ATCSJ
Open Access
Applied Technology and Computing Science Journal
p-ISSN 2621-4458 | e-ISSN 2621-4474

Available online to journal2.unusa.ac.id
UNUSA
S4- Accredited – SK No.79/M/KPT/2023
Journal Page is available to https://journal2.unusa.ac.id/index.php/ATCSJ

Development of a Rice Leaf Disease Detection Application Using Python-Based Computer Vision and YOLO

Fajar Maulana[1], Yomei Hendra[2,*], Guswita Helmi[3], Radhiatul Husna[4], Amma Liesvarastranta Haz[5], Evianita Dewi Fajrianti[6]

[1,2,3]*Informatics Adzkia University, Indonesia*
Taratak Paneh, Kuranji, Padang City
[1]fajar@adzkia.ac.id , [2*]yomeihendra@adzkia.ac.id , [3]guswitahelmi0211@gmail.com

[4]*Andalas University, Indonesia*
Limau Manis, Pauh, Padang City
[4]radhiatulhusna@sci.unand.ac.id

[5,6]*Department of Informatic and Computer Engineering, Politeknik Elektronika Negeri Surabaya, Indonesia*
[5]liesvarastranta@pens.ac.id , [6]evianita08@pens.ac.id

Article history:

Received 2 December 2025
Revised 15 December 2025
Accepted 31 December 2025
Available online 23 February 2026

Keywords:

Computer Vision,
YOLO,
Disease Detection,
Rice Leaf,
Python.

Abstract

Rice is one of Indonesia's primary agricultural commodities and is highly vulnerable to various leaf diseases, including blast, blight, brown spot, and tungro, which can significantly reduce crop productivity. To address this issue, an automated and accurate detection system is needed to assist farmers in identifying rice leaf diseases at an early stage. This study aims to develop a rice leaf disease detection application using computer vision technology based on Python and the YOLO (You Only Look Once) algorithm. The research methodology consisted of several stages: problem identification, data acquisition, data exploration, model development, evaluation, and deployment. The dataset was obtained from Roboflow and comprised five classes: blast, blight, brown spot, healthy, and tungro. The YOLO model was trained using Google Colab with optimized parameters to enhance detection performance. Experimental results demonstrate that the proposed model achieved an accuracy of 95% and a mean Average Precision (mAP) of 95%, indicating strong performance in detecting and classifying rice leaf diseases. The system was implemented as a web-based application using Flask and Bootstrap, allowing users to upload images of rice leaves and obtain real-time detection results. This application enables farmers to identify plant diseases quickly and accurately, facilitating timely and effective intervention to minimize crop losses.

[1*]Corresponding author

ISSN 2621-4474 (online) 2621-4458 (print) © 20XX The Authors. Published by UNUSA
This is an open access article and licensed under a Creative Commons Attribution-Share Alike 4.0 International License
DOI: https://doi.org/10.33086/atcsj.v8i2.8486

I. INTRODUCTION

Agriculture is a key sector that strongly influences food security and the economy in Indonesia. One of the most important commodities in this sector is rice (Oryza sativa), which serves as the primary source of carbohydrates for most of the population. However, rice production often faces constraints, one of which is leaf disease outbreaks that are not detected early. Farmers' limited knowledge of early disease symptoms leads to delayed treatment and results in a decline in both the quality and quantity of harvested yields.

Leaf diseases affecting rice plants—such as blast, blight, brown spot, and tungro—pose major challenges to improving agricultural output. These diseases cause discoloration and the appearance of lesions on leaves, which can disrupt photosynthesis and impair plant growth. Therefore, a disease detection system that is fast, accurate, and easily accessible to farmers is needed.

Along with technological advances, the application of computer vision and artificial intelligence has been widely adopted in agriculture to automatically monitor plant health. One of the most popular methods with high accuracy for object detection is You Only Look Once (YOLO). YOLO is a deep learning algorithm capable of recognizing and classifying objects in images in real time. The combination of YOLO and the Python programming language provides an efficient and practical solution for detecting plant diseases from digital images of leaves.

Several previous studies have demonstrated the effectiveness of using YOLO for plant disease detection. For example, Putra Pranjaya et al. (2024) used YOLOv5 for rice leaf disease classification and achieved an mAP of 95%, while Deden Miftah Fauzi et al. (2024) reported a precision of 92.5% and a recall of 90.8% in detecting rice disease types. These findings highlight the strong potential of this technology for broader implementation.

Based on this background, this study aims to develop a web-based rice leaf disease detection application using computer vision, Python, and YOLO. Through this application, farmers are expected to be able to identify diseases independently, quickly, and accurately, thereby improving crop yields and overall food security.

II. LITERATURE REVIEW

A. Artificial Intelligence

Artificial intelligence (AI) was first introduced in 1950 (Kaul et al., 2020). AI refers to the capability to enhance a system's intelligence through appropriate configurations based on scientific standards (Esteva et al., 2021). AI is a branch of computer science that focuses on developing systems and machines capable of performing tasks that typically require human intelligence. AI uses data and algorithms to model capabilities such as language understanding, face recognition, problem solving, decision making, and learning from data.

B. Computer Vision

Computer vision is a component of artificial intelligence that imitates the human neural system, particularly the brain and the eyes. Machine vision tasks span multiple levels: lower-level tasks include detecting edges in images, while higher-level tasks involve understanding an entire scene to detect objects in images or videos in real time (Anugrah Pramesta et al., 2022).

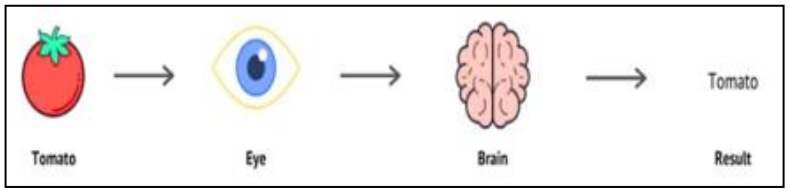

Figure 1. Analogy of the Human Visual System

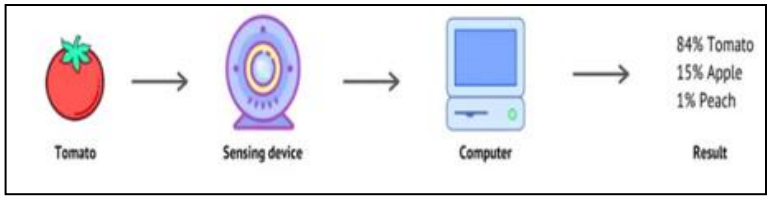

Figure 2. Computer Vision Analogy

Computer vision enables decision-making about real-world physical objects captured through devices or sensors, and it allows computers to recognize images in a human-like manner (Susim & Darujati, 2021).

C. Image Processing

Image processing is a technology used to address image-processing problems (Susim & Darujati, 2021). It refers to methods for processing images to enhance image quality so that they can be more easily interpreted by humans or computer systems, either as photographs or moving images (Rilo Pambudi et al., 2020). In image processing, images are represented as bits in an array containing real and complex values.

D. YOLO

You Only Look Once (YOLO) is an object detection method that is considered effective within deep learning approaches. This is because YOLO extracts features from an image and computes bounding boxes with class probabilities, then produces the object class results along with location information in the image (Ardiansyah & Hasan, 2023).

E. Python

Python is a high-level programming language created by Guido van Rossum and released in 1991. Python has become increasingly popular in recent years. It is a versatile language that can be used for various purposes, including machine learning and deep learning (Alfarizi et al., 2023). Python is an object-oriented programming language that can be used to develop software and run across multiple operating systems. It is currently widely used in data science and analytics due to its strong support for repositories. These repositories provide functions for data analysis and machine learning, data preprocessing tools, and data visualization (Koretsky, 2023).

F. Flask

Flask is lightweight and categorized as a microframework because it does not require specific tools or libraries to be used (Ngantung & Pakereng, 2021). Flask is a well-known and easy-to-use framework for developing web applications using the Python programming language. Due to its high flexibility, Flask has become a popular choice among developers for building web applications.

G. Roboflow

Roboflow is a web-based platform for managing datasets and serves as a computer vision development framework that offers improved processing and model training techniques for data collection. With Roboflow, users can share datasets along with their processing workflows, use bounding boxes to label (annotate) objects to be detected, and perform preprocessing such as grayscale conversion and augmentation (Hayati et al., 2023).

III. METHODS

This study adopts a software engineering approach based on the Artificial Intelligence (AI) Project Cycle, which consists of six main stages: problem scoping, data acquisition, data exploration, modelling,

evaluation, and deployment. All stages were implemented to produce an effective rice leaf disease detection application that can be used by users in real time.

A. Problem Scoping

The initial stage is problem scoping, which aims to understand and define the problems faced by farmers in early detection of rice leaf diseases. The objective of this stage is to design a computer vision–based solution that enables users to automatically identify diseases from leaf images.

B. Data Acquisition

The dataset used in this study was obtained from the Roboflow platform. It consists of five classes: blast, blight, brown spot, healthy, and tungro. The images were annotated using bounding boxes to mark the target objects in each image. The dataset was then split into three subsets: training, validation, and testing.

C. Data Exploration

At this stage, the collected data were analyzed to understand class distribution, image quality, and the augmentation techniques used to increase data size and diversity. Data augmentation included rotation, scaling, and brightness adjustments to make the model more robust to real-world environmental variations.

D. Modelling

The object detection model used in this study is YOLOv5 (You Only Look Once version 5). The model was trained using Google Colab with a GPU configuration and training parameters such as batch size, learning rate, and epochs, which were tuned to improve accuracy. The training process used the previously annotated data and produced a detection model capable of recognizing all five disease classes.

E. Evaluation

Model evaluation was conducted using metrics such as accuracy, precision, recall, F1-score, and mean Average Precision (mAP).

a. Accuracy

Accuracy is a metric used to measure the overall ability of a system to correctly classify instances. To compute accuracy, the following equation can be used:

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

b. Precision

Precision indicates how accurately the YOLO model classifies objects as positive. A higher precision value means fewer objects are incorrectly detected as positive (false positives). Precision can be calculated using the following equation:

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

c. Recall

Recall indicates how well the YOLO model detects objects that are truly positive. A higher recall value means fewer objects are missed (false negatives). Recall can be calculated using the following equation:

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

d. F1-Score

The F1-score is useful for evaluating overall model performance, particularly when the dataset contains imbalanced classes. The F1-score ranges from 0 to 1, where a value closer to 1 indicates better performance with optimal precision and recall. The F1-score can be calculated using the following equation:

$$FI = \frac{2*Precision*Recall}{Precision+Recall} \quad (4)$$

e. Mean Average Precision (mAP)

mAP computes the average precision for each object class in the dataset and then takes the mean across classes. Precision is calculated at different threshold levels to produce a precision–recall curve. To compute mAP, the area under the curve (AUC) of the precision–recall curve is calculated. The mAP value can be obtained using the following equation:

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k \quad (5)$$

F. Deployment

The developed application, named Agricare, was built using Flask as the backend and Bootstrap as the frontend. The application runs through a web interface, allowing users to upload images, use a camera, or perform detection via video. The detection results are displayed directly on the screen in real time.

IV. RESULTS AND DISCUSSIONS

During the implementation stage, the rice leaf disease application utilizing the YOLO model was integrated into a website-based application. The trained YOLO model for detecting various rice leaf diseases was deployed on the application backend. To connect the model with the website, the Flask framework was used to handle user requests and execute the model. The website interface was designed using HTML, CSS, and Bootstrap to ensure usability and ease of use. Users can upload rice leaf images through the Check page, which are then processed by the model to detect diseases. The detection results are presented to users in clear and easy-to-understand information.

A. Modelling Results

After the model was trained using different numbers of epochs, the runs/train/ directory contained several exp folders, each storing files that can be used to review the training outcomes. To compare recall, precision, and F1-score across different epoch settings, this information is presented in the following table.

Table 1. Comparison of Results for Each Epoch

Epoch	Precision	Recall	F1-score
10	1.00	0.93	0.81
20	1.00	0.96	0.87
50	1.00	0.97	0.91
75	1.00	0.98	0.92
100	1.00	0.98	0.93

a. Confusion Matrix

The first aspect to consider when assessing whether a model performs well is the confusion matrix. For the previously trained model, based on the confusion matrix shown below, it can be concluded that the model is performing well.

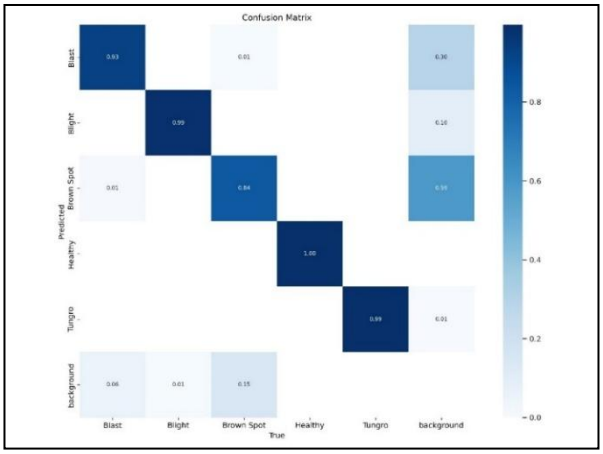

Figure 3. Confusion Matrix

b. Recall

Recall is an important metric for evaluating model performance. Based on the recall curve below, each class shows a recall value that reaches 0 at a confidence level of 0.98, indicating that the model has achieved the desired score and demonstrates good performance.

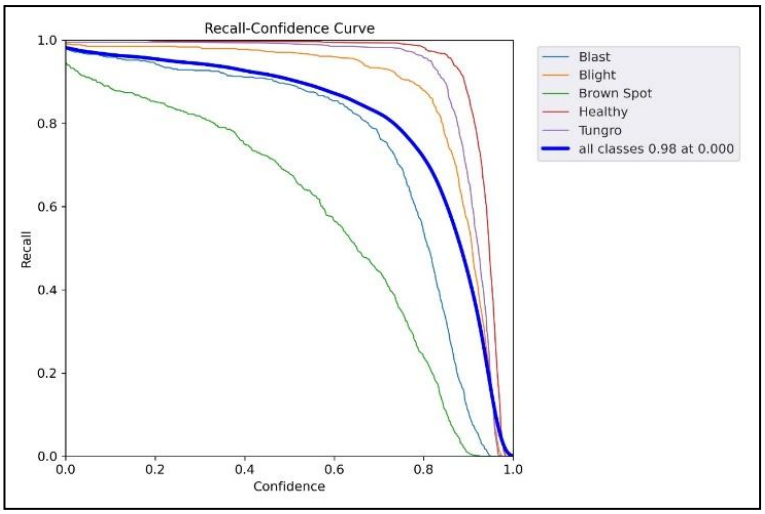

Figure 4. Recall

c. Precision

Precision reflects model quality, where an upward-trending curve indicates improving performance. Based on the curve below, the precision curve increases, suggesting that the model quality is improving. The precision value reaches 1 at a confidence level of 0.970, indicating that the model identifies classes very accurately at that confidence threshold.

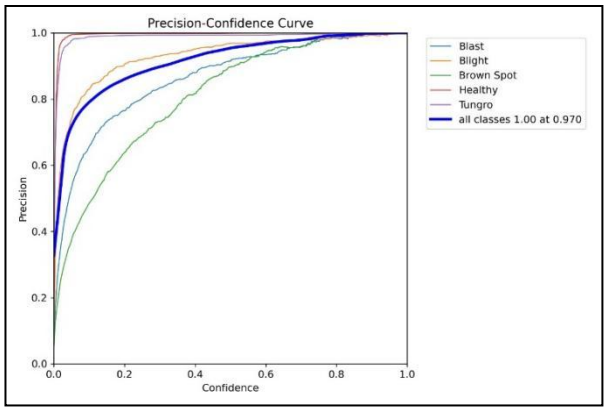

Figure 5. Precision

d. Precision–Recall Curve

After analyzing recall and precision, these two metrics were compared to obtain a score. The results were 0.942 for the blast class, 0.982 for blight, 0.839 for brown spot, 0.995 for healthy, and 0.995 for tungro. In the context of the precision–recall relationship, this study prioritizes high precision with lower recall, as accurate and precise object detection is required.

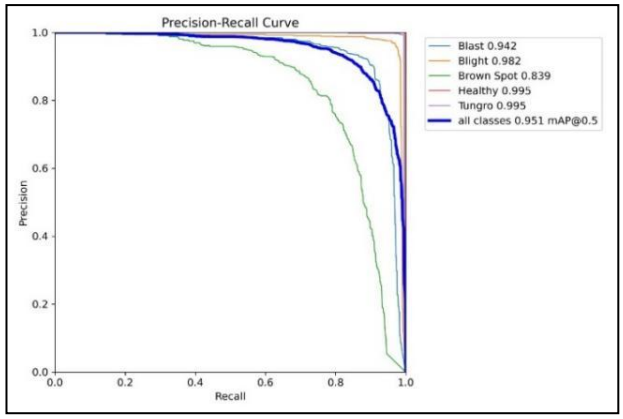

Figure 6. Precision Recall Curve

e. F1-Score

The F1-score is an important metric for assessing whether the model meets the desired performance. Based on the F1 curve below, the F1-score reaches 0.93 (93%) at a confidence threshold of 0.442. With a high F1-score, it can be concluded that the model has good quality and is able to deliver optimal results.

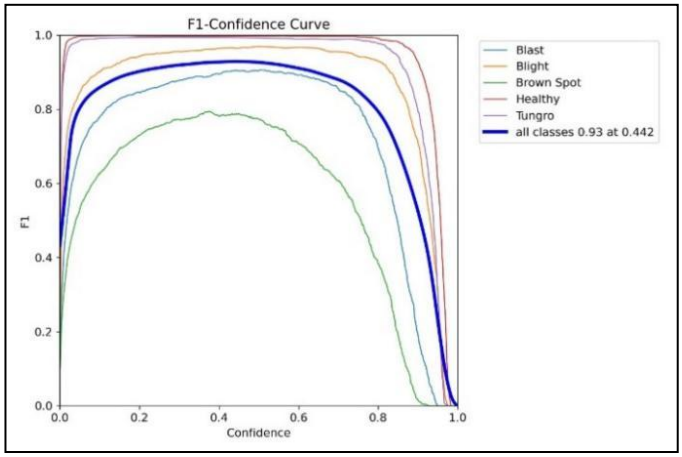

Figure 7. F-1 Score

f. Loss Graph

Loss is an important indicator for evaluating model performance. The lower the loss value, the better the model performs in object classification. The loss curves show a decreasing trend for both training and validation, indicating that the model is learning and improving over time.

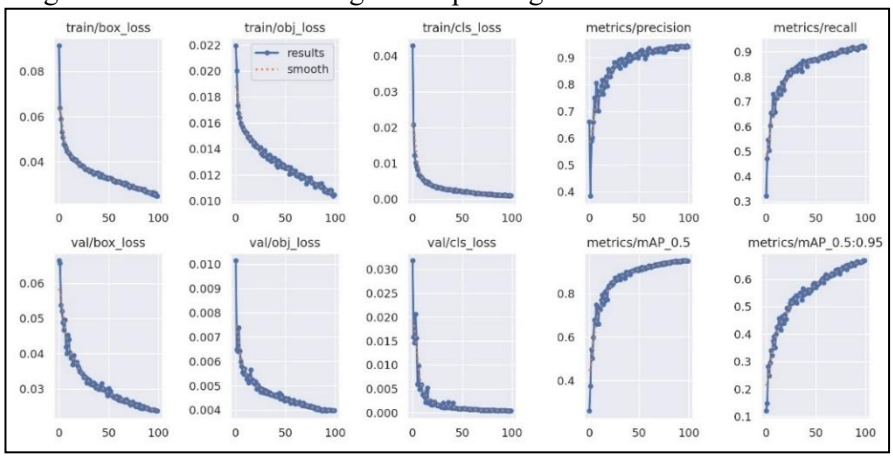

Figure 8. Loss Graph

B. Accuracy and mAP Calculation Results

At this stage, the confusion matrix is broken down into two main parts: the confusion matrix table and the table identifying TP, FP, FN, and TN.

Table 2. Confusion Matrix

Predicted/True	Blast	Blight	Brown Spot	Healthy	Tungro
Blast	0.93	0.01	0.00	0.00	0.06
Blight	0.00	0.99	0.01	0.00	0.00
Brown Spot	0.01	0.00	0.84	0.00	0.15
Healthy	0.00	0.00	0.00	1.00	0.00
Tungro	0.00	0.00	0.00	0.01	0.99

Table 3. Per-Class Calculations

Kelas	Precision	Recall	F1-score
Blast	0.93	0.989	0.959
Blight	0.99	0.99	0.99
Brown spot	0.84	0.988	0.909
Healthy	1.00	1.00	1.00
Tungro	0.99	1.00	0.995

After precision, recall, and F1-score were determined, the next step was to calculate the overall accuracy for each class. The mAP calculation was performed by averaging the prediction accuracy across classes. Based on the calculations, the obtained accuracy was 0.95, indicating the model's correctness in disease detection. The mean Average Precision (mAP) also reached 0.95, reflecting the overall model performance. These results demonstrate the effectiveness of the model in producing accurate predictions. Overall, the model shows very strong performance in detecting diseases on rice leaves.

C. System Implementation

During system implementation, users are introduced to how to use Agricare, starting from accessing the home page, uploading rice leaf images, and viewing the disease detection results. The system allows users to capture rice leaf images in real time or upload images through the Check feature. The YOLO model then

detects and classifies the disease types present. The detection results are displayed with bounding boxes and the corresponding confidence score. Administrators can manage the training dataset to improve model accuracy, while users only need to upload an image to obtain automated analysis results.

The Check page is the main page in Agricare used to detect diseases on rice leaves. Users can upload an image or use the camera in real time for analysis. The YOLO model processes the image and displays the detection results, including the disease name, bounding boxes, and confidence score.

Figure 9. Successful Camera Capture

The camera is used to analyze static images. In this application, the camera processes images directly to detect the condition of rice leaves. The system operates by analyzing the captured image without requiring manual uploads, thereby improving user efficiency. With this feature, disease identification becomes faster and more practical.

Figure 10. Rice Leaf Captured in Real Time

Rice leaves can be captured directly using the camera. The system then analyzes the image to identify the disease type. This process is performed in real time using the YOLO model to ensure accurate detection.

Figure 11. Successful Video Detection

The video feature captures and analyzes rice leaf images in real time using the YOLO model. It provides two buttons: Stop and Save. The system detects diseases by displaying bounding boxes, the disease name, and the detection confidence, without requiring manual image uploads. This feature makes the process faster and more practical.

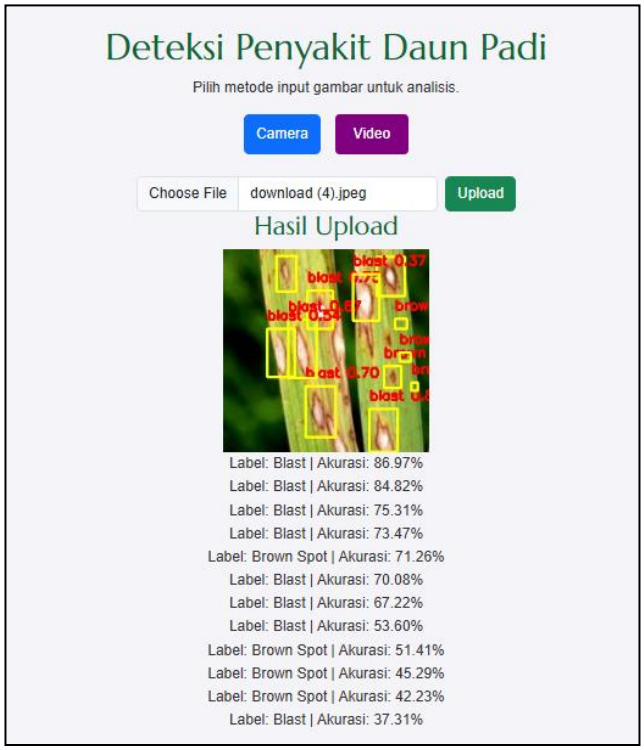

Figure 12. Successful Image Upload

The upload feature allows users to submit rice leaf images for analysis by the YOLO model. With this feature, users can utilize existing images without having to take photos directly using the camera. The analysis is performed automatically to provide fast and efficient results.

V. CONCLUSIONS AND RECOMMENDATIONS

Based on the findings of this study, it can be concluded that the development of a rice leaf disease detection application using Python-based computer vision and the YOLO algorithm has been successfully implemented. The development process followed a systematic workflow—comprising problem scoping, data acquisition, data exploration, modeling, evaluation, and deployment—to ensure optimal model performance. The Agricare application was developed as a web-based platform with a Flask backend and a Bootstrap-based user interface, allowing users to upload images or utilize a real-time camera feature for disease detection. The dataset, obtained from Roboflow, was processed through labeling and trained using Google Colab. Evaluation results indicate that the model achieved an accuracy of 95% and a mean Average Precision (mAP) of 95%, demonstrating strong effectiveness in detecting and classifying rice leaf diseases. Through bounding-box detection and class probability outputs, Agricare enables farmers to identify diseases accurately and efficiently without the need for specialized equipment, thereby supporting improved agricultural productivity and reducing the risk of crop losses due to misdiagnosis.

For future research, several improvements are recommended to enhance the robustness and practical applicability of the system. These include expanding the dataset to incorporate more diverse field conditions, such as variations in lighting, background complexity, and different stages of leaf growth, as well as including additional disease categories to broaden detection coverage. Furthermore, conducting field trials in real farming environments is essential to evaluate the system's reliability, usability, and performance under practical conditions. Such enhancements would strengthen the system's generalizability and increase its potential for large-scale implementation in the agricultural sector.

VI. REFERENCES

[1] Agustiani, S., Arifin, Y. T., Junaidi, A., Wildah, S. K., & Mustopa, A. (2022). Klasifikasi Penyakit Daun padi Menggunakan random forest dan color histogram. Jurnal Komputasi, 10(1), 65-74. https://doi.org/10.23960/komputasi.v10i1.2961.

[2] Aini, Q., Lutfiani, N., Kusumah, H., & Zahran, M. S. (2021). Deteksi dan Pengenalan Objek Dengan Model Machine Learning: Model Yolo. CESS (Journal of Computer Engineering, System and Science), 6(2), 192. https://doi.org/10.24114/cess.v6i2.25840.

[3] Anwar, M., Kristian, Y., & Setyati, E. (2023). Klasifikasi penyakit tanaman cabai rawit dilengkapi dengan segmentasi citra daun dan buah menggunakan YOLO v7. INTECOMS: Journal of Information Technology and Computer Science, 6(1), 540-548. https://doi.org/10.31539/intecoms.v6i1.6071.

[4] Arifin, N., Insani, C. N., & Rasyid, M. R. (2023). Klasifikasi Tingkat Kematangan Buah Tomat menggunakan Computer vision untuk Smart Agriculture. Jurnal SAINTIKOM (Jurnal Sains Manajemen Informatika dan Komputer), 22(2), 509-516. https://doi.org/10.53513/jis.v22i2.8387.

[5] Ayuni, D. P., Irsyad, M., Yanto, F., & Sanjaya, S. (2023). Augmentasi Data Pada Implementasi Convolutional Neural Network Arsitektur Efficientnet-B3 Untuk Klasifikasi Penyakit Daun Padi. ZONAsi: Jurnal Sistem Informasi, 5(2), 239-249. https://doi.org/10.31849/zn.v5i2.13874.

[6] BPS-Statistics Indonesia. (2024). Executive Summary: Paddy Harvested Area and Production in Indonesia 2023 (Final Figures). 1–54. https://www.bps.go.id/id/publication/2024/05/06/69834d72f7ef1c32eee5c4b6/ringkasan-eksekutif-luas-panen-dan-produksi-padi-di-indonesia-2023--angka-tetap-.html

[7] Fauzi, M. D. M., Al Mudzakir, T., Sukmawati, C. E., & Indra, J. (2024). Deteksi Jenis Penyakit Pada Tanaman Padi Menggunakan Yolo V5. KLIK: Kajian Ilmiah Informatika dan Komputer, 5(1), 39-48. https://doi.org/10.30865/klik.v5i1.2009.

[8] Hartati, S. (2021). Kecerdasan Buatan Berbasis Pengetahuan. Ugm Press.

[9] Hasan, N. F. (2023). Deteksi dan Klasifikasi Penyakit Pada Daun Kopi Menggunakan Yolov7. Jurnal Sisfokom (Sistem Informasi Dan Komputer), 12(1), 30-35. https://doi.org/10.32736/sisfokom.v12i1.1545.

[10] Hawari, F. H., Fadillah, F., Alviandi, M. R., & Arifin, T. (2022). Klasifikasi Penyakit Tanaman Padi Menggunakan Algoritma Cnn (Convolutional Neural Network). Jurnal Responsif: Riset Sains Dan Informatika, 4(2), 184-189. https://doi.org/10.51977/jti.v4i2.856.

[11] Ibrahim, M., & Latifa, U. (2023). PENERAPAN ALGORITMA YOLOV8 DALAM DETEKSI WAKTU PANEN TANAMAN PAKCOY BERBASIS WEBSITE. JATI (Jurnal Mahasiswa Teknik Informatika), 7(4), 2489-2495. https://doi.org/10.36040/jati.v7i4.7154.

[12] Julianto, B., Farida, I. N., & Dara, M. A. D. W. (2023). Implementasi Metode CNN Pada Aplikasi Android Untuk Deteksi Penyakit Pada Daun Padi. Prosiding SEMNAS INOTEK (Seminar Nasional Inovasi Teknologi), 7(2), 963–970. https://doi.org/10.29407/inotek.v7i2.3522.
[13] Kaul, V., Enslin, S., & Gross, S. A. (2020). History of artificial intelligence in medicine. Gastrointestinal endoscopy, 92(4), 807-812. https://doi.org/10.1016/j.gie.2020.06.040.
[14] Khaira, U., Weni, I., & Wilia, W. (2024). Rancang Bangun Aplikasi Deteksi Penyakit Tanaman Jagung Melalui Citra Daun Berbasis Android Menggunakan Algoritma Convolutional Neural Network. Jurnal Pepadun, 5(1), 1-11. https://doi.org/10.23960/pepadun.v5i1.210.
[15] Khoiruddin, M., Junaidi, A., & Saputra, W. A. (2022). Klasifikasi Penyakit Daun Padi Menggunakan Convolutional Neural Network. Journal of Dinda: Data Science, Information Technology, and Data Analytics, 2(1), 37-45. https://doi.org/10.20895/dinda.v2i1.341.
[16] Kurniawan, D. (2022). Pengenalan machine learning dengan python. Elex Media Komputindo.
[17] NUR HIKMAH, N. U. R., & Ali Khumaidi, A. K. RANCANG BANGUN PROTOTIPE PENGUSIR HAMA BURUNG MENGGUNAKAN SENSOR GERAK RCWL MICROWAVE BERBASIS INTERNET OF THINGS. Jurnal Simetris. https://doi.org/10.24176/simet.v11i2.5071.
[18] Pramesta, A. Perancangan Aplikasi Identifikasi Penyakit Kerdil dan Leaf Blight Pada Tanaman Lada Berdasarkan Citra Daun Berbasis Website (Bachelor's thesis, Fakultas Sains dan Teknologi UIN Syarif HIdayatullah Jakarta). https://doi.org/10.69916/jkbti.v1i3.10.
[19] Pranjaya, A. P., Rizki, F., Kurniawan, R., & Daulay, N. K. (2024). Klasifikasi Penyakit Pada Daun Tanaman Padi Berbasis YoloV5 (You Only Look Once). KLIK: Kajian Ilmiah Informatika dan Komputer, 4(6), 3127-3136. https://doi.org/10.30865/klik.v4i6.1916.
[20] Rijal, M., Yani, A. M., & Rahman, A. (2024). Deteksi Citra Daun untuk Klasifikasi Penyakit Padi menggunakan Pendekatan Deep Learning dengan Model CNN. Jurnal Teknologi Terpadu, 10(1), 56-62. https://doi.org/10.54914/jtt.v10i1.1224.
[21] Saputra, R. A., Wasiyanti, S., Supriyatna, A., & Saefudin, D. F. (2021). Penerapan Algoritma Convolutional Neural Network Dan Arsitektur MobileNet Pada Aplikasi Deteksi Penyakit Daun Padi. Jurnal Swabumi, 9(2), 184-188. https://doi.org/10.31294/swabumi.v9i2.11678.
[22] Sayuthi, M., Hanan, A., Muklis, M., & Satriyo, P. (2020). Distribusi hama tanaman padi (Oryza sativa L.) pada fase vegetatif dan generatif di Provinsi Aceh. Jurnal Agroecotania: Publikasi Nasional Ilmu Budidaya Pertanian, 3(1), 1-10. https://doi.org/10.22437/agroecotania.v3i1.11286.
[23] Virgiawan, I., Maulana, F., Putra, M. A., Kurnia, D. D., & Sinduningrum, E. (2024). Deteksi dan tracking objek secara real time berbasis Computer vision menggunakan metode YOLO V3. Humantech: Jurnal Ilmiah Multidisiplin Indonesia, 3(3). https://doi.org/10.32670/ht.v3i3.4348.
[24] Yuliany, S., & Rachman, A. N. (2022). Implementasi Deep Learning pada Sistem Klasifikasi Hama Tanaman Padi Menggunakan Metode Convolutional Neural Network (CNN). Jurnal Buana Informatika, 13(1), 54-65. https://doi.org/10.24002/jbi.v13i1.5022.
[25] Yusuf, M. N., & Hakim, D. L. (2020). Analisis Saluran Pemasaran Komoditas Padi (Studi Kasus di Desa Selasari Kecamatan Parigi Kabupaten Pangandaran). Jurnal Ilmiah Mahasiswa AGROINFO GALUH, 7(1), 97-111. http://dx.doi.org/10.25157/jimag.v7i1.2563.