

AI-Based Plant Pest Detection and Treatment Recommendation Using YOLOv8

Ashitha D ¹, T Chandraiah ², Prem Singh M ³, Poornima B H ⁴, Basavanna M ⁵

¹Department of Studies in Computer Applications (MCA), Davangere University, Davangere, Karnataka, India.
ashithadevlanaik962@gmail.com

²Department of Computer science, Yuvaraja College, University of Mysore, Mysore, Karnataka, India.
chandruycm@gmail.com

³Department of Computer Science, Government College (Autonomous), Mandya, Karnataka, India.
premmingsingh1973@gmail.com

^{4,5}Department of Studies in Computer Science, Davangere University, Davangere, Karnataka, India.
poornima.bh@davangereuniversity.ac.in, basavanna.m@davangereuniversity.ac.in

ABSTRACT

Plant pests are a major cause of crop damage, reduced agricultural productivity, and economic losses for farmers. The traditional way of pest identification is done manually by agricultural professionals, which is time-consuming, labour-intensive, and not always easy for farmers in remote areas. To overcome these challenges, this work introduces an AI-based Plant Pest Detection and Treatment Recommendation System that uses an advanced deep learning method to identify pests quickly and accurately. The proposed system is based on the YOLOv8 (You Only Look Once Version 8) object detection model, trained with PyTorch, which will detect multiple kinds of pests from the uploaded crop images. OpenCV is used to preprocess the images and annotate them, and Flask is used to create an easy-to-use web application for uploading images and displaying the results.

Keywords: Pest Detection, OpenCV, Deep Learning, YOLOv8

I. INTRODUCTION

Plant pests and diseases cause 20-40% of the losses in global crop production annually, a significant threat to global food security, according to the Food and Agriculture Organisation (FAO). In addition, plant diseases cause at least USD 220 billion in losses annually, and invasive insect pests cause at least USD 70 billion in losses worldwide. Identification of crop pests is more critical than ever because of the effects of climate change on their spread and intensification.

To overcome these challenges, this project introduces an AI-Based Plant Pest Detection and Treatment Recommendation System that leverages the YOLOv8 deep learning algorithm, PyTorch, OpenCV, Flask, and MongoDB. Automatically recognizes pests in uploaded photos, provides chemical controls, organic controls, prevention tips, pest detection data, and can create downloadable PDF reports, with a confidence level indicated for the recognized pest. It enhances capabilities over the traditional system of pest detection by website by adding real-time pest detection, treatment recommendations, confidence analysis, pest detection history management, storing data in MongoDB, and finally generating a PDF report for the pest detection. The integrated solution would help farmers and agricultural professionals to make faster and better decisions.

Problem Statement

Our problem is to provide a system that can automatically detect pests in a picture of a plant. What's interesting is that this system needs to be able to identify types of pests and be easy to use. We're using a computer program called YOLOv8 to make this system. Agriculture is among the industries that are undergoing a transformation due to AI. Predictive tools rely on AI to analyze vast amounts of data and identify patterns to make predictions. This helps farmers to be able to make decisions. AI can assist with various production issues, including pest and weather management, waste reduction, and yield enhancement.

Objectives

1. To prepare and label image data to enable robust training of deep learning models.
2. To apply the algorithm for pest identification and classification YOLOv8 based object detection.
3. Train the model and assess its performance using suitable performance measures like accuracy, precision, recall, and F1 score.
4. To create a user-friendly interface to enable users to upload images and get the pest prediction easily.
5. To provide prompt and accurate pest detection results for prompt and appropriate agricultural decision-making.

Sample Images

For this paper, a variety of sample images of crop leaves were taken from the Kaggle repositories. This Project is divided into 15 sections, which are Pest Detection includes:

This Paper consists of **12 Pest Detection** includes:

The dataset used in the proposed **Plant Pest Detection System** consists of images representing different types of agricultural pests. The dataset contains 12 pest classes: Ants, Bees, Beetles, Caterpillars, Earthworms, Earwigs, Grasshoppers, Moths, Slugs, Snails, Wasps, and Weevils. The dataset focuses specifically on pest identification and does not include a separate healthy-leaf class.



Fig. 1 Sample Input Images of Pest Detection

The data for this project was collected from public databases of agricultural pest images and from the internet, to create a robust model for the detection of pests. It features some 11,500 annotated images of 12 classes of pests

<https://www.kaggle.com/datasets/vencerlanz09/agricultural-pests-image-dataset>

The dataset was divided into training, validation, and test sets, and processed using annotated bounding boxes like YOLO. This wide variety helps the YOLOv8 model to identify and classify various pests effectively in real-world agricultural settings.

The images were collected from different sources and represent variations in pest appearance, size, shape, orientation, and background conditions. These variations help the deep learning model learn important visual characteristics of different pest classes and improve its ability to identify pests in different image conditions.

II. RELATED WORK

[1] Fuentes et al. (2024) proposed a real-time tomato pest and disease detection system using deep learning algorithms Faster R-CNN and SSD. When released in the field, the system was able to detect multiple tomato pests with high detection accuracy, proving the effectiveness of deep learning in precision agriculture. Research Gap: Limited classes of pests and tomato crops were developed. Limitation: It was a complicated process and couldn't be implemented on a lightweight device for real-time use. To facilitate the recognition of agricultural pests using

[2] AI, Wu et al. (2024) created a large-scale benchmark (IP102) for agricultural pest recognition, comprising more than 75,000 images of 102 agricultural pest types. Training and testing of deep learning models: The dataset was used to train and test deep learning models for a variety of pest species. Research Gap: This work was more focused on creating a dataset than building an efficient detection model. Limitation: The dataset is imbalanced, and pests appear in varying numbers, so it is hard to detect the pests.

[3] Zhou et al. (2025) summarized the recent deep learning techniques employed in plant disease and plant pest detection. Contribution: Review consolidated CNN-, YOLO-, and Transformer-based solutions, and identified their advantages for agricultural monitoring. Gap in research: Current techniques require a large annotated dataset and computational resources. Challenge: Many models are yet to be able to generalise in complex field environments.

[4] The unified detection model JOLO-DP was proposed by Zhou and Wei (2025), which is used to detect 15 common rice diseases and pests. The model advanced the identification of multiple diseases and pests at once in paddy field conditions with the utilization of an improved YOLO architecture. Research Gap: The system developed here is suggested to be applicable to rice crops only and can't be directly applied to other crops in agriculture. Limitation: Uneven illumination, background clutter, and overlapping pest objects have an impact on detection accuracy.

[5] The detection accuracy of pests: Yang et al. (2025) developed an improved YOLOv8 model named IP-YOLOv8, which used multi-scale feature fusion and a dynamic detection head. Results: The model performs better than YOLOv8. Research Gap: Complex model optimisation. Limitations: Misclassifications continued for similar species of pests.

III. METHODOLOGY

The methodology begins with the collection of a diverse pest image dataset containing different pest species captured under various environmental conditions. The dataset is then annotated to identify pest locations and prepare it for model training. During preprocessing, the images undergo resizing, normalisation, and noise reduction to improve image quality and ensure compatibility with the YOLOv8 model. The annotated dataset is divided into training, validation, and testing sets to evaluate the model's performance and improve its generalization ability.

1. Requirement Analysis

The project needs are collected and analysed in this phase. System goals, function requirements, and user needs are determined. The features include pest image upload, real-time pest detection via YOLOv8, treatment recommendations, detection history management, integration with MongoDB, and PDF report generation are finalized. The software and hardware requirements are also determined.

2. System Design

The general architecture of the system is designed at this stage. The database structure, user interface, application flow, and module interrelationships are designed. The technologies used are chosen: Flask, OpenCV, PyTorch, YOLOv8, MongoDB, HTML, and CSS.

3. Implementation (Coding)

During the implementation stage, the developed modules are created with the selected technologies. YOLOv8 model is trained on the pest image dataset, and the image preprocessing is implemented using OpenCV. The web application is developed in Flask, the detection history is stored in MongoDB, and the treatment recommendations are included by implementing the pest remedy module. The development of all modules is integrated into a complete working system.

4. Integration and Testing

After coding, all the developed modules are combined into a complete system in the integration phase. Once integrated, the system undergoes thorough testing to check for defects, ensure functionality, and verify that the system meets the requirements specified in the SRS.

Different testing levels such as unit testing, integration testing, system testing, and acceptance testing are performed. The primary objective here is to deliver a bug-free and reliable product.

4. Testing

The system is tested to check its functionality, accuracy, and performance. The following testing is conducted to check the correct working of modules: Unit testing, Integration testing, System testing, and User acceptance testing.

5. Deployment

Once tested successfully, the application is deployed to the end users. The configuration of the trained YOLOv8 model, Flask application, and MongoDB database is done in the deployment environment. The web interface allows users to upload pest images, retrieve detection results, and get treatment suggestions, along with a history of detections and PDF reports.

6. Maintenance

The last stage of the Waterfall model is "maintenance". The system is continually monitored to correct bugs, increase detection accuracy, expand the pest database, etc., to make better treatment recommendations, and to improve the overall performance of the system.

System Architecture

The complete workflow of the proposed ai based plant pest detection system is shown in the system architecture. The process starts with the user uploading a picture of a plant with pests in it to the Web Application Interface (Flask/Streamlit UI). The image uploaded is then processed in the Image Preprocessing Module, where image size, noise removal, and normalisation are carried out to enhance image quality.

The proposed system is significantly improved in terms of usability and accessibility by the use of the frontend layer. Services Layer: Prior to the AI model analyzing the image, the Services Layer processes the image and prepares the data. Image preprocessing enhances the quality of images and prepares the input data to be analysed by the model.

ARCHITECTURE DIAGRAM

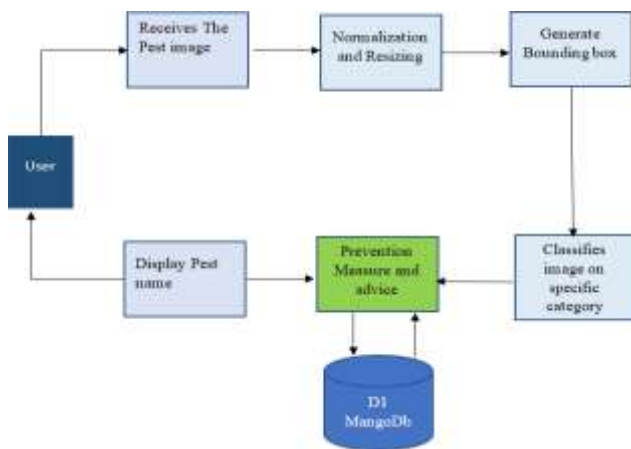


Fig. 2 System Architecture

Workflow Explanation

1. Input: Users load crop or pest images via the web app interface. The images are sent to the Image Preprocessing Module, where image operations, including resizing, normalization, noise removal, and image enhancement, are carried out that would enhance the quality of the images before analysis.

2. Preprocess Image: The enhanced images are passed to the Pest Detection Module, which utilizes the YOLOv8 model to detect any pest object in the image and generate bounding boxes around the detected pests.

3. Detected Pest Objects: Detected pest information is transmitted to the Pest Classification Module, where the system classifies the detected pest to a specific category, based on the trained dataset.

The information on the pest category is passed to the Treatment Recommendation Module, from which the appropriate pest management information is retrieved from the database.

Treatment Recommendation Module: This module will offer you:

- Recommended pesticides
- Dosage instructions
- Preventive measures
- Organic control methods
- What to do if there is beep or buzzer activity.
- Crop protection tips.

The answer is that the system stores all the pest categories, treatment information, pesticide recommendations, and prediction history in MongoDB. The Recommendation Module fetches treatment information from the database for each pest and keeps a record of the pest predictions for

future use.

4. The result display module: The result is displayed to the user through the web application interface, including the detected pest name, confidence score, recommended pesticides, and preventive measures. Besides, the results generated will be saved in the database for Prediction History, so users can check past detection results and see how pests were detected in the past.

Each processing step is a part of producing a comprehensive Pest Detection Result that provides both pest identification and treatment recommendations to take action.

The final recommendation report is presented to the user, allowing for informed decision-making when using pest management and crop protection.

Algorithm: CNN algorithm for pest detection

Input: Crop leaf image dataset $D = \{x_1, x_2, \dots, x_n\}$ with pest labels $Y = \{y_1, y_2, \dots, y_n\}$

Output: Detected pest type, bounding box, and confidence score.

1. Collect Dataset

Collect crop leaf images containing different types of pests and healthy leaves.

2. Image Preprocessing

- Resize all images to **640 × 640 pixels**.
- Normalize pixel values.

3. Normalization:

$$x' = 255x$$

Where x is the original pixel value and x' is the normalized value.

4. Initialize YOLOv8 Model

Use the pre-trained **YOLOv8n model** and fine-tune it using the pest dataset.

5. Feature Extraction

The YOLOv8 backbone extracts important features from the input image, such as **shape, texture, color, and edges** of the pest.

Feature Fusion

The YOLOv8 neck combines features from different scales to detect both small and large pests.

Pest Detection

The YOLOv8 detection head predicts:

- Pest class
- Bounding box
- Confidence score

6. Confidence Score:

$$\text{Confidence} = P(\text{Object}) \times \text{IoU}$$

Where $P(\text{Object})$ is the probability that a pest is present and IoU measures how well the predicted bounding box matches the actual pest location.

1. Classification
The detected pest is classified into its corresponding category, such as Aphid, Whitefly, Bollworm, Beetle, or Healthy.
2. Performance Evaluation
The model is evaluated using Accuracy, Precision, Recall, and F1-Score.

Accuracy: 1.

Accuracy

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

2. Precision

$$\text{Precision} = \frac{TP}{TP+FP}$$

Where:

- TP = True Positive (correctly detected pest)
- TN = True Negative (correctly detected healthy leaf)
- FP = False Positive (wrongly predicted pest)
- FN = False Negative (missed pest)

3. Recall:

$$\text{Recall} = \frac{\text{Total Actual Pests Correctly Detected Pests}}{\text{Total Actual Pests}}$$

TECHNOLOGY USED

• Python

The Plant Pest Detection System is created using the main programming language: Python. Provides an easy and effective AI/ML/Computer Vision application development platform. The development of the backend, training of the model, processing the images, integration with the database, and system integration are made with Python. With its extensive library and frameworks, it's the language used in agricultural AI applications.

YOLOv8 (You Only Look Once Version 8)

The deep learning model that has been developed and widely used for the detection and classification of pests is YOLOv8,

a state-of-the-art object detection model developed by Ultralytics, which can detect and locate multiple pests in a single image. It is accurate, quick, and has real-time detection capability, applicable for agricultural monitoring systems.

• OpenCV

OpenCV (Open Source Computer Vision Library) is used to perform Computer Vision and image processing. It can resize images, normalize, remove noise, and perform image

enhancement, and visualization of detection results. The image quality enhancement feature in OpenCV, before analysing it by the YOLOv8 model, improves the accuracy of image detection.

• Flask

Flask is a simple Python-based web framework that is used to create the web-based user interface and back-end services. It uploads images, communicates with the AI model, responds to the user's input, and displays the results of image detection. The combination of flask makes it easy to communicate back and forth between the front-end interface and processing modules in the back end.

• PyTorch

PyTorch is an open-source Deep Learning Framework for training and deployment of YOLOv8. It provides efficient tensor operations, automatic differentiation, and GPU acceleration, which are essential for training large-scale deep learning models. A model is optimized and built in PyTorch for pest detection.

• NumPy

What do you know about NumPy? NumPy is a numerical computing library that works with multidimensional arrays and numerical computations. It can assist in the manipulation of image data, matrix operations, and numerical processing required for the training of models and image preprocessing.

• Pandas

A powerful data handling library for data analysis and manipulation is pandas. It helps organize pest image datasets, analyze training image data, and annotate images. Pandas offers efficient data structures that make data easier to handle when developing the models.

Frameworks & Models

This is the YOLOv8 Object Detection Model. This is the YOLOv8 Object Detection Model.

In the proposed system, the main model is YOLOv8, which is an Artificial Intelligence model. It detects, localizes, and classifies pests in a single step, which will greatly decrease the process time while ensuring high accuracy. The model predicts bounding boxes around each pest and outputs scores of confidence for each prediction.

• Ultralytics Framework

YOLOv8 is trained in the Ultralytics framework and trained. It simplifies the training, validation, testing, and deployment of models and performs an object detection task better.

• Computer Vision Framework

The system employs computer vision techniques to extract meaningful information from pictures of pests. The framework can be used to process the image and extract features, and then use the AI model to identify different species of pests.

• Deep Learning Framework

The image information of pests can then be automatically learned to identify the visual patterns and features with deep learning methods. The trained neural network is able to recognize complex features like shape, color, texture, and structural patterns that can discriminate various types of pests.

IV. Results and Analysis

The architecture of the proposed system is designed as a modular system with the aim of improving the scalability, maintainability, and efficiency of processing. The system is divided into independent functional modules, including image acquisition, processing, pest recognition, classification, suggestion of treatment, management of the database, and the presentation of the results.

Input Handling

Image Preprocessing

The uploaded images undergo preprocessing to improve their quality before AI analysis.

The preprocessing stage includes:

- Image resizing

- Image normalization
- Noise removal
- Contrast enhancement

These operations ensure that the image meets the input requirements of the YOLOv8 model and improve detection accuracy.

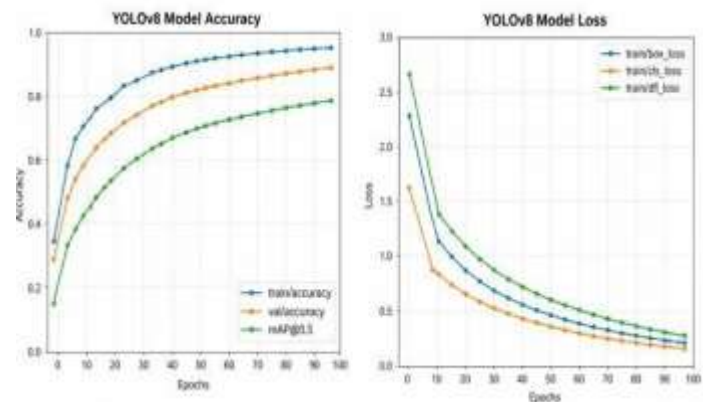


Fig. 1 Performance Graph

Table 1 Comparison table between CNN and YOLOV8 based on the accuracy and loss values.

Models	Accuracy (%)	Loss Value
CNN	98.7	0.05
YOLOV8	92.1	0.18

- **CNN:** CNN achieves an accuracy of 98.7% with a loss value of 0.05. The high accuracy and low loss indicate that the CNN model performs well in classifying the pest images.
- **YOLOv8:** YOLOv8 achieves an accuracy of 92.1% with a loss value of 0.18. Although its accuracy is lower than CNN in this comparison, YOLOv8 performs object detection, meaning it can identify the pest class and locate the pest using bounding boxes

Table 2. Reported Performance of Existing Deep Learning Methods and the Proposed System

Method	Task	Dataset / Benchmark	Reported Result
Faster R-CNN + VGG-16 [1]	Object Detection	Tomato Diseases and Pests Dataset	83.06% Mean AP
ResNet-50 [2]	Image Classification	IP102	49.4% Accuracy
Proposed CNN [3]	Image Classification	NBAIR, Xie1, Xie2	96.75%, 97.47%, 95.97% Accuracy, respectively
TSCNNA [4]	Object Detection	Common Crop Pest Dataset	93.16% Precision
IP-YOLOv8 [5]	Object Detection	IP102	60.1% Precision; 54.2% Recall; 59.2% mAP50; 38.4% mAP50:95
AEC-YOLOv8n [6]	Object Detection	IP102	58.9% Precision; 62.6% Recall; 67.1% mAP50; 43.1% mAP50:95
ExquisiteNet [7]	Image Classification	IP102	52.32% Test Accuracy
InsectNet [9]	Insect Classification	Citizen-Science Insect Dataset	>96% Mean Per-Class Accuracy
Proposed YOLOv8	Pest Detection and Classification	12-Class Agricultural Pest Dataset	92.1% Accuracy; 0.18 Loss

Snapshot



Fig. 2 Home page



Fig. 3 Detection Page



Fig. 4 Result Page



Fig 5 View Detection History

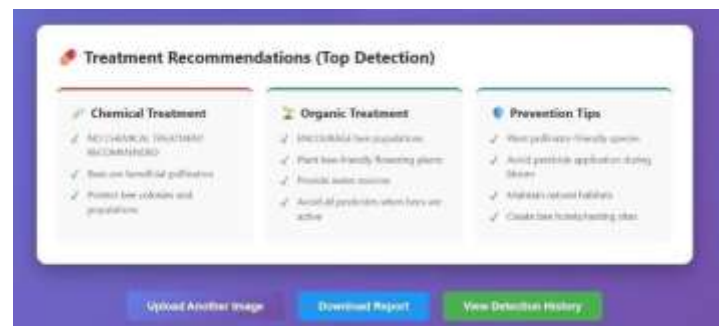


Fig 6. Treatment Recommendation

YOLOv8 is a single-stage object detection model that can predict in real time and with high accuracy. MongoDB delivers a fuss-free way to access treatment advice and a treatment prediction history quickly and efficiently. The modular design enables the separate update of the various components, including the detection model, the environmental data, or the list of pests, and the corresponding treatment recommendations.

CONCLUSION

The Plant Pest Detection system is powered by the YOLOv8 object detection model, which enables detection and classification of a few pest species from images of crops uploaded at the time of analysis. The proposed system can reduce the identification time and labor as well as improve the identification accuracy compared with manually inspecting and expertly identifying the pests. Compared with traditional image classification methods, bounding box detection can be used to precisely detect pests in the image and is more effective.

To conclude, the system proposed is fruitful at achieving the overall aim of the research, such as actual detection of pests, smart recommendation of treatment, and significant presentation of direction in pest management. It can help minimize losses, boost productivity, decrease the amount of pesticides that are used, and assist farmers in using more efficient and sustainable farming methods. Overall, the study shows the promise of AI in solving real-world challenges in agriculture and in advancing the development of smart farming.

V. REFERENCE

- [1] A. Fuentes, S. Yoon, S. C. Kim, and D. S. Park, "A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition," *Sensors*, vol. 17, no. 9, Art. no. 2022, 2017, doi: 10.3390/s17092022.
- [2] X. Wu, C. Zhan, Y.-K. Lai, M.-M. Cheng, and J. Yang, "IP102: A large-scale benchmark dataset for insect pest recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 8787–8796, doi: 10.1109/CVPR.2019.00899.
- [3] K. Thenmozhi and U. S. Reddy, "Crop pest classification based on deep convolutional neural network and transfer learning," *Computers and Electronics in Agriculture*, vol. 164, Art. no. 104906, 2019, doi: 10.1016/j.compag.2019.104906.
- [4] X. Wang, S. Zhang, X. Wang, and C. Xu, "Crop pest detection by three-scale convolutional neural network with attention," *PLOS ONE*, vol. 18, no. 6, Art. no. e0276456, 2023, doi: 10.1371/journal.pone.0276456.
- [5] C. Yang, Y. Wang, L. Yun, H. Wang, Y. Han, and Z. Chen, "IP-YOLOv8: A multi-scale pest detection algorithm for field-scale applications," *Horticulturae*, vol. 11, no. 9, Art. no. 1109, 2025, doi: 10.3390/horticulturae11091109.
- [6] J. Wei, H. Gong, S. Li, M. You, H. Zhu, L. Ni, L. Luo, M. Chen, H. Chao, and Y. Sun, "Improving the accuracy of agricultural pest identification: Application of AEC-YOLOv8n to large-scale pest datasets," *Agronomy*, vol. 14, no. 8, Art. no. 1640, 2024, doi: 10.3390/agronomy14081640.
- [7] S.-Y. Zhou and C.-Y. Su, "Efficient convolutional neural network for pest recognition—ExquisiteNet," in *Proc. 2nd IEEE Eurasia Conf. IoT, Commun. Eng. (ECICE)*, 2020, pp. 216–219.
- [8] J. White, C. Agrawal, A. Ojha, A. Agnihotri, M. Sharma, and J. Doshi, "BOLLWM: A real-world dataset for bollworm pest monitoring from cotton fields in India," in *Proc. PML4DC Workshop at ICLR*, 2023.
- [9] S. Chiranjeevi, M. Saadati, Z. K. Deng, J. Koushik, T. Z. Jubery, D. S. Mueller, M. E. O. Neal, N. C. Merchant, A. Singh, A. K. Singh, S. Sarkar, A. Singh, and B. Ganapathysubramanian, "Deep learning powered real-time identification of insects using citizen science data," *arXiv preprint arXiv:2306.02507*, 2023, doi: 10.48550/arXiv.2306.02507.
- [10] M. Shoaib, A. Sadeghi-Niaraki, F. Ali, I. Hussain, and S. K. Shah, "Leveraging deep learning for plant disease and pest detection: A comprehensive review and future directions," *Frontiers in Plant Science*, vol. 16, 2025, Art. no. 1538163, doi: 10.3389/fpls.2025.1538163.
- [11] [12] S. Banerjee, T. Mallick, A. Chakraborty, H. N. Saha, and N. T. Thakur, "Evaluation of state-of-the-art deep learning techniques for plant disease and pest detection," *arXiv preprint arXiv:2508.08317*, 2025.
- [12] [13] Y. Zhou and Y. Wei, "YOLO-DP: A unified detection model for rice diseases and pests using an improved YOLO architecture," *Scientific Reports*, vol. 15, 2025.
- [13] [14] M. Ejaz, M. A. Khan, S. Ahmed, et al., "Deep learning techniques for crop pest classification: A comprehensive review," *arXiv preprint arXiv:2507.01494*, 2025.
- [14] [15] A. Allmendinger, A. O. Saltık, G. G. Peteinatos, A. Stein, and R. Gerhards, "Assessing the capability of YOLO- and transformer-based object detectors for real-time weed detection," *arXiv preprint arXiv:2501.17387*, 2025.
- [15] A. Allmendinger, A. O. Saltık, G. G. Peteinatos, A. Stein, and R. Gerhards, "Assessing the capability of YOLO- and transformer-based object detectors for real-time weed detection," *arXiv preprint arXiv:2501.17387*, 2025.
- [16] [16] V. Kotthapalli, R. Sapkota, and M. Karkee, "Evolution of YOLO models: A review from YOLOv1 to YOLOv11 for real-time object detection," *arXiv preprint*, 2025.
- [17] [17] A. Upadhyay, P. Sharma, R. Singh, and S. Kumar, "Deep learning and computer vision techniques for plant disease detection in precision agriculture: A review," *Artificial Intelligence Review*, vol. 58, 2025.
- [18] [18] M. Shafay, A. Hassan, M. Ali, and S. Ahmed, "Recent advances in artificial intelligence for plant disease detection: A review," *Plant Methods*, vol. 21, no. 1, 2025.