International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

YOLO-DRIVE INTERNET OF THINGS BASED COMPUTER-AIDED DIAGNOSIS SYSTEM FOR OLIVE LEAF DISEASES RECOGNITION

Saud Saad Alqahtani[1], Hossam El-Din Moustafa[1,2] and Elsaid A. Marzouk[1,2], Ramadan Madi Ali Bakir[3,4, *]

[1] Electronics and Communications Engineering Department, Faculty of Engineering, Mansoura University, Mansoura 35516, Egypt

[2] the Dean of the Faculty of Artificial Intelligence, Horus University, New Damietta 34518, Egypt

[3] Semiology Department, National Research Institute of Astronomy and Geophysics (NRIAG), Cairo 4037101, Egypt

[4] Nahda University, Faculty of Engineering, Department of Communication and Computer Engineering, Beni Suef 65211, Egypt.

* Correspondence: Author: ahr_bakir@yahoo.com

Abstract

Addressing the existing gap in the practical deployment of deep learning models for agricultural disease diagnostics, this paper presents OLIVE-CAD, an Internet of Things (IoT) enabled Computer-Aided Diagnostics system specifically engineered for the real-time recognition of olive leaf diseases. The system integrates a high-resolution camera module for image acquisition, an embedded microprocessor for on-site processing, and an IoT module for remote data transmission via PAHO-MQTT to a cloud platform. For the core diagnostic component, a YOLOv12 Convolutional Neural Network (CNN) model was developed and trained on a comprehensive dataset of 11,315 olive leaf images, categorized into 'Aculus', 'Scab', and 'Healthy'. This dataset underwent meticulous preparation, including rigorous bounding box annotation and manual review to ensure data quality. The novelty of this work lies in its end-to-end integration of a custom-built, on-site hardware solution with a high-performance deep learning model and a scalable IoT data pipeline. This provides a practical, field-deployable system rather than a theoretical model. The trained YOLOv12 model achieved a high performance, with a mean average precision (mAP@0.5) of 98.2% and class-specific accuracies of 97% for 'healthy', 84% for 'aculus', and 95% for 'scab'. These results translate into significant benefits for agricultural management, including real-time diagnostics for early disease detection, which enables swift intervention and reduces crop loss. Furthermore, the system's ability to transmit structured data provides farmers with data-driven insights for monitoring disease progression over time. System validation involved real-life data capture and processing from 60 printed samples (20 samples for each class: healthy, aculus, and scab) to simulate field conditions. The model successfully classified the majority of these samples with high accuracy, a finding that is consistent with the model's confusion matrix. The diagnostic results were smoothly

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

transmitted to a cloud-based PostgreSQL database with TimescaleDB for efficient storage, and presented on a Grafana dashboard for real-time data viewing.

Keywords: Olive leaf Disease; OLIVE-CAD system; YOLO; IoT; PostgreSQL; Grafana

1. Introduction

Reviewing the existing literature [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11]reveals a prevailing focus among researchers on developing sophisticated deep learning models for olive leaf disease identification. While these efforts have yielded impressive diagnostic accuracies, a notable gap exists in the practical implementation of these models within real-world agricultural settings. Current research primarily concentrates on data collection and model optimization for improved model quality and accuracy, often deferring practical application to future work. Common recommendations for practical deployment include drone integration for direct farm sampling or the development of mobile applications[12]. However, these approaches present inherent limitations; drone deployment may face security constraints, and mobile applications could be hindered by unreliable internet connectivity for data transmission to IoT platforms or centralized data collection centers[13]. This research addresses this critical gap by prioritizing the practical application of olive-leaf disease detection models. Consequently, our focus shifts from an extensive comparative analysis of model performance metrics to identifying the most suitable model for integration into a practical system. The emphasis will be placed on the hardware and software components necessary for real-time disease identification and seamless data transmission to an IoT server.

Our research delivers several significant contributions:

- **Custom YOLOv12 Model Development and Annotation Pipeline**: We developed a specialized YOLOv12 Convolutional Neural Network model for olive leaf disease detection. This involved creating a custom Python annotation code for image labeling, followed by manual review to ensure annotation accuracy. The model was then rigorously trained and achieved a peak accuracy of 98.2%.
- **IoT System Integration for Real-time Diagnostics**: The trained YOLOv12 model was seamlessly integrated with a custom-built IoT hardware system, which includes a high-resolution camera, a Raspberry Pi microprocessor, and a cellular modem. This integration enables the real-time recognition of olive leaf diseases directly in the field.
- **Real-world System Validation**: The OLIVE-CAD system underwent successful validation in a real-world setting. We captured and processed images from 60 diseased olive leaves, and the embedded YOLOv12 model accurately classified most samples, with results efficiently transmitted to an IoT cloud platform.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The organization of the paper, **Section II**: The research methodology; **Section III**: experimental results; and Finally**, Section V**: summery and future work.

2. Methodology

The proposed methodology, comprehensively outlined in Figure 1, details the systematic approach undertaken to develop, train, and deploy the OLIVE-CAD system for the real-time recognition of olive leaf diseases. This multi-faceted process begins with the meticulous preparation and annotation of a large-scale image dataset, followed by the development and rigorous evaluation of a custom YOLOv12 deep learning model. Finally, the methodology culminates in the seamless integration of the trained model with a purpose-built IoT hardware system for real-world application and subsequent validation in an agricultural setting.

2.1.Dataset description and preparation

The methodology begins with the meticulous preparation of a comprehensive olive leaf image dataset. This dataset, totaling 11,315 samples, was collected from three distinct sources [14], [15], [16] to ensure a diverse representation of olive leaf conditions. The detailed composition of this dataset is presented in Table 1: Dataset Composition. As shown the dataset is structured into three primary classes: "Healthy" olive leaf samples, and two significant olive leaf diseases, "Aculus olearius" and "Peacock spot (olive scab)". Aculus olearius, commonly known as the olive bud mite, causes deformations and discoloration of olive leaves, often leading to reduced yield [17]. Peacock spot, or olive scab, caused by the fungus Spilocaea oleaginea, is characterized by circular, dark spots with a light halo on the leaves, severely impacting photosynthesis and overall tree health[18]. Visual examples of these disease symptoms are presented in Figure 2. This comprehensive dataset, encompassing both healthy and diseased samples with clear visual indicators of these specific maladies, serves as the foundational input for the entire deep learning pipeline.

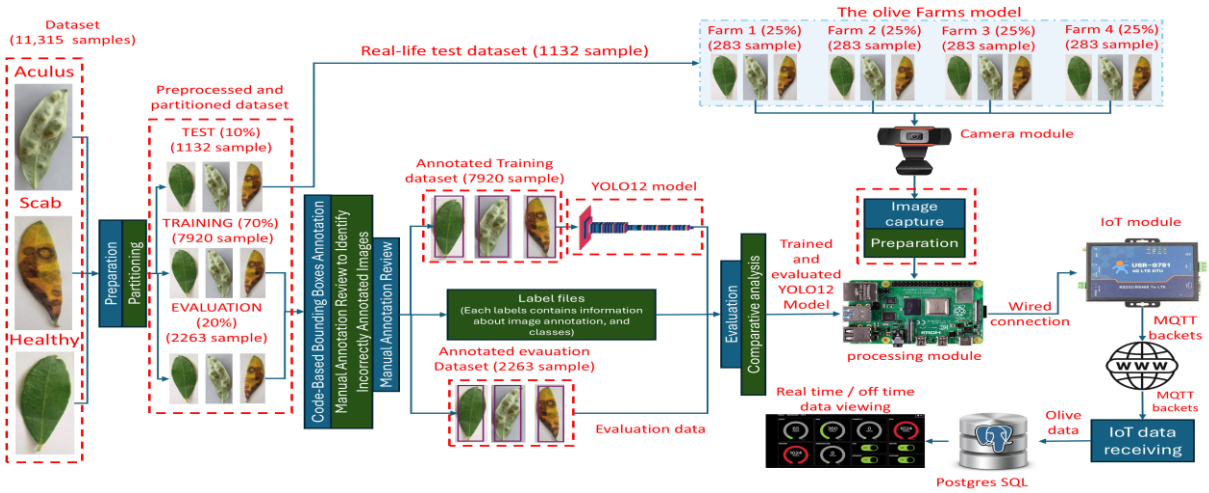


Figure 1. Comprehensive Methodology for OLIVE-CAD System Development and Deployment.

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Table 1. Detailed Composition of the Olive Leaf Disease Dataset, showcasing the distribution of healthy, Aculus olearius, and Peacock spot (olive scab) samples across various sources

Dataset	Reference	Source	Total/Dataset	Health	Aculus olearius	Peacock spot	Total
Dataset 1	[14]	Kaggle	3400	1,050	1,460	890	11315
Dataset 2	[15]	Kaggle	6961	1,926	1,778	3,257	
Dataset 3	[16]	Kaggle	954	572	382 (in one file, manually separated)		

Following data collection, the Preparation Partitioning stage is critical for effective model training and unbiased evaluation. The entire dataset (11,315 samples) is first logically divided as detailed in Table 2: Dataset Partitioning Strategy for Model Training and Evaluation. As shown in Table 2, a "real-life Test dataset" comprising 10% (1132 samples) of the total dataset is specifically set aside for the final assessment of the trained IoT model's real-world performance, as depicted feeding into "The olive Farms model". The remaining 90% of the dataset

is then further partitioned for the YOLOv12 model development. This portion is split into a "Training" subset of 70% (7920 samples) and "Evaluation" subset of 20% (2263 samples).

Figure 2. Representative olive leaf samples showing Healthy, Aculus olearius, and Peacock spot conditions

Table 2. Breakdown of the 11,315 Olive Leaf Dataset Samples into Training, Testing, and Evaluation Subsets for Model Development and Validation.

Dataset samples	Subset	Percentage	Samples	Description
11315	Training	70%	7920	Used to train the YOLOv11 deep learning model.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

	Evaluation	20%	2263	Used to evaluate the performance of the trained YOLOv11 model
	Test	10%	1132	Dedicated for final real-world validation of the integrated IoT system.

2.2.Image Annotation

The Annotation Stage is a crucial process that transforms raw image data into a format interpretable by the YOLOv12 object detection model. This stage, detailed in the flowchart in Figure 3: Automated Image Annotation Process Flowchart, begins with Code-Based Bounding Boxes Annotation. For each image in both the training and evaluation subsets, the process initializes parameters, creates output folders, and then loads each image. If an image loads successfully, it undergoes a series of pre-processing and segmentation steps: applying a Gaussian blur [19], converting to HSV color space [20], defining specific HSV color ranges to isolate green leaf areas, and creating a mask through color filtering [21]. This mask is then refined using further Gaussian blurring and morphological operations to produce a clean final mask. Subsequently, contours are detected from this refined mask. If contours are found, the system enters a loop to identify the "best leaf contour" by calculating and scoring various properties for each contour, including its area, aspect ratio, solidity, and its centeredness within the image. The contour with the highest score is selected. Once the best leaf contour is identified, a padded rectangular bounding box is then generated around it and drawn on the output image. The coordinates of this bounding box are then meticulously scaled and normalized to a 0-1 range relative to a fixed target image size (224x224 pixels), generating YOLO-compatible label files. Representative examples of these automatically generated annotations for healthy, Aculus olearius, and Peacock spot infected olive leaves are presented in Figure 4. Following this initial automated labeling, a rigorous Manual Annotation Review is conducted by human experts. This manual verification is paramount to identifying and correcting any erroneously or imprecisely annotated images, ensuring the highest possible quality and accuracy of the bounding box labels for both the "Annotated Training dataset" and "Annotated Evaluation dataset," which are vital for robust model training and reliable evaluation.

2.3.YOLOv12 Model Development and Training

The core of the diagnostic system is the YOLOv12 model, a state-of-the-art CNN specifically chosen for its real-time object detection capabilities [22]. The training of this model relies on the meticulously prepared and annotated data.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

A. Training Inputs

The primary input for training is the "Annotated Training dataset" (7920 samples). This dataset consists of olive leaf images paired with their corresponding annotation label files. The "Annotated Evaluation dataset" (2263 samples) serves a dual purpose: it acts as the validation set during the training process to monitor the model's performance on unseen data and prevent overfitting, and subsequently as the final test set for a rigorous, unbiased evaluation of the trained model.

B. Annotation Label File Content

The annotation labels are provided in the standard YOLO format, where each line in a (.txt) file corresponds to one detected object in the image [23]. The format specifies the class ID of the object followed by its normalized bounding box coordinates. The normalization ensures that the coordinates are between 0 and 1, making them independent of the image's original dimensions and suitable for models that resize inputs. The structure of each line in a YOLO label file is as follows, as detailed in Table 3. Figure 5 illustrates a sample image for each class (Healthy, Aculus olearius, Peacock spot) overlaid with its bounding box, alongside the content of its corresponding YOLO (.txt) label file, demonstrating the format of the annotation data used for training [24]. For this research, the output classes of the YOLOv12 model are defined with specific integer IDs: 0: Healthy, 1: Aculus olearius, and 2: Peacock spot

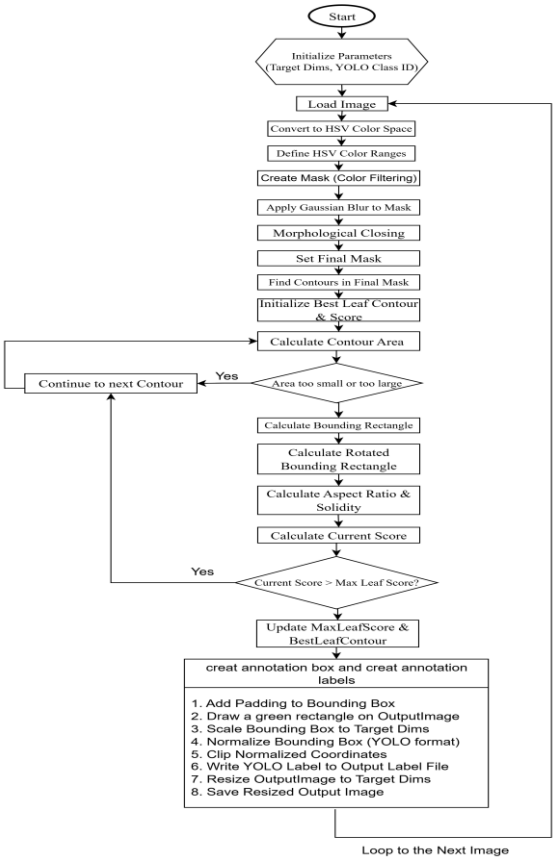


Figure 3. Automated Image Annotation Process Flowchart. This flowchart details the step-by-step procedure for the code-based bounding box annotation.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

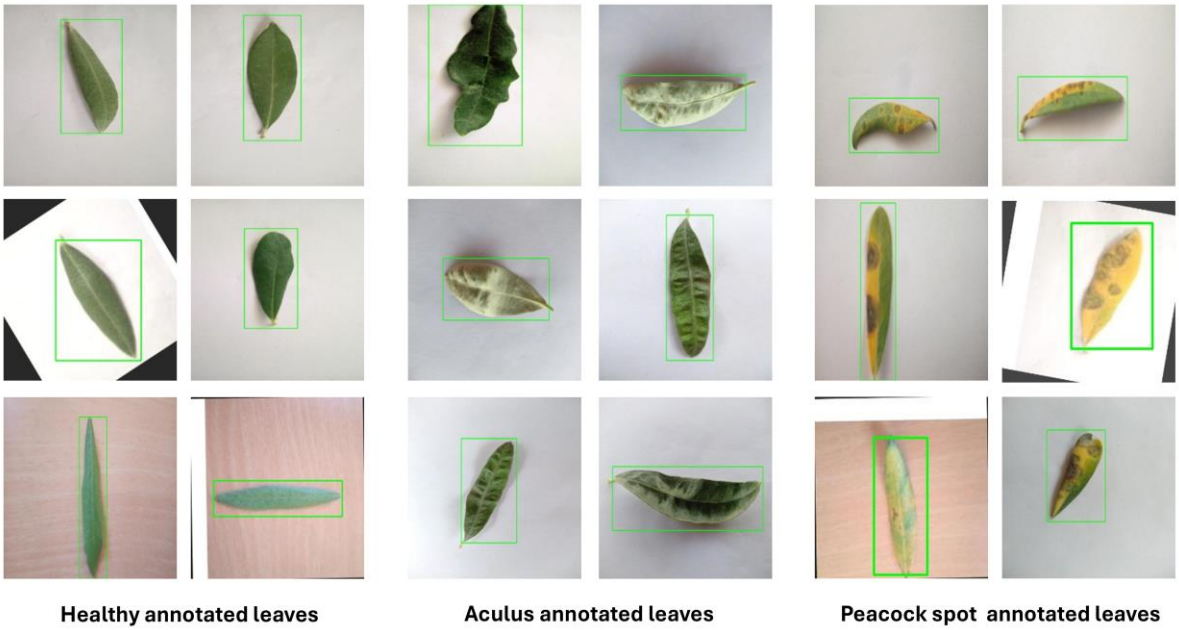


Figure 4. Sample Automatically Annotated Olive Leaf Images. This figure presents visual examples of the output from the automated annotation process.

Table 3. YOLO Annotation Label File Format

Field	**Description**	**Range**
class_id	Integer representing the object class.	0, 1, 2
x_center_norm	Normalized X-coordinate of the bounding box center.	0.0 - 1.0
y_center_norm	Normalized Y-coordinate of the bounding box center.	0.0 - 1.0
width_norm	Normalized width of the bounding box.	0.0 - 1.0
height_norm	Normalized height of the bounding box.	0.0 - 1.0

Figure 5. Sample Annotated Olive Leaf Images with YOLO Label Details

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

C. Training Process and Evaluation Metrics

During training, the YOLOv12 model iteratively adjusts its internal parameters by processing batches of images from the "Annotated Training dataset" [24]. The objective is to minimize a defined loss function, which quantifies the discrepancy between the model's predicted bounding boxes and class probabilities, and the ground-truth annotations. The validation set is periodically used to evaluate the model's performance on unseen data, guiding hyperparameter tuning and preventing overfitting. Upon completion of the training phase, the final performance of the trained YOLOv12 model is rigorously assessed using the independent "Annotated Validation dataset". This evaluation is crucial to ensure the model's generalization capability to unseen data [25]. The key metrics generated and analyzed to gauge the model's effectiveness include: Accuracy, Precision, Recall, F1-score, and recall. The trained YOLOv12 model demonstrated strong performance across these metrics, achieving a peak accuracy of 98.2% in identifying and localizing the target olive leaf conditions. This high accuracy, coupled with its real-time processing capabilities, justifies its selection as the primary diagnostic component for the OLIVE-CAD system.

2.4. The Egyptian Olive Farms Model and IoT System Integration and Validation

A. The Egyptian Olive Farms Model

Egypt holds a significant position in the global olive industry, particularly in table olive production and consumption. The country boasts approximately 165,000 hectares of olive groves, ranking tenth worldwide in cultivated area. Notably, Egypt is the world's leading producer of table olives, with a production volume of around 600,000 tons in 2022/23, and also stands as the world's leading consumer of table olives [26], [27], [28]. The Egyptian government has demonstrated an ambitious commitment to expanding this sector with an initiative to plant 100 million olive trees, over 60% of which has already been implemented across various regions. This highlights the critical economic and agricultural importance of olives to Egypt, making effective disease management a high priority. To directly address this high priority for disease management and to demonstrate its practical utility, the OLIVE-CAD system was deployed across four distinct olive farms in Egypt, representing key olive-growing regions. These farms serve as real-world examples for data collection and system performance assessment:

- **Farm A**, **Siwa Oasis**: Located in the Western Desert, Siwa is renowned for its traditional olive cultivation and high-quality olive oil, benefiting from a unique desert oasis climate.
- **Farm B**: **Fayoum Governorate** Situated southwest of Cairo, Fayoum is a fertile agricultural region known for its diverse crops, including olives, supported by irrigation from the Nile.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

- **Farm C, Ismailia Governorate**: Positioned along the Suez Canal, Ismailia is an important agricultural area with significant olive groves, benefiting from its strategic location and climate.
- **Farm D, Sadat City**: Part of the larger Wadi El Natrun and Noubaria agricultural expansion areas, Sadat City represents modern, large-scale olive cultivation projects.

A conceptual map illustrating the geographical distribution of these four example farms across Egypt is presented in Figure 6. This map visually demonstrates the diverse geographical coverage chosen for the system's validation, highlighting its potential applicability across various Egyptian olive-growing environments

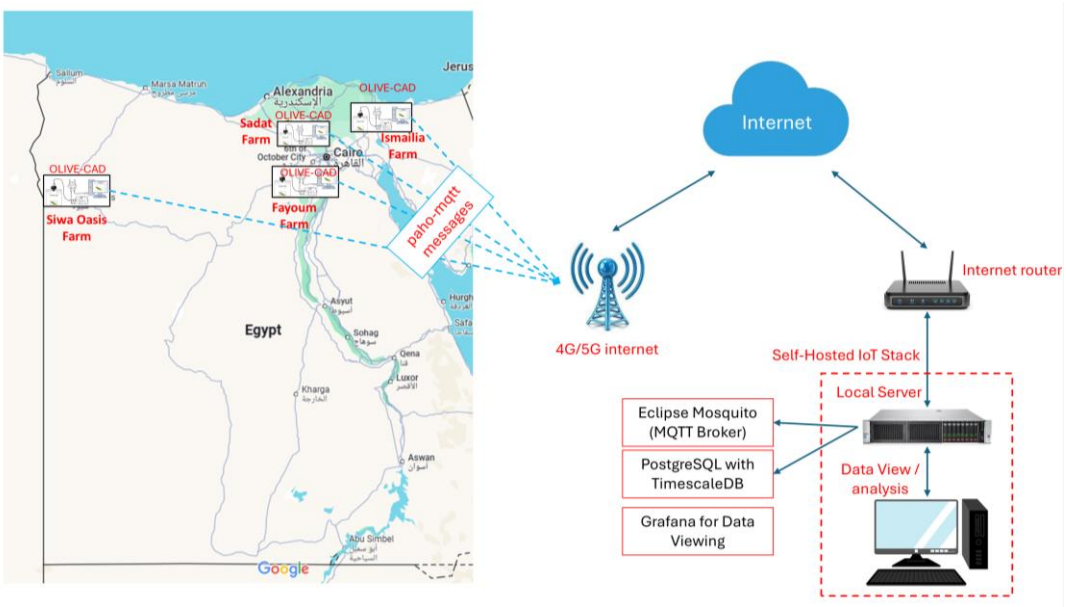


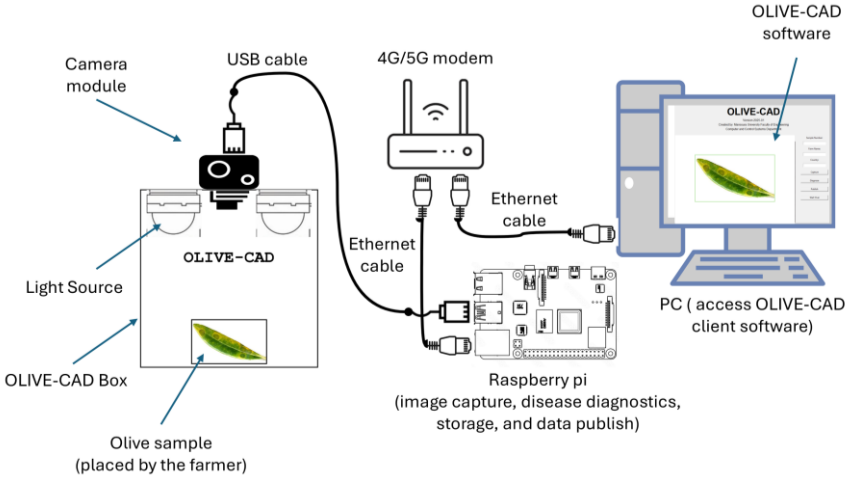
Figure 6. Conceptual Map of OLIVE-CAD Deployment Farms and System Architecture

B. IoT System Integration and Validation

The developed OLIVE-CAD system was installed in each of these four farms. As depicted in Figure 7 each farm hosts a local OLIVE-CAD unit. This unit is designed as a compact, self-contained system for on-site image acquisition and preliminary processing. At the core of each unit is an OLIVE-CAD Box, which serves as a controlled environment for placing the olive leaf samples. Inside this box, a Light Source ensures consistent and optimal illumination for image capture, minimizing variations due to external lighting conditions. A Camera module is strategically positioned to capture high-resolution images of the olive samples placed by the farmer within the box. This camera module is connected via a USB cable to a Raspberry Pi [29], [30], which acts as the central Processing module for the local unit.

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The Raspberry Pi is responsible for several critical functions: (1) Image Capture: It controls the camera module to acquire images of the olive leaves, (2) Disease Diagnostics: It hosts the Trained YOLOv12 Model, enabling real-time inference on the captured images to immediately detect and classify olive leaf diseases. (3) Storage: It provides local storage for the captured images and diagnostic results before transmission, and (4) Data Publish: It manages the communication of diagnostic results. The Raspberry Pi is connected via an Ethernet cable to a 4G/5G modem. This modem provides the necessary internet connectivity for the local OLIVE-CAD unit. The 4G/5G modem, in turn, connects to the broader Internet, facilitating secure transmission of diagnostic data to a centralized Self-Hosted IoT Stack located on a Local Server



[31], [32], [33]. The communication between the IoT module and the server is facilitated by MQTT packets [34], [35], [36], a lightweight, efficient messaging protocol ideal for constrained IoT environments, ensuring reliable data transfer even with limited bandwidth.

Figure 7. OLIVE-CAD Local Unit Architecture. This diagram shows the components of an OLIVE-CAD farm unit, including the camera, Raspberry Pi, and 4G/5G modem

The Self-Hosted IoT Stack on the Local Server comprises several key components: (1) Eclipse Mosquitto (MQTT Broker) [37], [38], [39]: This acts as the central hub for receiving and distributing MQTT messages from all the OLIVE-CAD units in the farms, (2) PostgreSQL with TimescaleDB: This robust database system is used for storing the received IoT data. PostgreSQL provides a reliable relational database, while TimescaleDB, an extension for time-series data, optimizes the storage and querying of the continuous stream of diagnostic results from the farms[40], [41], [42], [43], and (3) Grafana for Data Viewing: Grafana is utilized as a powerful visualization tool, connected to the PostgreSQL database [44], [45]. It enables "Real-time / off-time data viewing," allowing users to visualize the diagnostic results instantly as they are received, or to review historical data, identify trends, and generate comprehensive reports for informed disease management strategies across all connected farms.

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

This integrated architecture ensures that the OLIVE-CAD system provides a complete solution for automated olive leaf disease diagnostics, from on-site image capture and AI-powered analysis to centralized data management and remote visualization

C. The OLIVE-CAD real-life testing

The real-life evaluation of the OLIVE-CAD system was conducted to assess its performance and practicality in simulated real-life agricultural settings. This phase specifically utilized the 10% "Testing" dataset (1132 samples), which was initially set aside to ensure an unbiased assessment of the integrated IoT system. For this evaluation, the 1132 samples from the evaluation dataset were logically divided into four equal parts, with each part conceptually allocated to one of the four distinct olive farms previously described (Siwa Oasis, Fayoum, Ismailia, and Sadat City). This "four farm model" serves as a simulation for a real-world deployment. To manage costs and logistical efforts associated with physical testing, a subset of 50 images from each farm's allocated evaluation data was physically printed from the dataset collected from the internet and then processed and analyzed using the deployed OLIVE-CAD units.

The image capture for this real-life evaluation was performed directly using the OLIVE-CAD GUI software, which was developed and deployed on the Raspberry Pi within each local OLIVE-CAD unit. The image capture for this real-life evaluation was performed directly using the OLIVE-CAD GUI software, which was developed and deployed on the Raspberry Pi within each local OLIVE-CAD unit (as detailed in Figure 8). This GUI facilitated the interaction for farmers or field technicians to place an olive leaf sample into the OLIVE-CAD Box, initiate image capture, and trigger the diagnostic process. Crucially, the trained YOLOv12 model was integrated directly into this Raspberry Pi-based processing module, enabling real-time inference on newly captured images. Figure 9 illustrates the comprehensive OLIVE-CAD system,

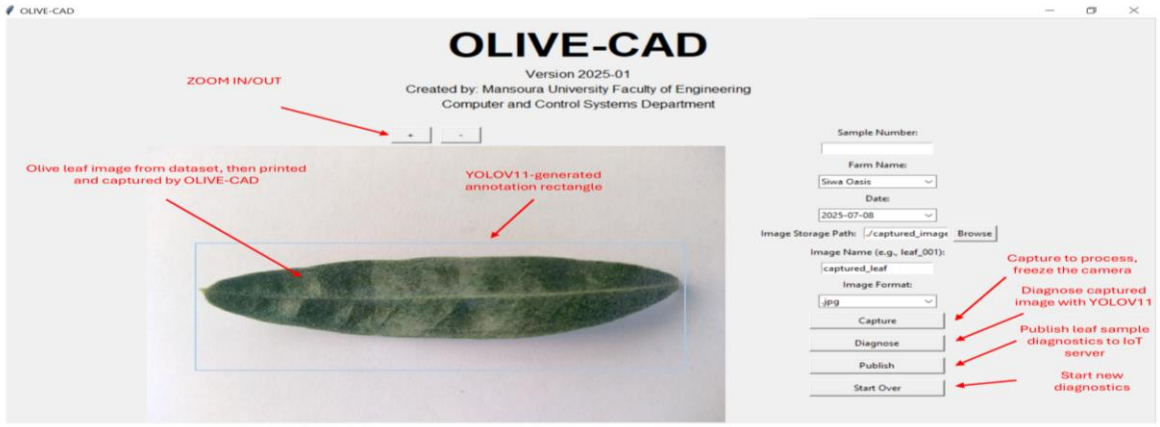


Figure 8. OLIVE-CAD Graphical User Interface, this figure displays the user interface of the OLIVE-CAD

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

showcasing its compact diagnostic box with its integrated front panel and sample entry. The figure further details the essential hardware components used in its development, the internal flashlight for optimal sample illumination, and the overall setup during sample capture and diagnosis. Additionally, the prepared printed samples, precisely positioned on sample entry holders, are shown, facilitating consistent image capture for diagnosis.

Upon processing each image, the OLIVE-CAD GUI provided immediate diagnostic results. These results, along with contextual information, were then transmitted to the central IoT server using the Paho-Mqtt client library. The information transmitted via MQTT messages included: (1) Farm Name: Identifying the specific farm (e.g., "Siwa Oasis Farm," "Fayoum Farm") from which the sample originated, (2) Sample Number: A unique identifier for each olive leaf sample analyzed, (3) and Diagnostic Results: The classification output from the integrated YOLOv12 model (e.g., "Healthy," "Aculus olearius," or "Peacock spot") along with the confidence score for the detection.

Figure 9. Detailed View of the OLIVE-CAD Diagnostic Box, Including Hardware Components, Internal Illumination, Operational Setup, and Prepared Sample Images

The entire data flow from image capture to data storage and visualization is comprehensively illustrated in Figure 10. the process initiates with Image Capture from the OLIVE-CAD unit, followed by Image Resize (224x224) and Image Preprocessing to prepare the visual data. The preprocessed image is then fed into the YOLOv12 model for disease diagnostics. The results from the YOLOv12 model, along with user inputs such as "Sample Name", "Farm Name", "Image storage path", and "Date" are then handled by the PAHO-MQTT client.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Before publishing, the system attempts to MQTT BROKER connect using predefined parameters (MQTT BROKER address, username, password, port, PostgreSQL host, database address, username, password, and OLIVE-CAD device ID). If the connection is unsuccessful, a "Not Connected to MQTT BROKER" message is displayed. Once connected, a data table (JSON) is created [46], [47], encapsulating the diagnostic information. This JSON data is then transmitted over the internet to the Mosquitto MQTT Broker. Upon reaching the broker, the data is channeled to TimescaleDB. A crucial check is performed: if a data table is not already created, a new one is generated. The diagnostic information then undergoes Data Import into the PostgreSQL database. Finally, the data is available for Data Export and Grafana data viewing, allowing for real-time and historical analysis of olive leaf health across all deployed farms.

3. Experimental results

The results are presented in two distinct sections to comprehensively document the system's performance. The initial section focuses on the deep learning component, detailing the training and rigorous evaluation of the YOLO12 model. This segment utilizes a standard set of evaluation metrics, including precision, recall, and mean average precision (mAP), to quantify the model's efficacy in object detection and classification for olive leaf diseases. The subsequent section is dedicated to the integration of this model within an IoT framework, demonstrating the practical application of the trained model for real-time analysis and providing actionable insights for agricultural management.

3.1. Results of YOLO Model Training and Evaluation

The model's performance was assessed by analyzing key training and validation metrics. Figure 11 illustrates the model's learning progression over a series of training epochs. The training loss curves, including bounding box loss, classification loss, and distribution focal loss, demonstrate a consistent and steep decline, indicating that the model was effectively minimizing error and learning the features of the dataset. Concurrently, the validation loss curves for the same metrics also showed a stable downward trend. This synchronicity between training and validation losses signifies that the model was not overfitting to the training data and was successfully generalizing its learned knowledge to unseen data. This is further substantiated by the steady increase in validation precision, recall, and mAP metrics throughout the training process.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

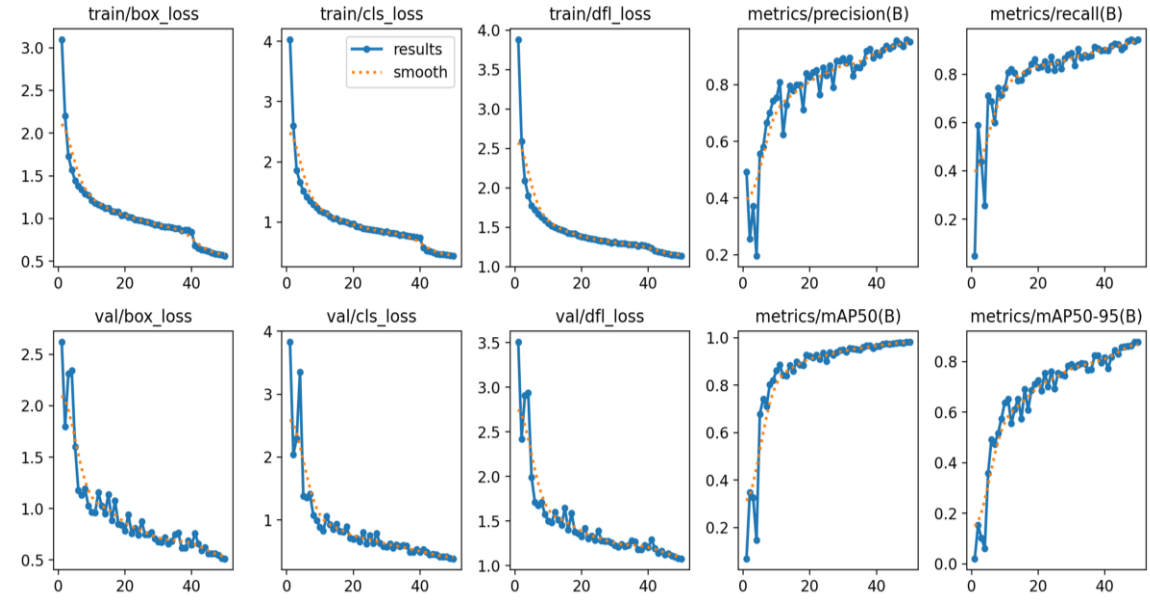


Figure 11. Training and Validation Metrics

A normalized confusion matrix, as depicted in Figure 12, was employed to visualize the model's classification performance at a class-specific level. The results reveal a high degree of accuracy, particularly along the diagonal, which represents correct classifications. The model achieved a high accuracy rate of 97% for the 'healthy' class and 95% for the 'scab' class. The 'aculus' class was correctly identified with an 84% accuracy rate. This analysis highlighted a primary area of misclassification between the 'aculus' and 'scab' classes, with 10% of actual 'aculus' leaves being predicted as 'scab'. This suggests that these two classes share some visual similarities that the model found challenging to distinguish consistently. However, overall inter-class confusion remained low, with negligible misclassifications with the 'background' class.

The model's performance was further evaluated through several key curves, each providing a different perspective on its behavior, as indicated in Figure 13. The Precision-Confidence curve demonstrates that the model maintains a high level of precision, approaching a value of 1.00 at a confidence threshold of 0.960. Conversely, the Recall-Confidence curve illustrates the trade-off between recall and prediction confidence, where recall remains high at lower confidence thresholds but decreases as the threshold increases. The F1-Confidence curve identifies the optimal balance between precision and recall, with the highest F1 score of 0.95 being achieved at a confidence threshold of 0.580. This point represents the ideal operational setting for the model to achieve a robust balance between minimizing false positives and false negatives. The Precision-Recall curve confirms the model's strong performance across all confidence thresholds. The large area under the curve

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

indicates a high average precision for each class, contributing to an overall mean average precision (mAP) of 0.982 at an IoU threshold of 0.5

Figure 12. Normalized Confusion Matrix

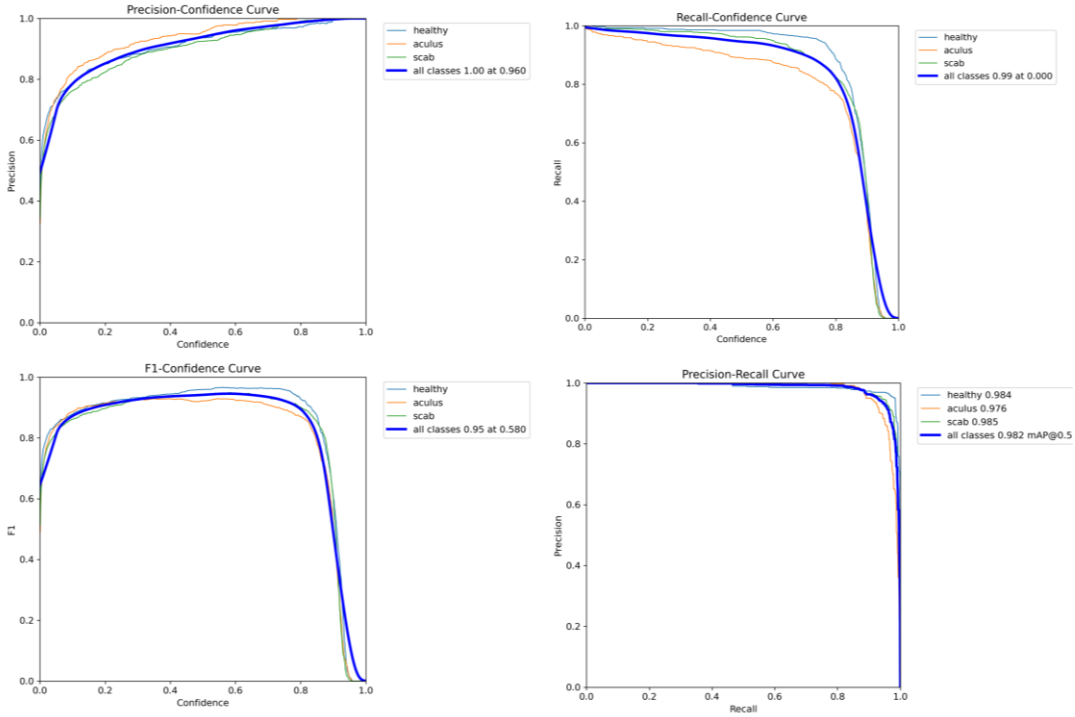


Figure 13. Precision-Confidence, Recall-Confidence, F1-Confidence, and Precision-Recall Curves

The training process was visually confirmed by examining training batches with augmented data. A representative training batch (Figure 14) shows several images with their corresponding ground truth bounding boxes and class IDs. The presence of rotated, scaled, and cropped images within the batch demonstrates the application of various data augmentation techniques.

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

This approach is crucial for enhancing the model's ability to generalize by exposing it to a broader range of visual variations, thus preventing overfitting and improving its robustness in diverse real-world conditions. The validation batch was visually inspected. The ground truth labels for the batch (Figure 15) show correctly annotated leaves belonging to the 'aculus' and 'scab' classes. A comparison with the model's predictions (Figure 16) reveals that the model successfully identified the majority of leaves with high confidence scores. This visual evidence aligns with the quantitative metrics, confirming that the model effectively detects and classifies olive leaf diseases in unseen images. The presence of a few minor misclassifications and bounding box inaccuracies in this batch provides a qualitative explanation for the small performance gaps observed in the confusion matrix and other evaluation curves.

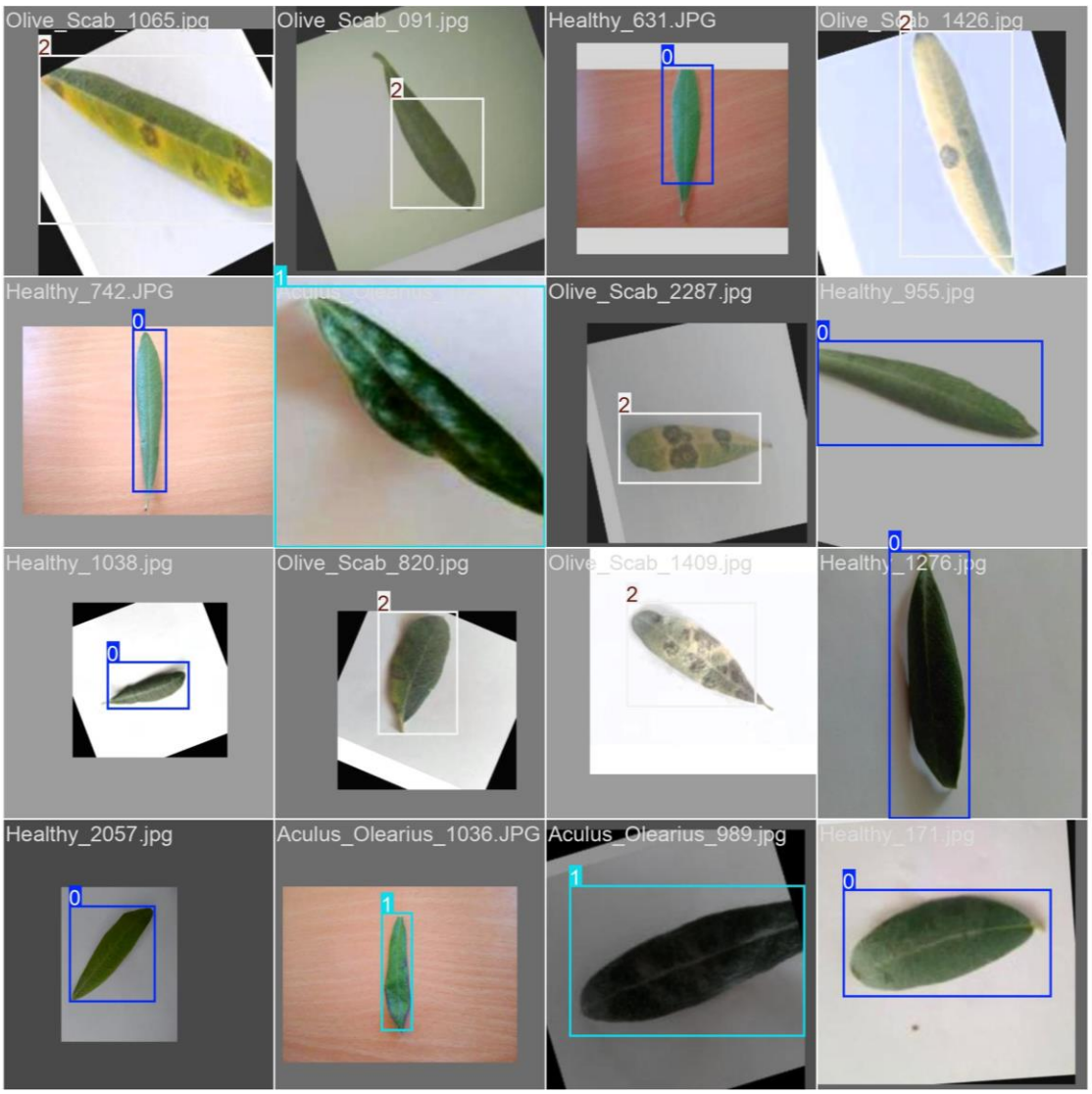


Figure 14. Training Batch

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Figure 15. Validation Batch Labels

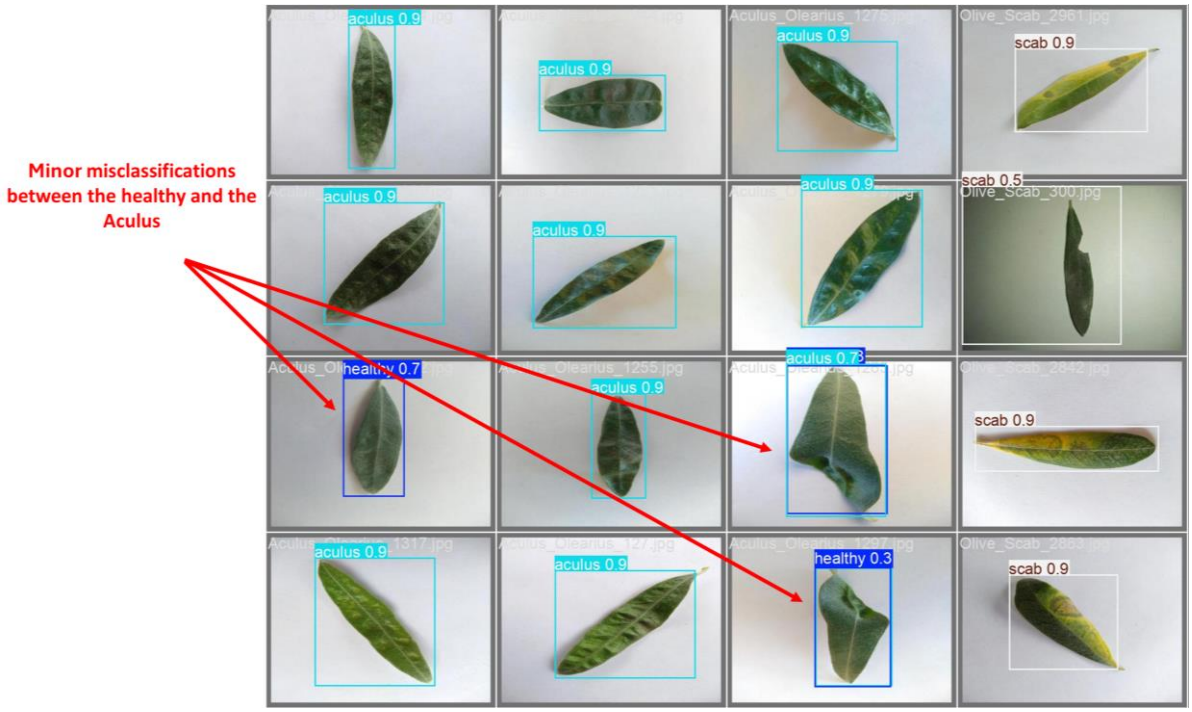


Figure 16. Validation Batch Predictions, the figure displays the model's predictions on a validation batch, showing bounding boxes, labels, and confidence scores.

3.2.Results of real-life application

For the real-life application experiment, a dedicated dataset was curated to test the system's performance in a controlled environment. The dataset consisted of 60 individual olive leaf samples, with 20 samples for each class: 'healthy,' 'aculus,' and 'scab', as shown in Figure 17.

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Given the limitation of not having access to physical olive leaves, images of the leaves were sourced from the internet. These images were then printed on high-quality photo paper and placed in a specialized image holder. This holder was then inserted into the custom-built OLIVE-CAD examination box, which housed the camera-AI module and an internal illumination system. Images were captured by the camera, diagnosed by the developed YOLOv12 model, and then published to the IoT server using a custom-developed Python script.

The classification results from this experiment, shown in Figure 18, provide a practical validation of the model's performance. The model successfully classified all 20 of the healthy samples with 100% accuracy. However, instances of misclassification were observed, which align with the trends identified in the confusion matrix (Figure 12). Specifically, some samples were misclassified, and in a few cases, a single leaf was classified as containing multiple diseases simultaneously. This behavior is consistent with the confusion matrix, which shows a degree of inter-class confusion, particularly between 'aculus' and 'scab,' as well as with the 'background' class. These results highlight a key limitation: while the model performs well on a majority of cases, it can struggle with samples that exhibit characteristics of multiple classes or are presented in a way that differs from the training data. This suggests that further refinement is needed to enhance the model's ability to differentiate between these visually similar diseases and reduce the occurrence of multi-label classifications for a single sample.

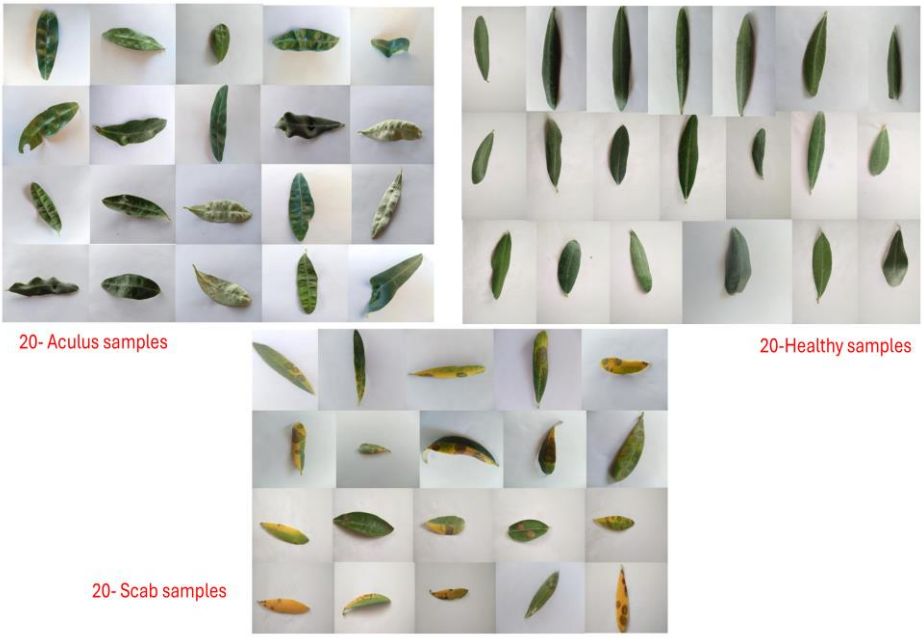


Figure 17. Olive Leaf Samples for Real-Life Application Experiment

The communication protocol between the OLIVE-CAD client and the IoT server is facilitated by a robust data serialization and transmission process, as illustrated in the system's data flow diagram (Figure 19). Following the YOLOv12 model's classification of an image, the client-side script extracts the diagnostic results and other relevant metadata. This data is then serialized into a structured JSON payload, which serves as a lightweight and efficient format for

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

communication. The detailed structure of this JSON packet is defined by several key-value pairs (Figure 20). It includes identifiers for the Sample Name and Farm Name, a precise Time stamp for the observation, and the Diagnostic Result from the model, which is represented by a numerical code (0 for healthy, 1 for aculus, and 2 for scab). A crucial component of this payload is the Leaf Image, which is a base64-encoded string of the classified and annotated image. This encoding allows the image to be transmitted as part of the JSON packet, ensuring that both the diagnostic result and visual evidence are sent together. The client then utilizes a PAHO-MQTT connection to publish this JSON payload to the IoT server, where it is stored for further analysis and visualization.

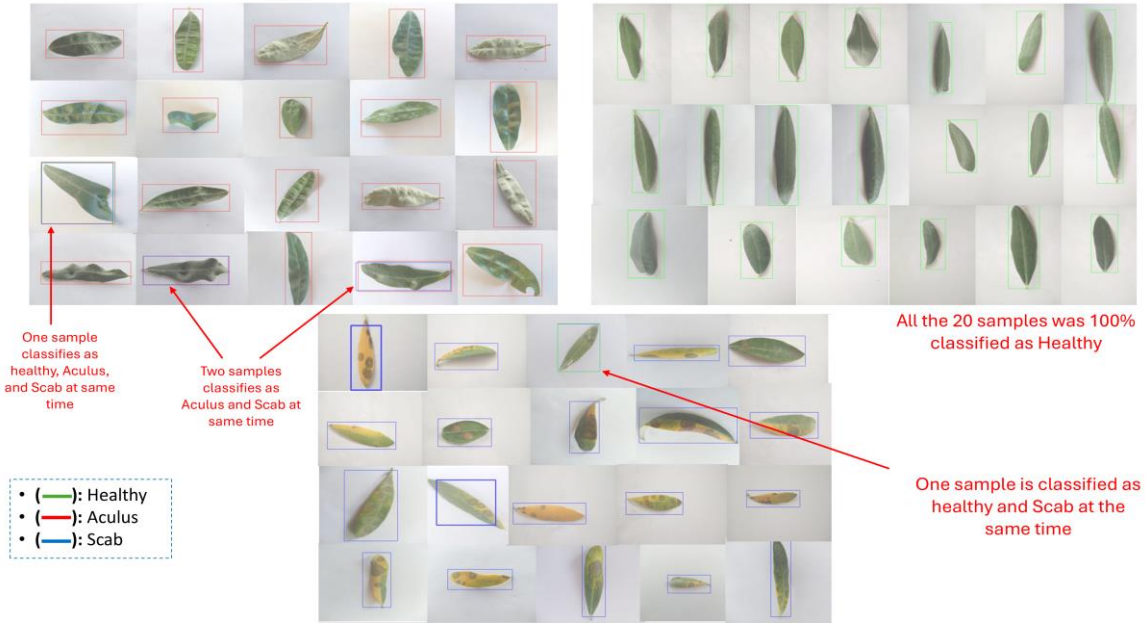
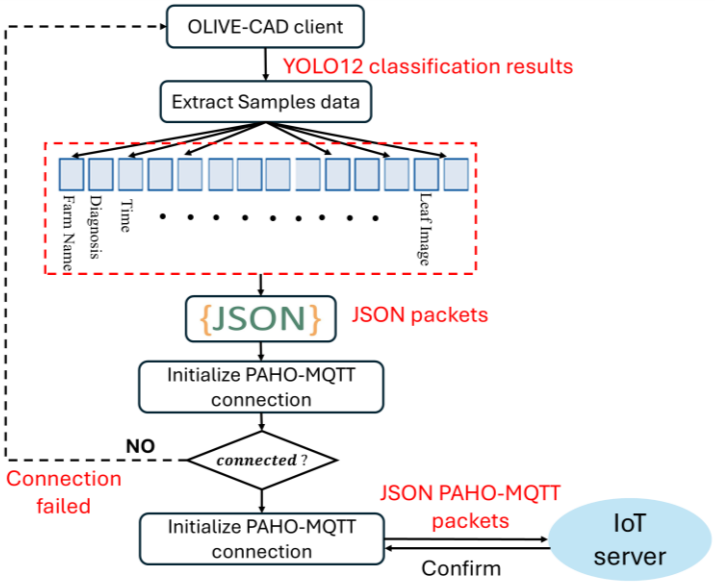


Figure 18. Real-life Application Experiment Results

Figure 19. IoT Data Publication Flow Diagram



International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

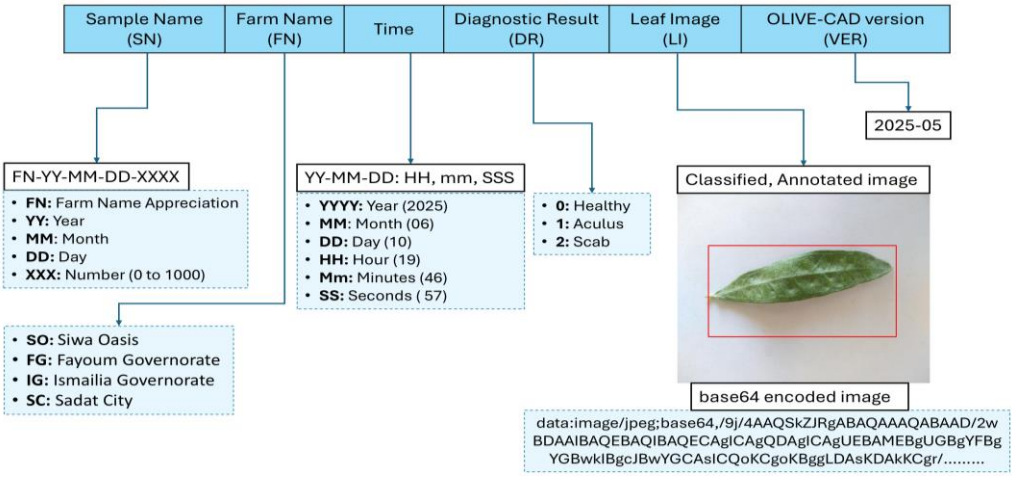


Figure 20. JSON Data Structure of the IoT Client

The final stage of the IoT pipeline involves storing the diagnostic data on a central server for analysis and visualization. The system uses a PostgreSQL database, integrated with TimescaleDB, to store all incoming JSON payloads from the client. This allows for efficient management of time-series data. As shown in Figure 21, a Grafana dashboard is used to visualize the stored information. The dashboard provides a user-friendly interface where specific farms can be selected, and the stored diagnostic data for each olive leaf sample can be viewed. The user can click on a specific data entry to display the corresponding annotated image, sample number, and diagnostic result. This visualization provides farmers with real-time, actionable insights, allowing them to monitor the health of their olive leaves and track disease trends over time. This functionality transforms the raw model predictions into a valuable tool for agricultural management.

Figure 21. Grafana Dashboard for the OLIVE-CAD IoT Server

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

4. Conclusions and Future Work

This paper introduced OLIVE-CAD; a novel IoT-enabled system designed for the real-time, on-site diagnosis of olive leaf diseases. Addressing a critical gap in agricultural technology, the system provides a complete, end-to-end solution that integrates image acquisition, deep learning-based diagnostics, and cloud-based data management. The system's architecture comprises a high-resolution camera for image capture, a Raspberry Pi microprocessor for on-site image processing, and an IoT module for remote data transmission via MQTT to a cloud platform. At the core of OLIVE-CAD is a custom-trained YOLOv12 convolutional neural network, which was developed using a meticulously prepared dataset of 11,315 olive leaf images. The model achieved high performance metrics (Figure 22), including a mean average precision (mAP@0.5) of 0.982, with class-specific accuracies of 97% for 'healthy', 84% for 'aculus', and 95% for 'scab'. The system's functionality was validated through a comprehensive real-life experiment involving 60 printed olive leaf samples, which were sourced from four distinct farms to simulate different real-world scenarios. This validation successfully demonstrated the seamless end-to-end flow of data: from image capture and on-site diagnosis by the embedded YOLOv12 model to its final storage in a PostgreSQL database with TimescaleDB, and its subsequent visualization on a Grafana dashboard.

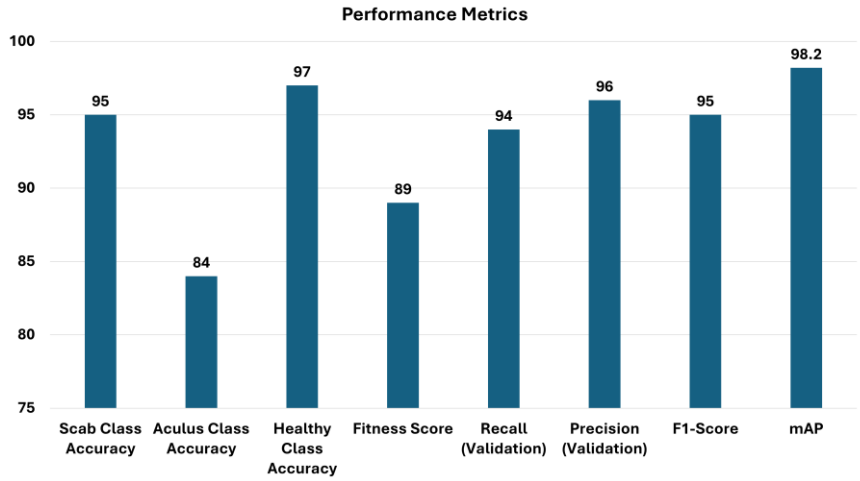


Figure 22. Performance Metrics of the YOLOv12 Model

The successful development and validation of the OLIVE-CAD system represent a significant step toward the practical deployment of deep learning for precision agriculture. By combining a high-performance YOLOv12 model with a robust IoT pipeline, this research demonstrates a feasible and scalable approach to automated disease detection. The system's ability to provide real-time diagnostics and transmit structured data offers tangible benefits for farmers, enabling early disease detection, proactive intervention, and data-driven insights into crop health over time. While the model showed a high overall accuracy, a key limitation was the observed

Received: August 07, 2025

International Journal of Applied Mathematics
Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

misclassification in some cases. As shown in Figure 23, these errors can be attributed to several factors. The analysis of the confusion matrix revealed a degree of confusion not only between visually similar disease classes, particularly 'aculus' and 'scab', but also with the 'background' class. This explains why some leaves were incorrectly classified or why the model occasionally identified defects in the photography and printing of the real-life samples as disease characteristics. For instance, the visual similarity between the natural curves of an aculus leaf and the spots of a scab infection was a common source of misclassification. This work also highlights the limitations of alternative diagnostic methods. Common recommendations for practical deployment include drone integration for direct farm sampling or the development of mobile applications. However, these approaches present inherent limitations. Drone deployment may face security constraints and the need for specialized expertise, while mobile applications could be hindered by unreliable internet connectivity for data transmission to IoT platforms or centralized data collection centers. In contrast, the fixed, custom-built nature of the OLIVE-CAD system provides a controlled environment that mitigates these variables, ensuring consistent image quality and more reliable diagnostics. This work establishes a strong foundation for future innovations in agricultural technology, paving the way for more efficient and sustainable farming practices.

Building upon the successful foundation of the OLIVE-CAD system, several avenues for future research and development can be pursued to enhance its capabilities and impact. To address the model's misclassification issues, future work will focus on dataset enhancement and quality control. This includes collecting a larger variety of images from real-life field environments, ensuring consistent lighting and photographic quality during image capture, and conducting an even more rigorous manual review to eliminate any incorrect annotations. This will improve the model's ability to generalize and distinguish between visually similar classes. Furthermore, the full potential of the system can be realized by leveraging the collected data for predictive analytics. Future efforts will involve integrating advanced AI models on the IoT server to ana

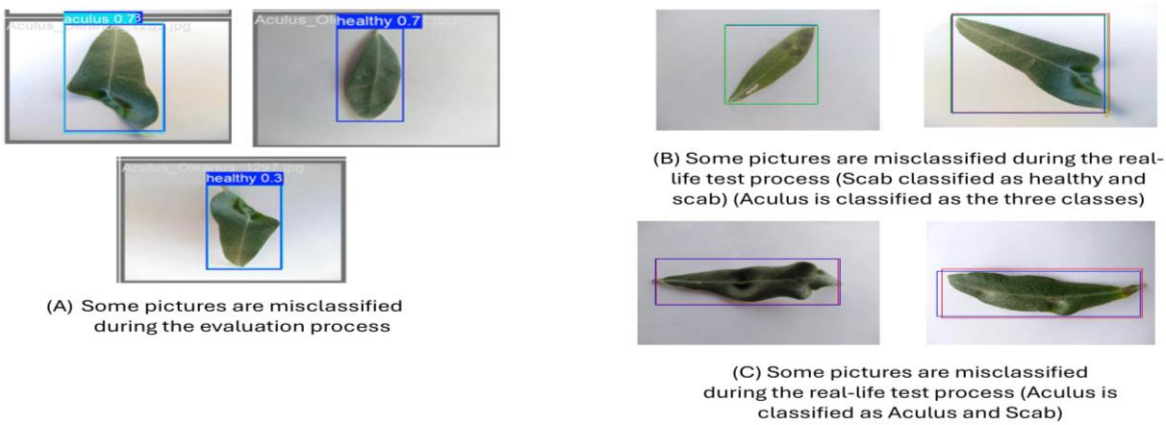


Figure 23: Misclassified Images during Evaluation and Real-life Tests

Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

lyze historical data trends. This will allow for the development of predictive algorithms that can forecast disease outbreaks and recommend proactive treatment strategies, moving the system from reactive diagnosis to predictive management. To maximize its practical utility, the OLIVE-CAD system could be integrated with existing agricultural management software, which would allow for the automated triggering of specific actions, such as dispatching alerts to farm managers or even controlling automated irrigation or pesticide spraying systems based on the diagnostic results.

References

[1] M. A. Mahmood and K. Alsalem, "Olive Leaf Disease Detection via Wavelet Transform and Feature Fusion of Pre-Trained Deep Learning Models.," *Computers, Materials & Continua*, vol. 78, no. 3, 2024.

[2] S. A. Patil, A. Bhosale, and A. Patil, "Advancements in plant leaf disease detection: a comprehensive review of latest trends and technologies," in *2024 3rd International Conference for Innovation in Technology (INOCON)*, IEEE, 2024, pp. 1–5.

[3] E. Jain and A. Singh, "Olive Leaf Disease Classification Using the Xception Model: A Deep Learning Approach," in *2024 International Conference on Cybernation and Computation (CYBERCOM)*, IEEE, 2024, pp. 715–719.

[4] Q. Bsoul and M. Jawarneh, "Olive Leaf Disease Detection using Improvised Machine Learning Techniques," *Semarak International Journal of Machine Learning*, vol. 5, no. 1, pp. 64–73, 2025.

[5] M. R. Kareem, "Image analysis and detection of olive leaf diseases using recurrent neural networks," *Al-Mustansiriyah Journal of Science*, vol. 35, no. 1, pp. 60–65, 2024.

[6] K. U. Majikumna, M. Zineddine, and A. E. H. Alaoui, "FLVAEGWO-CNN: Grey Wolf Optimisation-Based CNN for Classification of Olive Leaf Disease via Focal Loss Variational Autoencoder," *Journal of Phytopathology*, vol. 172, no. 6, p. e13438, 2024.

[7] A. Diker *et al.*, "An effective feature extraction method for olive peacock eye leaf disease classification," *European Food Research and Technology*, vol. 250, no. 1, pp. 287–299, 2024.

[8] S. Huaquipaco *et al.*, "Peacock spot detection in olive leaves using self supervised learning in an assembly meta-architecture," *IEEE Access*, 2024.

[9] I. F. Kallel, M. Kallel, K. Kammoun, M. Salhi, and M. A. Triki, "Deep learning models for automatic detection and classification of pathogens in olive leaves," in *2024 4th*

Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET), IEEE, 2024, pp. 1–6.

[10] M. Dammak, A. Makhloufi, B. Louati, and A. Kallel, "Detection and Classification of Olive Leaves Diseases Using Machine Learning Algorithms," in *International Conference on Computational Collective Intelligence*, Springer, 2024, pp. 292–304.

[11] E. Guermazi, A. Mdhaffar, M. Jmaiel, and B. Freisleben, "LIDL4Oliv: A Lightweight Incremental Deep Learning Model for Classifying Olive Diseases in Images.," in *ICAART (2)*, 2024, pp. 583–594.

[12] E. Roma, P. Catania, M. Vallone, and S. Orlando, "Assessing the effectiveness of pruning in an olive orchard using a drone and a multispectral camera: a three-year study," *Agronomy*, vol. 14, no. 5, p. 1023, 2024.

[13] M. A. Mahmood and K. Alsalem, "Olive Leaf Disease Detection via Wavelet Transform and Feature Fusion of Pre-Trained Deep Learning Models.," *Computers, Materials & Continua*, vol. 78, no. 3, 2024.

[14] "Olive Leaf Image Dataset." Accessed: Aug. 12, 2025. [Online]. Available: https://www.kaggle.com/datasets/habibulbasher01644/olive-leaf-image-dataset

[15] "Olive Leaf Disease Detection." Accessed: Aug. 12, 2025. [Online]. Available: https://www.kaggle.com/code/mahmoudlimam/olive-leaf-disease-detection/input

[16] "Olive Leaf Disease." Accessed: Aug. 12, 2025. [Online]. Available: https://www.kaggle.com/datasets/serhathoca/zeytin

[17] M. Bagga and S. Goyal, "A Comparative Study of the Deep Learning Based Image Segmentation Techniques for Fruit Disease Detection," *Reviews in Agricultural Science*, vol. 13, no. 1, pp. 81–104, 2025.

[18] H. Hamzaoui *et al.*, "Assessment of Peacock Spot Disease (Fusicladium oleagineum) in Olive Orchards Through Agronomic Approaches and UAV-Based Multispectral Imaging," *Horticulturae*, vol. 11, no. 1, p. 46, 2025.

[19] B. Lee, H. Lee, X. Sun, U. Ali, and E. Park, "Deblurring 3d gaussian splatting," in *European Conference on Computer Vision*, Springer, 2024, pp. 127–143.

[20] C. Wu, D. Wang, and K. Huang, "Enhancement of mine images based on hsv color space," *IEEE Access*, vol. 12, pp. 72170–72186, 2024.

[21] J. Bayón, J. Recas, and M. Guijarro, "An optimized, color-adaptive blue light filtering approach using a novel color space transformation," *Displays*, vol. 90, p. 103124, 2025.

Received: August 07, 2025

Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[22] R. Sapkota *et al.*, "Yolov12 to its genesis: A decadal and comprehensive review of the you only look once (yolo) series," *arXiv preprint arXiv:2406.19407*, 2024.

[23] N. Jegham, C. Y. Koh, M. Abdelatti, and A. Hendawi, "Yolo evolution: A comprehensive benchmark and architectural review of yolov12, yolo11, and their previous versions," *arXiv preprint arXiv:2411.00201*, 2024.

[24] R. Sapkota, Z. Meng, M. Churuvija, X. Du, Z. Ma, and M. Karkee, "Comprehensive performance evaluation of yolov12, yolo11, yolov10, yolov9 and yolov8 on detecting and counting fruitlet in complex orchard environments," *arXiv preprint arXiv:2407.12040*, 2024.

[25] N. Jegham, C. Y. Koh, M. Abdelatti, and A. Hendawi, "Yolo evolution: A comprehensive benchmark and architectural review of yolov12, yolo11, and their previous versions," *arXiv preprint arXiv:2411.00201*, 2024.

[26] B. H. Saad, M. M. H. Morsi, and S. Sherif, "Farmers' awareness, attitude, and role towards climate change: case of olive production in Egypt," *Journal of the Advances in Agricultural Researches*, vol. 29, no. 2, pp. 263–278, 2024.

[27] S. S. ElGarhy and S. Ghenmy, "AN ECONOMIC STUDY OF OLIVE PRODUCTION AND ESTIMATING THE IMPACT OF PRODUCTION AND MARKETING PROBLEMS ON THE ECONOMICS OF ITS PRODUCTION IN NORTH SINAI GOVERNORATE," *Sinai Journal of Applied Sciences*, vol. 14, no. 1, pp. 29–54, 2025.

[28] A. M. Taha and H. E.-H. Khalifa, "Olive in Egypt: Cultural Practices," *Olives and Olive Related Products-Innovations in Production and Processing: Innovations in Production and Processing*, vol. 35, 2025.

[29] P. Marques, L. Pádua, J. J. Sousa, and A. Fernandes-Silva, "Advancements in remote sensing imagery applications for precision management in olive growing: A systematic review," *Remote Sens (Basel)*, vol. 16, no. 8, p. 1324, 2024.

[30] A. Joice *et al.*, "Applications of Raspberry Pi for Precision Agriculture—A Systematic Review," *Agriculture*, vol. 15, no. 3, p. 227, 2025.

[31] A. ElSanhoury, I. Galal, K. AlKady, A. ElKhodary, D.-T. Phan-Huy, and A. M. Hassan, "First Real-Time Detection of Ambient Backscatters using Uplink Sounding Reference Signals of a Commercial 4G Smartphone," *arXiv preprint arXiv:2501.15576*, 2025.

[32] D. Mojaravscki and P. S. Graziano Magalhães, "Comparative evaluation of color correction as image preprocessing for olive identification under natural light using cell phones," *AgriEngineering*, vol. 6, no. 1, pp. 155–170, 2024.

Received: August 07, 2025

Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[33] A. Liopa-Tsakalidi, V. Thomopoulos, P. Barouchas, A. D. Boursianis, and S. K. Goudos, "A LoRaWAN-based IoT platform for smart irrigation in olive groves," *Smart Agricultural Technology*, vol. 9, p. 100673, 2024.

[34] A. Morchid, R. Jebabra, H. Qjidaa, R. El Alami, and M. O. Jamil, "Agri-tech innovations for sustainability: a fire detection system based on MQTT broker and IoT to improve environmental risk management," *Results in Engineering*, vol. 24, p. 103683, 2024.

[35] A. Liopa-Tsakalidi, V. Thomopoulos, P. Barouchas, A. D. Boursianis, and S. K. Goudos, "A LoRaWAN-based IoT platform for smart irrigation in olive groves," *Smart Agricultural Technology*, vol. 9, p. 100673, 2024.

[36] V. Vitaskos, K. Demestichas, S. Karetsos, and C. Costopoulou, "Blockchain and internet of things technologies for food traceability in olive oil supply chains," *Sensors*, vol. 24, no. 24, p. 8189, 2024.

[37] M. N. K. Boutros, A. Meroth, N. Sharaf, and Y. A. Zaghloul, "Development of a MQTT-Based Server Software 'Agriculture Precision,'" in *2024 22nd International Conference on Research and Education in Mechatronics (REM)*, IEEE, 2024, pp. 172–177.

[38] M. Has, D. Kreković, M. Kušek, and I. Podnar Žarko, "Efficient data management in agricultural IoT: Compression, security, and MQTT protocol analysis," *Sensors*, vol. 24, no. 11, p. 3517, 2024.

[39] A. Hermansyah, H. Ubaya, A. A. Aljali, N. Afifah, and S. Kusuma, "Quality of Service (QoS) Analysis of MQTT Protocol for Smart Farming Monitoring System Optimization," 2025.

[40] M. Correia, J. Alves, R. Sequeira, and J. Exposto, "Development of an Integrated Portal for Farm Audit Management with Geospatial Support".

[41] C. F. D. S. Rosette and M. S. F. S. Kelly, "AgroNLP–Aplicación web para consultas en lenguaje natural (NLP) sobre bases de datos agrícolas usando Deep Seek api," *Revista Cubana de Geomática*, vol. 7, no. 1, 2025.

[42] K. Panduru and J. Walsh, "Smart agriculture: software platform for telematics monitoring in farm machinery," in *2024 35th Irish Signals and Systems Conference (ISSC)*, IEEE, 2024, pp. 1–6.

[43] M. Correia, J. Alves, R. Sequeira, and J. Exposto, "Development of an Integrated Portal for Farm Audit Management with Geospatial Support".

[44] X. Wang, Q. Gao, Y. Tao, F. Marinello, and Q. Huang, "A DF-LCA Modeling and Evaluation Case Study on an Agricultural Pests and Diseases Knowledge Graph Based

Volume 38 No. 12s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Q&A System," in *2024 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*, IEEE, 2024, pp. 609–614.

[45] L. Antonelli, H. Badir, H. Bazza, S. Bimonte, and S. Rizzi, "Requirements engineering for continuous queries on IoRT data: A case study in agricultural autonomous robots monitoring," in *Proceedings of the 26th International Conference on Enterprise Information Systems (Volume 2)*, SciTePress, 2024, pp. 113–120.

[46] M. S. Basir, Y. Zhang, D. Buckmaster, A. Raturi, and J. V Krogmeier, "Meta Ag: An Automatic Agricultural Contextual Metadata Collection App," *Smart Agricultural Technology*, p. 101073, 2025.

[47] I. Garg, N. R. Wudaru, and P. Ramakrishna, "Study on JSON, its uses and applications in engineering organizations," *ResearchGate, March*, 2024.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

Received: August 07, 2025