the integration of this model into mobile devices requires strict parameter efficiency. The use of the Python programming language with the Ultralytics library and the PyTorch *framework* facilitates an optimal model training process. Subsequently, the model conversion process to the TensorFlow Lite (TFLite) format is performed so that the heavy model can run lightly on the Android operating system without drastically reducing accuracy [9].

Based on a literature review, a significant research gap remains in the implementation of object detection for smart agriculture. Previous studies predominantly utilized Convolutional Neural Networks or earlier YOLO architectures (e.g., YOLOv5, YOLOv8) and evaluated their models in high-resource computing environments [3]. Furthermore, existing mobile agricultural solutions often depend on cloud-based inferences requiring stable internet connections [10], which is a severe constraint in rural plantation areas like Serang Village. This research fills that gap by providing an offline, on-device inference solution by optimizing the latest YOLO11 architecture specifically for strawberry diseases..

The primary objective of this research is to develop a strawberry plant disease detection system using the YOLO11 architecture, which is then implemented into an Android application. To determine the most efficient method for mobile deployment, this research conducts a rigorous comparative analysis of three YOLO11 variants: YOLO11n (Nano), YOLO11s (Small), and YOLO11m (Medium). By evaluating these variants based on mean Average Precision and real-time inference speed, the most optimal model is selected and integrated using the TensorFlow Lite format. With this comparative and practical approach, it is expected that the disease identification process can be carried out quickly, accurately, and offline directly through mobile devices by farmers. Ultimately, this enables timely disease management to minimize crop yield losses and increase the productivity of sustainable strawberry farming in the Central Java region.

2. Research Methodology

This research was conducted through a systematic and structured framework to ensure the development of an accurate, efficient, and mobile-ready disease detection model. The comprehensive sequence of the research stages is illustrated in Figure 1.

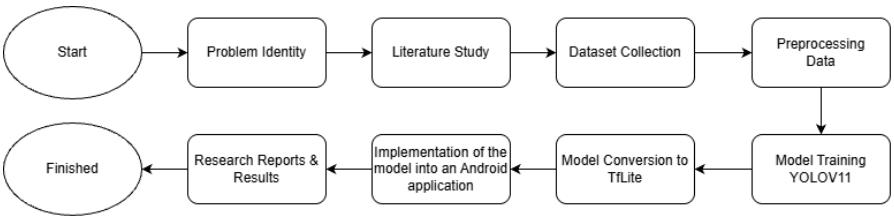

Figure 1. Research Stages

Based on Figure 1, the research methodology is divided into several sequential phases: problem identification, dataset collection and preprocessing, comparative YOLO11 model training, model conversion to TensorFlow Lite, Android application implementation, and final system testing. The utilization of a structured deep learning pipeline is essential for computer vision tasks in agriculture, as it minimizes empirical bias and enhances reproducibility compared to conventional manual observation methods [11].

Furthermore, the core of this methodology relies heavily on the YOLO11 architecture. As highlighted in recent literature, the YOLO framework is widely adopted in agricultural object detection due to its single-shot detection mechanism, which efficiently balances high mean Average Precision (mAP) with low inference latency [12]. By following the stages outlined in Figure 1, the raw visual data is systematically transformed into an optimized, deployable mobile solution.

2.1. Research Stages

Based on the workflow depicted in Figure 1, the procedure follows an intelligent system development pipeline consisting of several sequential phases. The first stage begins with a literature study and field observations to identify the critical issues regarding the decline in strawberry production due to fungal diseases. The second stage is the collection of 1,688 primary leaf images, covering healthy, leaf spot, and powdery mildew conditions to serve as the ground truth. The third stage involves data preprocessing and labeling using Roboflow to meet standard object detection formats. The fourth stage is the core training process utilizing the YOLO11 architecture, which was selected for its state-of-the-art ability to extract multi-scale features with minimal computational overhead. The fifth stage focuses on quantitative model evaluation and conversion into the TensorFlow Lite format to ensure compatibility with mobile hardware. The final stage is the implementation and functionality validation on an Android application using the Black-box Testing method to ensure the tool's reliability for field use by farmers. [13].

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

2.2. Dataset Collection

The data used in this study is primary data obtained through direct image capturing at Mulia Plantation, Serang Village, as well as relevant secondary data from public repositories. A total dataset of 1,688 digital images in *.jpg* format was successfully collected. The dataset is categorized into three main classes: Healthy leaves with 599 images, Leaf Spot with 556 images, and Powdery Mildew with 533 images. Data collection was performed using a smartphone camera with variations in viewing angles, distances ranging from 10–30 cm, and different light intensity conditions to simulate the actual usage environment in the plantation. The entire primary data collection process was conducted between 09:00 and 10:00 WIB to obtain optimal and consistent natural lighting quality. Once the data was gathered, an annotation process was carried out using the *Roboflow* platform. This process produced bounding box coordinates stored in text file format, compliant with the YOLO training data structure. The quality of annotation is crucial as the model learns the visual characteristics of fungal pathogens, such as *Mycosphaerella fragariae* and *Podosphaera aphanis*, based on the defined areas. Overall, this annotation process resulted in 4,679 annotation labels distributed across the entire dataset, with the Healthy class being the most dominant, totaling 2,254 annotations.

2.3. Data Preprocessing and Augmentation

The dataset is systematically divided into three subsets with a ratio of 70% for training data, 20% for validation data, and 10% for testing data. This splitting strategy is a fundamental practice in deep learning to ensure the model has sufficient data to learn underlying patterns while maintaining an adequate independent set to evaluate its generalization capability without bias [14]. The specific distribution of the initial data split consists of 1,181 images for training, 330 images for validation, and 177 images for testing. To mitigate the risk of overfitting and enhance model robustness, data augmentation techniques are applied exclusively to the training set. Augmentation artificially expands the dataset by simulating various field conditions, allowing the neural network to become invariant to orientation and lighting changes commonly encountered in outdoor agricultural environments [15]. The augmentation techniques employed in this research include random rotation, horizontal flipping, as well as saturation and brightness adjustments. The detailed outcome of this augmentation process is presented in Table 1.

Table 1. Data Augmentation Results

	Percentage	Initial Amount	Augmentation	Final amount
Training	70%	1.181	Ya	3543
Validation	20%	330	Tidak	330
Testing	10%	177	Tidak	177
Total	100%	1688	-	4050

As shown in Table 2, applying these augmentation techniques successfully increased the number of training images threefold, from the initial 1,181 images to 3,543 images. Consequently, the total size of the dataset utilized for the entire modeling process reaches 4,050 images. This expanded dataset provides a more comprehensive learning foundation for the YOLO11 architecture, significantly improving its ability to accurately detect leaf spots and powdery mildew under varying visual conditions.

2.4. YOLO11 Architecture Implementation

The core of the disease detection system relies on training the preprocessed dataset using the YOLO11 architecture. The complete sequence of the model training and evaluation process is visually depicted in Figure 2.

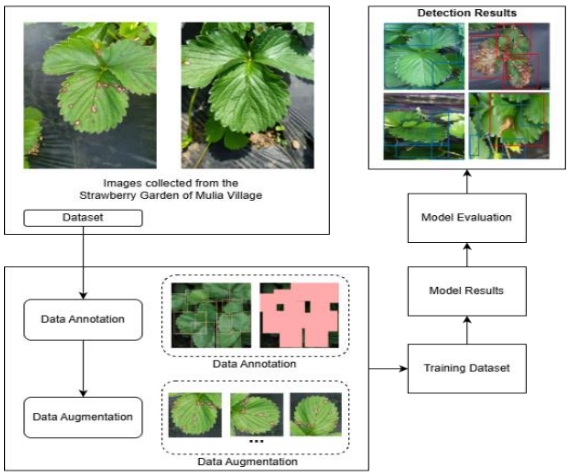

Figure 2. Illustration of Work Stages

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

As illustrated in Figure 2, the model training process involves systematically feeding the augmented dataset into the YOLO11 algorithm, evaluating its output, and iteratively adjusting the network weights until optimal convergence is achieved. The training was conducted using the Google Colab environment, supported by GPU acceleration to significantly expedite computational time and handle the intensive matrix multiplications required by deep neural networks [11].

To determine the most optimal model for mobile deployment, the training was performed comparatively on three architectural variants: YOLO11n (Nano), YOLO11s (Small), and YOLO11m (Medium). This comparative approach is crucial in edge computing research to identify the best trade-off between detection accuracy and inference speed under limited hardware constraints [16]. The training phase was meticulously configured with specific hyperparameters to maximize learning efficiency. The batch size was set to 16 to balance GPU memory utilization and gradient stability, while a total of 120 epochs was established to ensure the model reached a stable convergence point without experiencing overfitting. The initial learning rate was set to 0.001. Furthermore, the researchers employed the AdamW (Adam with Decoupled Weight Decay) optimizer. AdamW was specifically selected because it manages neural weight updates through more effective weight decay regulation, preventing the model from relying too heavily on specific visual features and thus improving overall generalization on complex agricultural image data [4]. During the training iterations, model performance was continuously monitored and evaluated through quantitative metrics, namely mean Average Precision, Precision, and Recall. This rigorous monitoring ensures that the final model weights exported from this stage are highly capable of localizing and classifying strawberry leaf diseases accurately.

2.5. Model Conversion and Android Implementation

To implement the model into an Android application, a conversion process to the TensorFlow Lite format was executed. TFLite is a lightweight framework specifically engineered for edge devices, enabling high-performance deep learning inference with a minimal memory footprint. In the initial stage, the model was converted while maintaining the Float32 precision base to ensure that the detection capabilities on the Android device remained identical to the performance achieved during the training phase in the GPU environment [10].

The converted model was subsequently integrated into a custom-built mobile application. To provide an intuitive user experience for farmers, the application's interface was carefully mapped out before the development phase. The structural design and interface layout of the proposed Android application are illustrated in the wireframes in Figure 3.

Figure 3. Android Application Wireframes

Based on the wireframes presented in Figure 3, the application features a streamlined navigation flow, allowing users to effortlessly access the real-time camera scanning feature or perform static image analysis from the device's gallery. The application was developed using the Kotlin programming language, paired with the Jetpack Compose framework to construct a reactive and declarative user interface[17]. The integration of the TFLite model into the real-time detection feature—as depicted in the scanning wireframe—was achieved utilizing the CameraX library. CameraX was implemented because it efficiently manages the camera lifecycle and streams high-resolution image frames seamlessly to the TFLite interpreter. Crucially, the system is designed to perform strictly on-device inference. This means that the entire computational processing of the strawberry leaf images is conducted locally within the smartphone's

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

hardware, completely eliminating the need for data transmission to an external cloud server. This offline capability directly addresses the internet connectivity constraints often faced in rural agricultural areas like Serang Village, ensuring the application remains highly reliable in the field.

2.6. System Testing

The testing phase is a crucial step to ensure that the developed YOLO11 model and the built Android application meet the expected standards of accuracy and functionality before being implemented extensively. The testing procedure in this study is conducted through two primary approaches: the evaluation of the object detection algorithm's performance and a comprehensive testing of the application's capabilities.

First, the performance of the YOLO11 model is evaluated using standard computer vision metrics on the testing set. The *mean Average Precision* (mAP) is utilized as the primary parameter for model accuracy across all disease classes, calculated using the following formula [18].

$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad (1)$$

where N represents the number of classes and AP_i is the Average Precision for the i-th class. To measure the ratio of true positive predictions compared to the total predicted positives, *Precision* is calculated as follows :

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

Furthermore, the *F1-score* is used to measure the balance between Precision and Recall, providing a harmonic mean of the two metrics :

$$\text{F1 - score} = 2 \text{ x } \frac{TP}{TP+FP} \quad (3)$$

The *Recall* metric is employed to measure the model's ability to identify all positive class information within the dataset, formulated as:

$$\text{Recall} = \frac{TP}{TP+FN} \quad (4)$$

In these equations, TP (*True Positive*) represents correctly detected disease instances, FP (*False Positive*) represents healthy leaves incorrectly identified as diseased, and FN (*False Negative*) refers to diseased leaves that the model failed to detect. Additionally, the model's speed is tested through the *Inference Time* metric to calculate the duration of processing a single image frame in milliseconds (ms). Second, the system is tested using the *Black-box Testing* method to validate the functionality of the features on the Android application interface. This testing focuses on the system's output in response to given inputs such as real-time camera feeds and gallery uploads without considering the internal code structure of the application [13].

3. RESULT AND DISCUSSION

Training experiments were conducted on three YOLO11 architecture variants, namely Nano (n), Small (s), and Medium (m), using the Google Colaboratory environment with NVIDIA Tesla T4 GPUs. The hyperparameter configuration was standardized with 120 epochs, a batch size of 16, and the AdamW optimizer with an initial learning rate of 0.001.

3.1. Dataset Acquisition and Pre-processing

The initial phase of the results focuses on the preparation and expansion of the dataset utilized for model training. To enhance the robustness of the YOLO11 model against real-world environmental variations, several data augmentation techniques were applied exclusively to the training set. The visual outcomes of these augmentation techniques are illustrated in Figure 4.

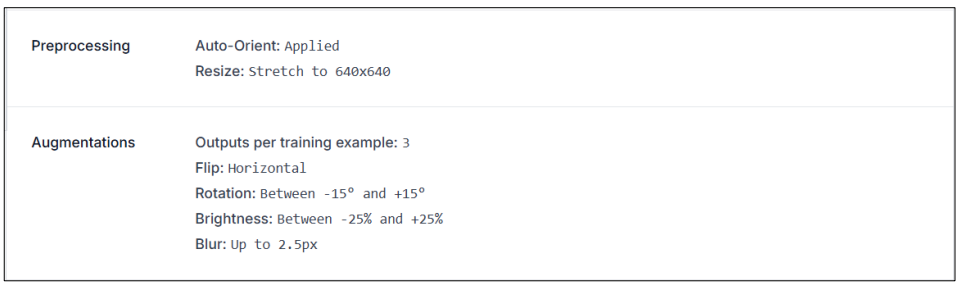

Figure 4. Data Augmentation Results

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

Based on Figure 4, the augmentation process successfully generated diverse visual variations from the original strawberry leaf images. To enhance the robustness of the model against field-specific challenges, a 3x augmentation factor was applied via Roboflow. This process expanded the training set from its original count to 3,543 images, while the validation and testing sets were maintained at 330 and 177 images, respectively. Techniques such as Horizontal Flip, Rotation -15° to +15°, Brightness adjustment -25% to +25%, and Blur up to 2.5px were implemented. These specific parameters were chosen to simulate real-world scenarios such as shaky hands during capture, different sun angles, and varied leaf orientations [19].

3.2. Training Results of YOLO11 Models

The experimental phase involved training three variants of the YOLO11 architecture Nano (n), Small (s), and Medium (m). The training was performed in a Google Colaboratory environment using an NVIDIA Tesla T4 GPU. All models were trained for 120 epochs with a batch size of 16 and utilized the AdamW optimizer with an initial learning rate of 0.001. The quantitative evaluation results at the end of the training process show that all three variants achieved high performance, as summarized in Table 2.

Table 2. Performance Comparison of YOLO11 Models

Model	Precision	Recall	mAP@50-95	mAP@50	Inferences Time (ms)
YOLO11n	**0.888**	**0.841**	0.815	0.917	**4.2 ms**
YOLO11s	0.874	0.829	0.811	0.908	5.5 ms
YOLO11m	0.875	0.836	**0.818**	**0.920**	10.8 ms

Based on Table 2, the YOLO11m model achieved the highest accuracy (mAP@50 of 92%). However, the YOLO11n variant was selected as the optimal model for mobile implementation. This decision was based on its computational efficiency, offering the fastest inference time (4.2 ms) with only a negligible 0.3% difference in accuracy compared to the Medium variant.

3.3. Analysis of the Selected Model

To evaluate the learning stability and convergence of the selected YOLO11n model, the training and validation metrics were continuously monitored throughout the process. The detailed training evaluation curves, including loss metrics and mean Average Precision (mAP) over 120 epochs, are illustrated in Figure 5.

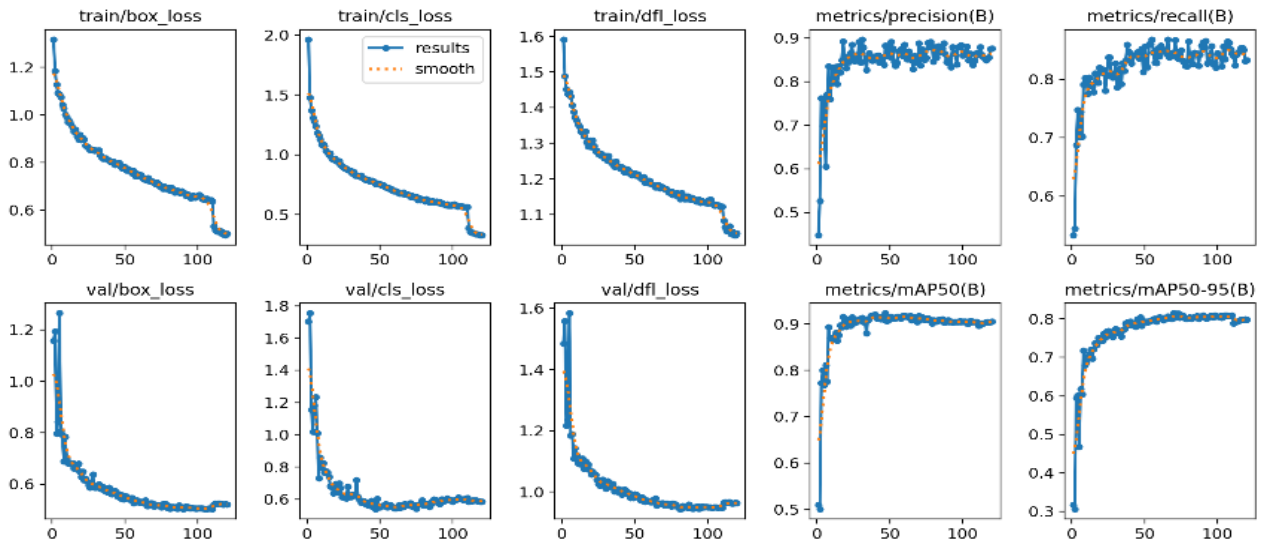

Figure 5. Yolo11n Model Training Evaluation Curve

Based on the learning curves presented in Figure 5, the training process of YOLO11n demonstrated rapid convergence. Specifically, the Box Loss, Classification Loss, and Distribution Focal Loss (DFL) all exhibited a sharp decline within the first 40 epochs. The narrow gap between the training and validation loss curves suggests that the model achieved high generalization without any significant overfitting, validating the effectiveness of the applied augmentation strategy. Furthermore, in terms of per-class analysis, Leaf Spot achieved the highest mAP@50 of 96.6%. This is attributed to the sharp visual contrast between the purple/brown spots and the green leaf surface. Powdery Mildew reached 92.1%, proving that the Nano architecture can effectively extract fine texture features like white fungal powder. The Healthy class had an mAP of 86.4%, which was slightly lower due to natural leaf variations that occasionally mimic early-stage disease symptoms.

To further identify the specific misclassification characteristics across each class, a confusion matrix was generated, as depicted in Figure 6.

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

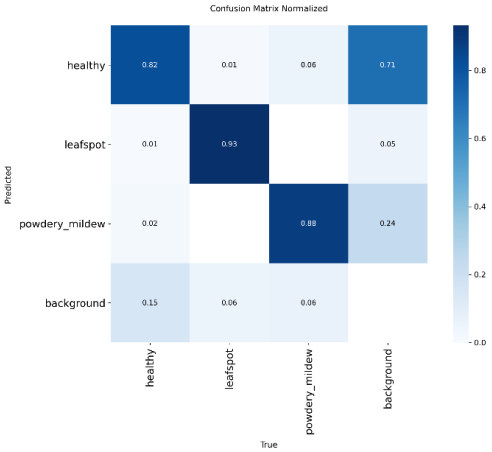

Figure 6. Confusion Matrics

Based on the data mapped in Figure 6, a detailed analysis of the Confusion Matrix revealed the model's discriminative strengths and limitations across different categories :

1. Leaf Spot: Recorded the highest accuracy at 93%. Its distinct purple-to-brown lesions provided high-contrast features that the model easily isolated from the green leaf background.
2. Powdery Mildew: Achieved 88% accuracy. Despite being a fine, white texture that can sometimes be mistaken for light reflections, the model successfully learned to identify the fungal patterns.
3. Healthy: Noted at 82%, with instances (71%) being misclassified as background. This is likely due to healthy leaves blending into the surrounding farm environment or mulch.

To further evaluate the model's reliability in making positive predictions, the relationship between precision and confidence thresholds is illustrated in Figure 7.

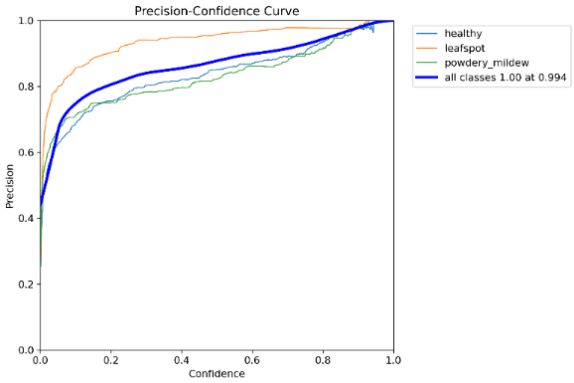

Figure 7. Precision-Confidence Curve

Based on Figure 7, the Precision-Confidence Precision-Confidence Curve illustrates the model's accuracy in its predictions. As the confidence threshold increases, the precision for all classes approach 1.0. This high precision ensures that the application minimizes False Positives, which is critical to preventing farmers from applying unnecessary and costly chemical treatments.

In contrast to precision, the model's sensitivity across different confidence levels is evaluated using the Recall-Confidence Curve, as shown in Figure 8.

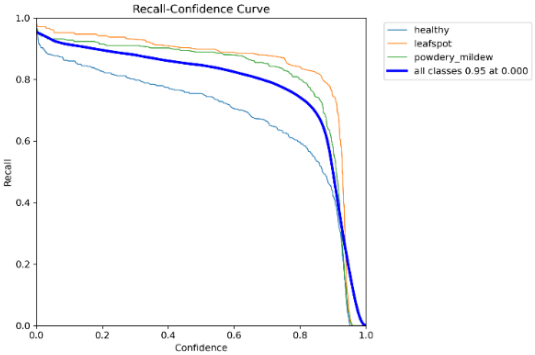

Figure 8. Recall-Confidence Curve

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

Based on Figure 8, the Recall-Confidence Curve measures the model's ability to identify all actual instances of the disease. The YOLO11n model maintained a high recall at lower confidence thresholds, ensuring that subtle symptoms, such as the initial white powder in "Powdery Mildew" are not overlooked (False Negatives). This high sensitivity is vital for early detection and rapid intervention in the field.

To provide a comprehensive overview of the trade-off between precision and sensitivity, the Precision-Recall (PR) curve was generated, as depicted in Figure 9.

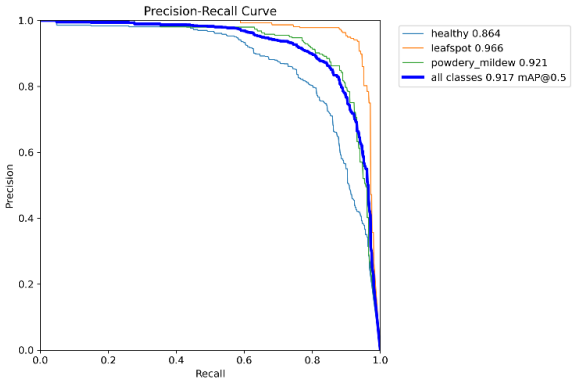

Figure 9. Precision-Recall Curve

As illustrated in Figure 9 Precision-Recall (PR) curve serves as a critical metric for evaluating the performance of the YOLO11n model, particularly in its ability to maintain high precision while maximizing the retrieval of all relevant disease instances. The Area Under Curve (AUC) for the global model reached 0.917, indicating a robust overall detection capability. The sharp curvature toward the top-right corner across all classes signifies that the YOLO11n architecture is highly effective at maintaining high precision even as the recall threshold increases.

Finally, to identify the optimal threshold that balances both metrics, the F1-Confidence Curve was analyzed, as presented in Figure 10.

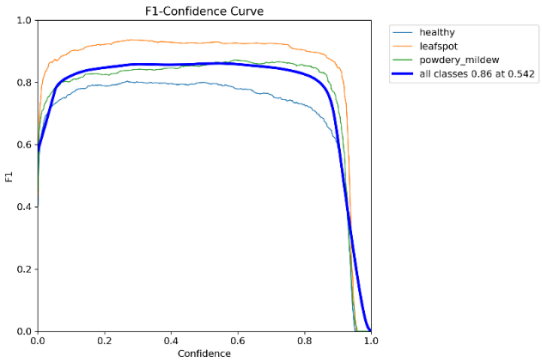

Figure 10. F1-Confidence Curve

Based on Figure 10, to determine the most reliable operational setting for the Android application, the F1-Score was analyzed in relation to the confidence threshold. The F1-Score represents the harmonic mean of precision and recall, providing a single metric that balances the trade-off between missing a disease (False Negative) and raising a false alarm (False Positive). The analysis showed that the F1-Score reached its peak at 0.86 at a confidence threshold of 0.542. This peak indicates the optimal point where the model achieves the best balance between accuracy and sensitivity.

3.4. Mobile Application Implementation and On-Device Performance

The transition from a high-performance cloud GPU environment to a mobile device requires a strategic selection of the model architecture. After evaluating the training results, the YOLO11n variant was selected as the final model for the Android application. This decision was driven by the necessity of balancing high-speed processing with the limited hardware resources of mobile processors. The model was exported using Float32 precision and converted into the TensorFlow Lite (.tflite) format, resulting in a highly optimized file size of only 10.3 MB. The successful integration of this optimized model into the user interface of the Android application is visually presented in Figure 11.

JURIKOM (Jurnal Riset Komputer), Vol. 13 No. 3, Juni 2026
e-ISSN 2715-7393 (Media Online), p-ISSN 2407-389X (Media Cetak)
DOI 10.30865/jurikom.v13i3.9542
Hal 963-972
https://ejurnal.stmik-budidarma.ac.id/index.php/jurikom

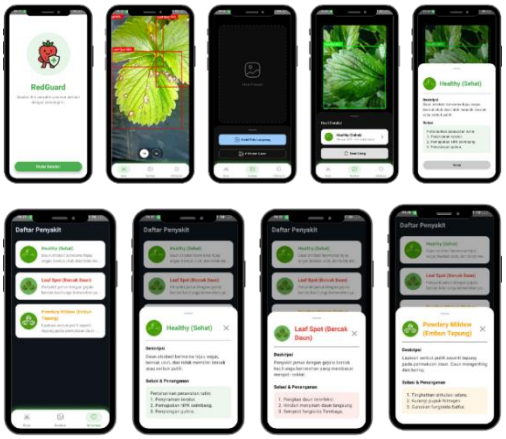

Figure 11. Android Implementation Results

Based on Figure 11, the developed Android application successfully runs the converted YOLO11n model natively, allowing users to perform real-time scanning directly through the device's camera. The interface seamlessly displays the bounding boxes, disease class labels, and confidence scores overlaying the detected strawberry leaves. To further evaluate the practical feasibility of this implementation, on-device testing was conducted to measure the system's operational latency. A comprehensive comparison of the real-time latency and application size across the three model variants when running on a mobile CPU is presented in Table 3.

Table 1. Real-time Latency Analysis on Android

Model	Ukuran model .tflite (mb)	Ukuran Aplikasi (mb)	FPS	Inferences Time (ms)
YOLO11n	**10,3 MB**	**38,4 MB**	**7~10**	**104~125ms**
YOLO11s	37 MB	64,4 MB	3	235~250 ms
YOLO11m	78,5 MB	105 MB	1	585~605ms

As detailed in Table 3, there is a significant performance gap between the three variants when executed on edge hardware. The YOLO11n model demonstrated superior responsiveness, maintaining a stable frame rate between 7 to 10 FPS. This speed is critical for the "live view" feature, ensuring that the bounding boxes track the strawberry leaves smoothly without significant lag. In contrast, the YOLO11m variant experienced a prohibitive latency of over 500 ms, which caused the application to stutter, making it impractical for rapid field scanning.

3.2 Discussion

The experimental results demonstrate that the implementation of the YOLO11n architecture for strawberry disease detection offers a significant advancement in precision agriculture. The achievemet of a 91.7% mAP@50 is particularly noteworthy when compared to previous research in the same domain. For instance, earlier studies utilizing YOLOv8 and K-Fold Cross-Validation on similar strawberry datasets often reported mAP scores ranging from 84% to 88% [20]. The YOLO11n model deployed in this study outperforms that metric (91.7%), demonstrating superior feature extraction despite utilizing a much lighter nano-scale architecture. The superior performance of YOLO11 in this study can be attributed to the integration of the C2PSA (Cross Stage Partial with Spatial Attention) and C3k2 modules, which enhance the model's ability to focus on specific disease textures while effectively ignoring complex background noise such as soil, mulch, and overlapping leaves.

Furthermore, another related study employed the YOLOv7 algorithm for identifying strawberry plant diseases, achieving a slightly higher accuracy rate of 92.5%. While the YOLOv7 architecture shows a marginally higher metric, it is computationally heavy and contains a massive number of parameters[11]. This makes it primarily suited for high-resource server-side processing and highly inefficient for standalone mobile applications. In contrast, the YOLO11n model proposed in this research maintains highly competitive diagnostic precision (91.7%) while dramatically reducing the computational burden. This efficiency allows the model to achieve a real-time inference speed of 104–125 ms (equivalent to 7–10 FPS) directly on a mid-range Android smartphone without any cloud dependency. This offline capability directly addresses the internet connectivity constraints often faced in rural agricultural areas like Serang Village.

The analysis of class-specific performance reveals that the model is highly effective at identifying Leaf Spot (96.6% mAP) due to the high-contrast necrotic lesions. However, the most significant contribution of this research is the high accuracy achieved for Powdery Mildew (92.1% mAP). In many practical scenarios, powdery mildew is difficult to diagnose manually in its early stages as it appears as a faint white powder that can be confused with light reflections on the leaf's waxy surface. The YOLO11n model's ability to extract these fine textural features ensures that farmers can intervene before the fungus spreads to the fruit.

4. CONCLUSION

This research successfully developed a mobile-based diagnostic tool for strawberry leaf diseases by implementing the YOLO11 architecture, with YOLO11n (Nano) proving to be the most effective for real-time agricultural use on Android devices. The model achieved a mean Average Precision (mAP@50) of 91.7% in identifying Healthy, Leaf Spot, and Powdery Mildew classes, with particularly strong performance on Leaf Spot detection at 96.6% precision and Powdery Mildew at 92.1% accuracy. Using the F1-Confidence curve, a confidence threshold of 0.542 was determined as the optimal balance between precision and recall for stable live field scanning. In addition, the shift from cloud-based GPU training to mobile-based TFLite implementation showed that YOLO11n offers superior efficiency, with a compact model size of 10.3 MB and an average inference time of 115 ms, enabling smooth performance at 7–10 FPS and outperforming heavier models such as YOLO11m. This application bridges advanced deep learning with practical farming needs, allowing farmers in places like Serang Village to conduct immediate, on-site disease diagnosis without internet access, although the study is still limited to three classes and natural daylight conditions.

REFERENCES

[1] B. P. S. Indonesia, "Produksi Tanaman Buah-buahan - Tabel Statistik." Accessed: Jul. 16, 2025. [Online]. Available: https://www.bps.go.id/id/statistics-table/2/NjIjMg==/produksi-tanaman-buah-buahan.html

[2] A. Aldrighetti and I. Pertot, "Epidemiology and control of strawberry powdery mildew: a review," *Phytopathol. Mediterr.*, vol. 62, no. 3, pp. 427–453, 2023, doi: 10.36253/PHYTO-14576.

[3] Y. Chen *et al.*, "CES-YOLOv8: Strawberry Maturity Detection Based on the Improved YOLOv8," *Agronomy*, vol. 14, no. 7, 2024, doi: 10.3390/agronomy14071353.

[4] J. Yao, Y. Li, Z. Xia, P. Nie, X. Li, and Z. Li, "WTAD-YOLO: A lightweight tomato leaf disease detection model based on YOLO11," *Smart Agric. Technol.*, vol. 12, no. July, p. 101349, 2025, doi: 10.1016/j.atech.2025.101349.

[5] A. Herbert, P. Sitohang, T. I. Hermanto, and C. D. Lestari, "Tumbuhan Stroberi Menggunakan Metode Convolutional Neural Network Arsitektur," *JITET (Jurnal Inform. dan Tek. Elektro Ter.*, vol. 12, no. 3, 2024.

[6] J. Lee, "YOLO with adaptive frame control for real - time object detection applications," *Multimed. Tools Appl.*, pp. 36375–36396, 2022, doi: 10.1007/s11042-021-11480-0.

[7] M. L. Ali and Z. Zhang, "The YOLO Framework: A Comprehensive Review of Evolution, Applications, and Benchmarks in Object Detection," *Computers*, vol. 13, no. 12, 2024, doi: 10.3390/computers13120336.

[8] M. Yaseen, "What is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector," vol. 8, pp. 1–10, 2024, [Online]. Available: http://arxiv.org/abs/2408.15857

[9] Praveen Borra, "A Survey of Google Cloud Platform (GCP): Features, Services, and Applications," *Int. J. Adv. Res. Sci. Commun. Technol.*, pp. 191–199, 2024, doi: 10.48175/ijarsct-18922.

[10] M. Reda, R. Suwwan, S. Alkafri, Y. Rashed, and T. Shanableh, "AgroAId: A Mobile App System for Visual Classification of Plant Species and Diseases Using Deep Learning and TensorFlow Lite," *Informatics*, vol. 9, no. 3, 2022, doi: 10.3390/informatics9030055.

[11] S. Pertiwi, D. H. Wibowo, and S. Widodo, "Deep Learning Model for Identification of Diseases on Strawberry (Fragaria sp.) Plants," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 13, no. 4, pp. 1342–1348, 2023, doi: 10.18517/ijaseit.13.4.19018.

[12] H. Hao, J. Xi, J. Dai, G. Wang, D. Liu, and L. Zhu, "DHN-YOLO: A Joint Detection Algorithm for Strawberries at Different Maturity Stages and Key Harvesting Points," *Plants*, vol. 14, no. 22, pp. 1–23, 2025, doi: 10.3390/plants14223439.

[13] I. Wahyudi, Fahrullah, F. Alameka, and Haerullah, "Analisis Blackbox Testing Dan User Acceptance Testing Terhadap Sistem Informasi Solusimedsosku," *J. Teknosains Kodepena* |, vol. 04, no. 01, pp. 1–9, 2023.

[14] M. Ihsan Muttaqin, A. Stefanie, and L. Nurpulaela, "Implementasi Metode Deep Learning Untuk Menentukan Tingkat Kematangan Buah Pepaya California," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 7, no. 3, pp. 1878–1884, 2023, doi: 10.36040/jati.v7i3.6948.

[15] D. Fitrianah, "Enhancing low-light pedestrian detection : convolutional neural network and YOLOv8 integration with automated dataset," vol. 14, no. 3, pp. 1969–1980, 2025, doi: 10.11591/eei.v14i3.8903.

[16] A. Dwivedi, M. Henige, K. Anklam, and D. Döpfer, "Real-time digital dermatitis detection in dairy cows on Android and iOS apps using computer vision techniques," *Transl. Anim. Sci.*, vol. 9, no. February, 2025, doi: 10.1093/tas/txae168.

[17] F. Dzulqarnain and T. Tukino, "Rancang Bangun Aplikasi Belajar Arab Untuk Android Menggunakan Jetpack Compose Dan Kotlin," *Comput. Based Inf. Syst. J.*, vol. 11, no. 1, pp. 25–35, 2023, doi: 10.33884/cbis.v11i1.6666.

[18] M. Mao and M. Hong, "YOLO Object Detection for Real-Time Fabric Defect Inspection in the Textile Industry: A Review of YOLOv1 to YOLOv11," *Sensors*, vol. 25, no. 7, pp. 1–43, 2025, doi: 10.3390/s25072270.

[19] C. Wen *et al.*, "Sugarcane stem node detection with algorithm based on improved YOLO11 channel pruning with small target enhancement," *PLoS One*, vol. 20, no. 9 September, 2025, doi: 10.1371/journal.pone.0332870.

[20] I. M. D. Pranata, I. W. A. S. Darma, I. M. S. Sandhiyasa, and I. K. A. G. Wiguna, "Strawberry Disease Detection Based on YOLOv8 and K-Fold Cross-Validation," *J. Ilm. Merpati (Menara Penelit. Akad. Teknol. Informasi)*, vol. 11, no. 3, p. 199, 2023, doi: 10.24843/jim.2023.v11.i03.p06.