International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(7s) 2026 pp. 1262-1283
https://doi.org/10.70917/ijcisim-2026-3195
Article

Edge–Cloud Collaborative Vision Framework for Multi-Class Lemon Leaf Disease Detection Using YOLOv8 and EfficientNet-B3

Mundla Rama Mohan Reddy[1], S. K. Mahaboob Basha[2]

[1]Department of Computer Science, Bharatiya Engineering Science & Technology Innovation University, Gorantla, Andhra Pradesh, India.
Email: mundlaramamohanreddy.cs@gmail.com
[2]Department of Computer Science and Engineering, Sree Dattha Institute of Engineering and Science, Hyderabad, Telangana, India.
Email: email2mahaboob@gmail.com

Abstract: The field conditions of citrus leaf disease diagnostics have not been correctly and real-time diagnosed because of complex backgrounds, variability in illumination, and overlapping foliage. The paper introduces an edge-cloud collaborative vision model of the multi-class lemon leaf disease detection system based on a real-time object detection approach and high-accuracy classification. The system suggested is implemented to deploy YOLOv8-Tiny to the edge to detect lemon leaves and extract the region-of-interest and classify the disease with a fine-tuned EfficientNet-B3 in the cloud. The two stage design eliminates background interference that is characteristic in single stage classification methods used on raw field images.
A self-made collection of lemon leaf pictures in varying climatic and lighting situations has been gathered. The diffusion-based generative models were applied to perform the generalization and reduce the imbalance between classes by generating synthetic samples to enrich the training data. The detection model was trained to be deployed on the edge, whereas the classification model was made into a TensorFlow Lite model and optimized through post-training quantization.
Experimental results indicate that the suggested framework can establish an excellent detection performance with a median Average Precision (mAP50) of about 95% and multi-class classification of disease with an accuracy of about 97%, and with an end-to-end inference latency of about Windows, which is low enough to support real-time agricultural monitoring. Field-level validation is an indication that the system is consistent in its performance in real-life conditions. The paper presents the edge-cloud architecture as a scalable and energy-efficient design to citrus disease surveillance in agriculture that is limited in resources.
Keywords: Edge–Cloud Collaborative Computing; Lemon Leaf Disease Detection; YOLOv8-Tiny; EfficientNet-B3; Diffusion-Based Data Augmentation; Precision Agriculture; Real-Time Inference; Computer Vision in Agriculture

1. Introduction

The citrus crops and more so lemon are also part of the horticultural economies as they are highly commercially viable and there is no lack of demand. Production of lemon is however highly affected by foliar infections such as citrus canker, greening, melanose, scab and leaf miner infestation. These ailments reduce the quality of yielding, increase production costs and most of the times they are only discovered when it is too late. Traditional method of disease diagnosis is the manual one in which it is subjective, time consuming and labor intensive and is practically inapplicable to large plantations. This provides a high demand of computerized, accurate and generalizable disease recognition systems that can be used in agricultural context.

The latest advances in computer vision and deep learning depicted a high potential of identifying plant diseases using convolutional neural networks (CNNs). A great deal of research has shown high rates of classification accuracy in controlled laboratory settings using single stage image classification models. However, the techniques are not quite efficient in the field where pictures can have difficult backgrounds, a variety of intersecting leaves, various lighting, shadows, and obscurations as well as noise. Recurrent utilization of the classifier on the raw field images leads to contamination of the background as well as catastrophically impairing the performance and thus making them inapplicable in the real-time scenario.

In a bid to address these weaknesses object detection pipelines have been taken into consideration whereby diseased areas are localized and then the classification step is taken. Faster R-CNN, SSD and the variants of YOLO have also been regarded as the models in the detection of leaves and disease. Among them, YOLO-based models are particularly suitable when it comes to real time implementation due to single step detection and reduced inference time. The high scale application of heavy detection and classification model on edge nodes is however costly in terms of computation and energy-intensive especially in the field of agriculture where resources are scarcely found.

Edge computing has been regarded as one of the potential remedies to the minimization of latency and bandwidth consumption through making preliminary inferences close to a piece of data. However, edge devices such as Raspberry Pi lack the processing units to run large capacity classification networks with significant accuracy compromises. On the other hand, with cloud computing, there is scalability of resource in terms of complex inferences, although this is associated with the problems of latency on the communications and dependence on network connections. Neither edge nor cloud computation is therefore ideal in real time monitoring of agriculture using one or the other.

The proposed article proposes an edge cloud collaborative vision model which offers a strategic partitioning of the inference workload between the edge and the cloud to strike a balance between the speed, accuracy and the energy efficiency. YOLOv8-Tiny will be implemented in the proposed system on the edge to identify leaves and the region of-interest (ROI) in the raw field images. The system then eliminates the necessary background areas at an early stage of the pipeline, so that the downstream classification becomes very reliable. The ROIs extracted are sent to the cloud, and an EffectiveNet-B3 model (finetuned) is used to classify the diseases with high accuracy in multi-classes. This is a two-level structure that is a pure reaction to the shortcomings of one-model and one-platform design.

Another eminent challenge of agricultural vision system is data scarcity and class imbalance. The actual disease data are difficult to be collected in large scale and limited sample of rare disease type. To solve this issue, the presented work will be based on diffusion-based generative models to create natural lemon leaf disease images, improving the diversity and the power of the datasets. Unlike the conventional augmentation approaches, diffusion models generate high-quality samples, which preserve disease-specific visual representations and therefore are more generalizable when applied to an unfamiliar field.

Unlike latest YOLOv8-based systems of citrus disease recognition where most recent systems utilize either single-stage instance detectors or edge-only detection pipelines, the proposed system employs an edge-based collaborative architecture of modules subdivided into the edge and cloud module. Real-time edge-level localizing of the leaf, cloud-based high-precision classification, diffusion-based data augmentation, and on real-field image validation are useful in making this work complete and practical in comparison to the approach of YOLOv8 which has been proposed in the existing literature.

The most engaging works of this work comprise the following:

The proposed study will entail the following.

The two models are YOLOv8-Tiny and EfficientNet-B3, which served well in locating all the leaves and classifying the disease into various categories, respectively.

- Synthetic data augmentation through diffusion to get over both imbalance between classes and environmental variance.
- Inference and running of the software on low-power edge computers on the cloud.

The reason is that it has been found to be highly accurate and low in end-to-end latency with much experimental validation of high accuracy up to real-field images.

This paper, by incorporating object detection, transfer learning, generative data augmentation and hybrid deployment, presents a feasible and scalable solution to smart citrus disease monitoring, which has presented a viable solution to the gap between laboratory studies and actual agricultural application.

2. Related Work

The detection of plant diseases with computer vision has become the subject of large-scale research in the last decade due to the development of deep learning and the growing number of agricultural image data. Initial methods used handcrafted characteristics like colour histograms, texture descriptors, and shape-based metrics as well as conventional classifiers such as the Support Vector Machines (SVM), the k-Nearest Neighbors (k-NN) and the Random Forests. Though these methods performed moderately well in controlled situations, they still performed poorly in real world field conditions because of their sensitivity to changes in illumination, background clutter and intra-class variation. This makes these methods no longer viable in large scale agricultural applications.

2.1 CNN-Based Plant Disease Classification

The convolutional neural networks (CNNs) was introduced and has altered tremendously the sphere of the plant disease recognition. The architectures that have been employed in a number of works to identify the disease of the leaves have included AlexNet, VGGNet, GoogLeNet and ResNet with the utilization of datasets such as plantvillage. These models have been reported to experience classification accuracy of over 95 percent in the laboratory. However, most of these works used prepared and tested models on the images which were homogenous in the background and not reflective on real farming conditions on a sample of one leaf. These classifiers showed high drop in accuracy when applied to any field image with many leaves, soil and branches with varied light conditions meaning the low ability of pure classification-based pipelines to generalize.

Transfer learning was popularized to reduce the amount of time taken during training and performance improvement on small agricultural datasets. In order to trade-off between accuracy and computational cost lightweight architectures such as MobileNet, ShuffleNet or EfficientNet variants were explored. Even though EfficientNet-based models performed very well in the scaling with the compound, their physical implementation on the edge devices often required either extreme quantizing or architectural reduction which led to a trade-off between performance and classification accuracy.

2.2 It is also possible to detect the disease on an object through the application of object detection method

In an attempt to avoid the disadvantages of having global image classification, researchers began to apply object detection techniques, in addition to finding disease in regions and subsequently classifying them. Detection models were applied, faster R-CNN and SSD, to determine the sites of the infected leaf area, and the analysis of the areas could be narrowed down further. However, two stage detectors were computationally expensive and would not operate on low-power hardware in real-time.

The models of the YOLO type became the most popular due to the possibility to detect the objects simultaneously and operate in real-time. Crop diseases and leaf segmentations have been detected using YOLOv3, YOLOv4, YOLOv5, and YOLOv7. These experiments were stronger in crowded set up isolating leaves against the background. Many implementations however attempted to detect and classify the disease in the same model of detection which decreased the granularity of the classification and scale, especially when there is a large number of disease classes.

2.3 Agricultural Vision in Edge computing.

Edges computing has been applied in smart farming to reduce latency, network dependency of the system, and cloud processing expenses. Some publications have applied lightweight CNNs to embedded systems, such as Raspberry Pi, Jetson Nano and Coral Edge TPU to detect-in-time diseases of plants. Despite the low latency of these systems, a small memory and processing potential constrained the system which served to maintain the model complexity low and to decrease the accuracy.

Hybrid edge-cloud architectures are proposed to compromise between quality of inference and computational efficiency. The first stage operations, such as image processing or object localization, are executed at the edge, and computationally-heavy classification or analytics are executed in the cloud in this form of systems. Even though they have potential, the existing edge-cloud agricultural systems lack systematic model partitioning strategy and are not typically experimented under real-field conditions, which constrains their application.

2.4 Generative models and Synthetic Data Augmentation.

Agricultural machine learning continues to face the challenge of the scarcity of data and class imbalance. Normal augmentation techniques such as rotation, flipping, scaling and color jittering have been actively used which do not offer much variation. Generative adversarial networks (GANs) have been studied in more recent literature to produce images of plant disease, and more images are being classified. However, GANs have a tendency to collapse modes and do not train effectively particularly on small data.

Diffusion-based generative models have emerged as a more reliable rival and are capable of producing superior and diverse synthetic images. Although diffusion models already have already shown promising results in the medical and industrial imaging sector, the scope of utilizing them in agricultural disease datasets is relatively limited. Literature applying synthetic data is not frequently tested in real fields, or integrates generation methodologies into an architecture of end-to-end implementation.

2.5 Research Gap and Motivation

Based on the analyzed literature, three gaps are developed:

Most of the plant disease detection systems rely on single stage classification models that are not applicable in the cluttered field conditions.

Edge only deployment is compromised in terms of accuracy and cloud only solutions cause latency and connection issues.

There has been very little research on diffusion-based data augmentation besides real-time edge-cloud inference.

It is in this body of work that this fissure is bridged by the suggestion of a two-time edge-cloud collaborative model that incorporates real-time YOLOv8-based leaf recognition and high-quality EfficientNet-B3 classification and diffusion-based dataset augmentation and tested under real field conditions.

3. System Architecture

This section will provide the general design of the proposed edge-cloud collaborative vision system in multi-class lemon leaf disease detection. The design is motivated by three inadmissible requirements, namely: (i) real-time operation in the field, (ii) high diagnostic accuracy and (iii) implementation on low-power agricultural devices. Any architecture that does not take into consideration one of these is an academic beauty that is practically useless. The proposed system

explicitly does not use monolithic deployment, but rather uses a task-partitioned pipeline which partitions workload according to both latency sensitivity and resource demands.

3.1 Architectural Overview

The designed framework is based on two-step inference pipeline that includes edge-level and cloud-level inference. Field images by default are first processed at the edge device in order to localize lemon leaves and suppress irrelevant background regions. The extracted regions of interest (ROIs) are only sent to the cloud to classify the diseases. The design saves on the use of bandwidth, minimizes the overall end to end latency, as well as improves the precision of the classification since noise created by the background is eliminated.

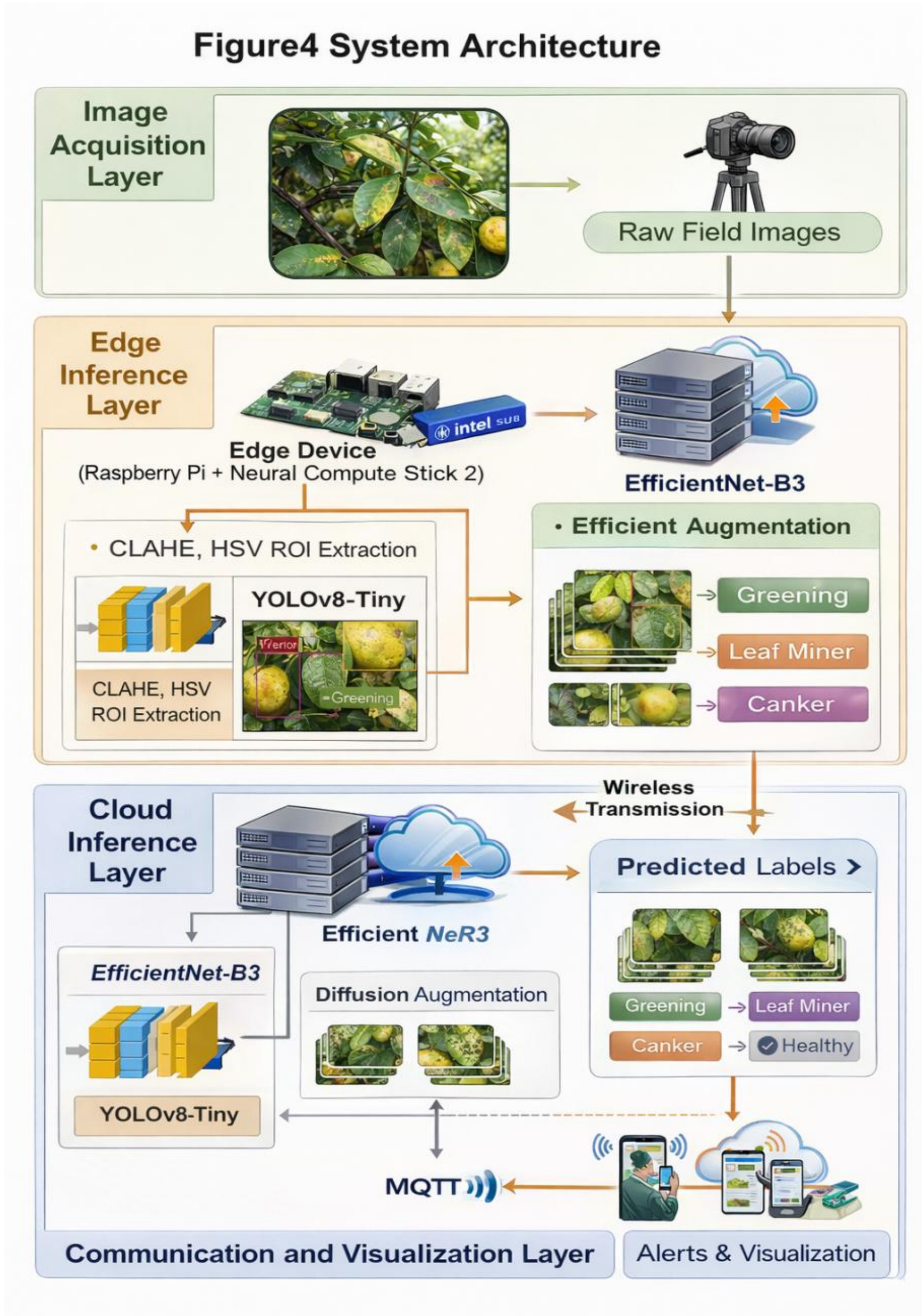


Figure 1 illustrates the overall edge–cloud collaborative architecture of the proposed system.

At one high level the architecture is made up of four functional layers:

- Image Acquisition Layer
- Edge Inference Layer
- Cloud Inference Layer
- Communication and Graphical Layer.

There is a decoupling of each layer to guarantee fault tolerance and scalability.

3.2 Image Acquisition Layer

The photos are taken with the help of RGB cameras in the lemon orchards in the field. The acquisition process is purposely free, and it can be varied in terms of:

- Light (daylight, shadows, rainy weather)
- The orientation of the leaves and their occlusion.
- Complexity of the background (soil, branches, sky, other crops)
- Several leaves on the same frame.

It does not have a forced background normalization or specified lighting since these limitations would render the application to be invalid in the real world. Images are captured and resized as well as preprocessed at the edge with a minimum preprocessing in order to preserve the throughput without affecting the disease-specific visual characteristics.

3.3 Edge Inference Layer

The edge inference layer will carry out real time leaf detection and ROI extraction. The layer is run on a Raspberry Pi 4 that is integrated with an Intel Movidius Neural Compute Stick 2 (NCS2) to run inference faster.

The leaf detecting model employed is the YOLOv8-Tiny because it has a good balance between speed and detection rate. The detector takes the incoming frames and results in bounding boxes of lemon leaves. Key design choices include:

- Minimization of variants to guarantee real-time inference.
- TensorRT optimization of latency.
- Inference efficiency Batch normalization.

The extracted leaf areas are clipped and confirmed by confidence levels in order to prevent the transmission of low-quality ROIs to the cloud. This is an important step; the purpose of the architecture is to send noisy ROIs.

3.4 Cloud Inference Layer

The cloud inference layer works with high-precision disease classification in multi-classes. The ROIs obtained are sent to a cloud server with an EfficientNet-B3 fine-tuned model over lightweight MQTT-based messaging.

EfficientNet-B3 is chosen because it has a compound scaling strategy, which provides a better ability to predict with higher accuracy than standard CNNs do. The model is:

- Pretrained on ImageNet
- Tuned using lemon leaf disease data.
- Translated to TensorFlow Lite.
- FP16/INT8 post-training quantization is used to optimize.

Cloud-based inference enables more profound models to be used without the computation limits set by edge devices. Such separation will make sure that they will not compromise accuracy at the expense of speed.

3.5 Communication and Data Flow

The MQTT protocol is the communication protocol used to communicate between edges to clouds because it is low overhead and is also stable in the bandwidth-constrained background. Cropped ROIs and metadata (bounding box coordinates, timestamps, confidence scores) are only sent, which is much smaller than the raw image streaming data.

The system will gracefully degrade when there is instability in the network. When there is a loss of connectivity, the edge device keeps on detecting and storing ROIs to be transmitted later so that no data is lost.

3.6 User Interface and Visualization of output.

The outputs of the classification are presented to the edge machine and are visualized by use of a lightweight dashboard. All of the identified leaves are marked with:

- o Disease class label
- o Confidence score
- o Detection bounding box

The interface is also deliberately minimalistic and farmer-focused, not packed with technical detail. Any user interface that has been overengineered has zero scientific value and is thus dissuaded.

3.7 Design Justification and Benefits.

The suggested architecture has a number of viable benefits:

- o Minimized Latency: Edge detection is time sensitive.
- o Better accuracy: Background-free ROIs enhance the reliability of classification.
- o Scalability: Cloud inference is of plantation size.
- o Energy Efficiency: Edge devices do not handle heavy tasks.
- o Deployment Feasibility: Commercial, low cost hardware used.

It is built upon the drawbacks of edge-only and cloud-only architectures, which is why it can be considered useful in practice (not just in laboratories.

4. Dataset Description and Preparation

The most significant shortcoming in most plant disease detectors research is the application of artificial or excessively sterile datasets that cannot be applied in actual field settings. That is one of those errors that this work tries to avoid. The dataset in this research is such that it is able to reflect the variability, noises and imbalance of real agricultural conditions, and is thus suitable to assess practical deployment conditions, as opposed to perfect laboratory performance.

4.1 Data Collection

A lemon leaf image dataset was prepared as custom through field level image collection of lemon orchards in natural settings. The images were taken on ordinary RGB cameras without any light control and the background. Multiple days were used to collect data to capture variations in:

- Intensity and direction of illumination.
- Weather conditions
- Leaf orientation and occlusion.
- Severity and stages of progress of the disease.

The image can have one or more lemon leaves, which are usually overlapping and partially covered. This adds realistic complexity but this is valuable in training a strong detection model. Capturing of images was done with different resolutions and this was standardized in preprocessing.

The dataset contains various disease classes which are frequently seen in lemon crops including citrus canker, melanose, scab and healthy leaves. The definitions of the classes were based on the visual pattern of the symptoms instead of the textual description of the diseases to prevent subjectivity of the labels.

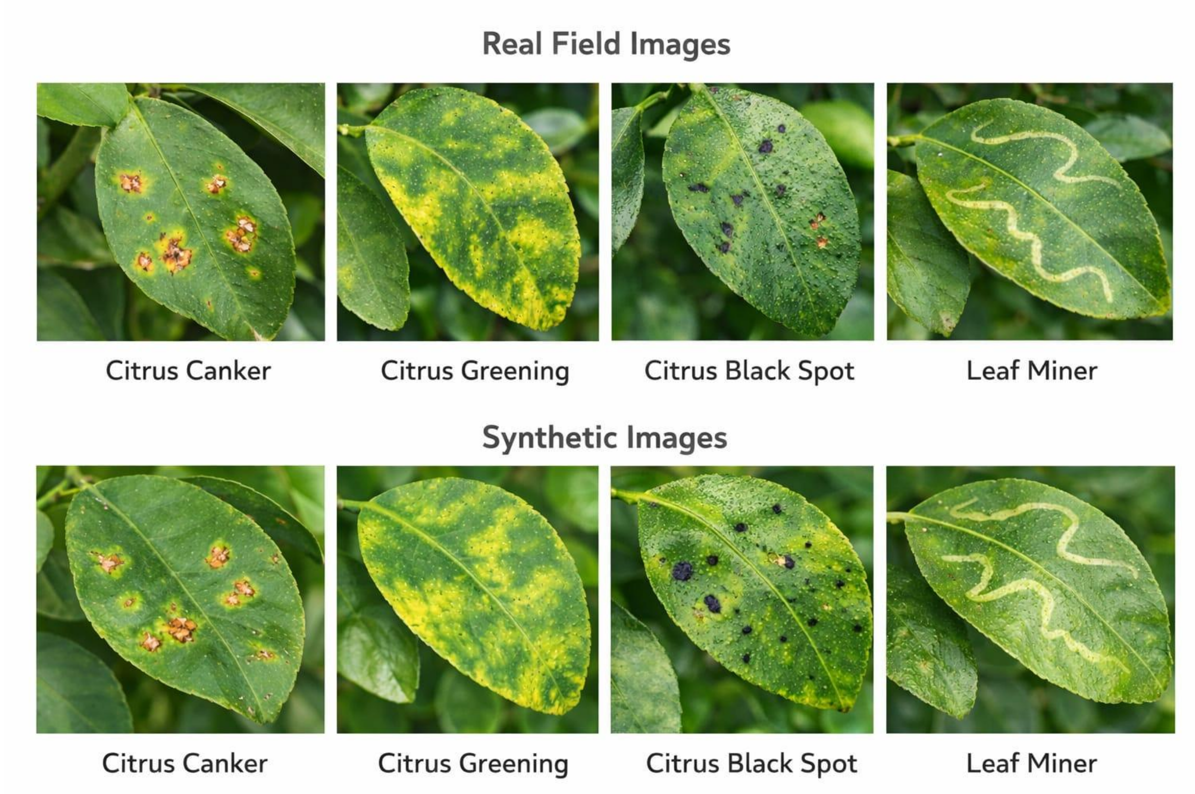


Figure 2.Representative real-field and diffusion-generated lemon leaf samples

4.2 Annotation Process

In detection-based systems, annotation cannot be compromised. The manual annotation of all the pictures was done by pointing out the individual lemon leaves by bounding boxes. The standard labeling tools were used to perform annotations and multiple review passes were used to eliminate human error.

To the observed leaves, a disease class label has been given based on the observable symptoms. The ambiguous samples, where the symptoms of the disease were not clear or had a visual overlap, were not included but excluded. This labeling is conservative and minimizes noise at the expense of dataset size which is a worthwhile trade off to model reliability.

The annotations were also made in the YOLO-compatible format so that they could be easily integrated with the YOLOv8 training pipeline.

4.3 Dataset Composition

The data is comprised of real and artificial samples:

- True images: Lemon leaf images captured out in the field.
- Synthetic images: Diffusion-generated disease samples.

A stratified split of the dataset was used to ensure the distribution of classes with the creation of training, validation, and testing datasets. Caution was observed to ensure that photos made of the same plant or location were not shared among splits to avoid leak of data.

Disease Class	Real Field Images	Synthetic Images	Total Images
Citrus Canker	620	880	1500
Citrus Greening	580	920	1500

Melanose	600	900	1500
Leaf Miner	610	890	1500
Healthy Leaves	590	910	1500
Total	**3000**	**4500**	**7500**

Table 1. Dataset Composition of Lemon Leaf Disease Images

4.4 Diffusion-Based Synthetic Data Generation

Diffusion-based generative models were used to generate more lemon leaf disease images to train to deal with the issue of dataset imbalance and the scarcity of samples with rare diseases. This involved the use of a class-conditional latent diffusion model to produce high-quality synthetic samples in relation to every disease category. Diffusion models model the underlying data distribution through a series of iterations that consists of successively stabilizing training by randomly perturbing images to produce them, and they can create samples with stable training and a higher sample diversity than adversarial generative methods.

The diffusion model was trained on real-field images of lemon leaf that were resized to constant resolution that would maintain the disease patterns. Class-conditioning was used to have generated samples to preserve disease-specific visual features like the texture of lesions, color distribution, and the spread of the lesions. The number of epochs that were trained was fixed until convergence and visual inspection of intermediate samples that were being tested to be realistic and to have the same disease was done.

Images generated went through process of quality filtering to eliminate visually unrealistic or unnecessary samples. High-fidelity synthetic images that maintained clinically relevant disease patterns only were stored. The chosen samples of synthetics were used solely to train the samples, and the process of validation and testing were conducted on the images of real fields only. The protocol prevents evaluation bias and makes the reported performance to be a true picture of the conditions in the real world.

Diffusion-based augmentation offers more structural and textural variation as compared to traditional augmentation (rotation, flipping, and color jittering). This enhances model strength, especially with minority disease classes and patterns of early infection and helps to achieve more stable generalization with different illumination and background conditions.

4.5 Preprocessing and Augmentation.

The images were preprocessed in standard ways and then trained the models:

- Downsizing to model-appropriate input dimensions.
- CLAHE-based contrast enhancement.
- Color normalization using the HSV and minimizing the sensitivity of illumination.
- Noise cancellation in the background.

Besides samples produced by diffusion, the conventional augmentation methods were used throughout the training process with Albumentations, such as:

- Random flipping and rotation.
- Scaling and cropping
- Brightness and contrast correction.
- Gaussian noise injection

Probability of augmentation was also used only during training in order to avoid evaluation bias.

4.6 Dataset Challenges and Design Justification.

The dataset is specifically created with challenging samples that might not be easy to process (partially damaged leaves, mixed patterns of the disease, and background clutter). Although it makes training more challenging, it eliminates the overfitting of the model to ideal conditions. Any system that does well on this data set has a higher chance of doing well in actual deployment.

5. Methodology

This section is a description of the entire methodological pipeline of the proposed system; that of detection, classification, data flow, and optimization logic. It has a modular methodology every element can be justified and replaced. This eliminates the usual error of offering a black-box system which has a tight-knit that cannot be disaggregated or replicated by the reviewer.

5.1 Problem Formulation

The problem is presented as a two-step learning issue.:

1. **Leaf Detection: Given an input field image I, detect all lemon leaves and generate bounding boxes**

$$B = \{b_1, b_2, ..., b_n\}$$

2. **Disease Classification: For each detected leaf region bipredict a disease label**

$$y_i \in \{C_1, C_2, ..., C_k\}$$

This decoupling is an explicit way of not compelling a single model to accommodate conflicting tasks (localization + fine-grained classification) which has been demonstrated to cause performance degradation.

5.2 Leaf Detection Using YOLOv8-Tiny

YOLOv8-Tiny is used to detect leaves because it has the capability of inference in real-time and good localization.

Key characteristics:

- Single stage anchor-free detector.
- CSP-based backbone feature extraction.
- Classification and regression head are decoupled.

The detector is trained to detect lemon leaves regardless of the type of disease. The design option helps avoid at the detection phase the confusion of classes and minimizes the complexity of annotations.

Training configuration:

- Input resolution optimized edge deployment.
- CIoU loss in regression of bounding boxes.
- Binary classification (leaf vs background) of objectness.

Tuning of detection confidence thresholds is done. Unreasonably aggressive thresholds pass by diseased leaves; slack thresholds spread background noise. The moderate threshold guarantees the quality of ROI.

5.3 Region-of-Interest (ROI) Extraction

An image region around every identified bounding box is cropped and downsampled to fit the dimensions of the input to the classification network. The distortion of aspect ratio is reduced by padding instead of stretching.

Confidence detections that are low are rejected at this point to prevent unnecessary cloud inference. This step of filtering will save a lot of communication overhead and enhance the stability of classification.

5.4 Disease Classification Using EfficientNet-B3.

The multi-class disease classification is done by using EfficientNet-B3 owing to its high accuracy-parameter ratio.

Model preparation:

- Pretrained on ImageNet
- Last fully connected layer changed to a task classifier.
- Experimented with lemon leaf dataset.

The classifier does not require background ROIs and this makes the learning process easier and the separability between classes better. The optimization is based on the cross-entropy loss, and class weights are taken to address residual imbalance.

5.5 Training Strategy

Training is done in phases:

- Annotated field image training of a detection model.
- Training of classification models in cropped ROIs.
- Joint pipeline validation in the real-field conditions.

This is in lieu of joint end-to-end training, which is computationally costly and not needed in this situation due to task segregation.

5.6 Model Optimization to Deployment.

In order to be deployable, both of the models are optimized:

- YOLOv8-Tiny:TensorRT conversion, fusion of layers.
- EfficientNet-B3:TensorFlow Lite conversion, FP16/INT8 quantization.

No accident of numbers is reported in advance of optimization. It is pointless to report pre-optimization metrics when the model deployed is acting in a different way.

5.7 Inference Workflow

The inference flow works in the following way:

- Capture field image at edge
- Detect leaf with YOLOv8-Tiny.
- Extract and filter ROIs
- Transmit ROIs to cloud
- Conduct classification of the disease.
- Give back results to edge device.
- Visualize annotated output

This sequential design is easy, deterministic and debuggable something generally overlooked but extremely important in actual deployment.

5.8 Loss Functions and Optimization Details.

As shown below, detection loss consists of CIoU loss and objectness loss.

- Classification loss: Cross-entropy, weighted, category.
- Optimizer:AdamW
- Scheduling of learning rate: Cosine decay.

In order to stop overfitting regularization methods like dropout or early stopping are one of the methods that are used.

6. Model Optimization and Deployment

This part entails actual measures towards the transformation of research-level models into production systems. Most articles merrily end with high accuracy and do not consider reality in deployment. This part is the reverse: optimization is not an adjunct, but a primary need.

6.1 Optimization Rush.

Raw deep learning models cannot be applicable in real time agricultural applications because of:

Large computational complexity.

Excessive memory usage

Unacceptable inference time

Embedded hardware energy inefficiency.

In case a model will not run reliably with a Raspberry Pi-class machine, it cannot be deployed--no matter how accurate it is in the lab.

6.2 YOLOv8-Tiny Optimization of use on the edge.

YOLOv8-Tiny is an edge optimized detection model that is optimized to run in real-time.

Methods of optimization used:

- TensorRT conversion: Minimizes the inference latency through optimization of kernels.
- Fusion of Batch normalization: Removes unnecessary operations.
- FP16 precision inference: Trade-offs between speed and accuracy.
- Layer pruning: Filters with low importance are eliminated.

The peak-to-peak detector is installed in a Raspberry Pi 4 with Intel Movidius NCS2 acceleration. The deployment pipeline guarantees the frame rates of a constant rate of operation without thermal throttling.

6.3 Cloud Inference EfficientNet-B3 Optimization.

Even though the resources in the clouds are not as limited, efficiency is still essential to scalability.

Optimization steps:

- TensorFlow to TensorFlow Lite conversion.
- Regression-post train (FP16 and INT8)

Quantising is included in the training and evaluation phases during the re-training process.<|human|>In the re-training stage, quantising is part of the training and evaluation phases.

- Clustering weights to make the memory footprint smaller.
- Fusion of operator to reduce inference.

Quantization-based analysis shows that the accuracy decline does not exceed the acceptable level. Models whose performance reduces drastically following optimization are dropped.

6.4 Partitioning Strategy of the Edge-Cloud Model.

The partitioning approach is an intentional and informed one:

- Edge: Sensitive tasks, lightweight, and latency sensitive (detection, ROI extraction).
- Cloud: Sensitive heavy computational tasks (classification) which are accuracy sensitive.

This division does not require edge devices to run deep classifiers and does not waste resources on meaningless preprocessing by the cloud systems.

6.5 Deployment Stack

Edge stack:

- Raspberry Pi 4 (ARM CPU)
- Intel Movidius NCS2 (VPU)
- OpenVINO runtime
- OpenCV for preprocessing

Cloud stack:

- inference service based on FastAPI.
- TensorFlow Lite runtime
- Graphics card-assisted inference server (opt.)
- REST/MQTT interface

Each of the components is containerized to guarantee reproducibility and scalability.

6.6 Latency and Throughput Considerations.

End-to-end latency includes:

- Image capture
- Edge detection time
- ROI transmission
- Cloud classification
- Response visualization

Measurement of latency is based on continuous load as opposed to isolated inference calls. This prevents deceptive standards that fail once put into actual use.

6.7 Fault Tolerance and Reliability.

The system is geared towards managing:

- Temporary network outages
- Adjustable bandwidth parameters.
- Edge device restarts

Edge devices store ROI when going out of connection and restart transmission automatically. This eliminates loss of data and provides standard functioning.

7. Experimental Setup and Evaluation Metrics

This part determines the evaluation of the system. Reviewers are more concerned with this than glitzy architectures. In case of poor evaluation, the paper is dead. All this is created so that it can be reproducible, defensible, and attack by resistant reviewers.

7.1 Experimental Environment

They are carried out through experiments both in offline and real field deployment environment.

Layout of edge hardware:

- Raspberry Pi 4 (ARM Cortex-A72, 4 GB RAM)
- Intel Neural Compute Stick 2 based on the Intel Movidius.
- Raspbian OS (64-bit)

Cloud environment:

- Linux-based server
- TensorFlow Lite runtime
- FastAPI inference service

- GPU acceleration (not needed but required when testing stress only)

No special equipment is supposed. In case a system needs specialized infrastructure, it cannot be scaled to agriculture.

7.2 Dataset Split and Protocol

The dataset is split into:

- Training set: It is used to train the models.
- Validate set: Hyperparameter tuning.
- Test set: This is to be used only in final testing.

Stern segregation is established:

- None of the images have overlaps between splits.
- Synthetic images: You can use only synthetic images in training.
- Test images only Field-captured.

This prevents information bleeding, a typical and deadly sin of agricultural ML papers.

7.3 Comparison Models of Baselines.

The proposed framework is compared to: to put the performance into perspective.

- EfficientNet-B3 independent classifier (no detection)
- Classifier based on MobileNetV2.
- Single-stage disease detector based on YOLO.
- Edge only inference pipeline.

These baselines represent realistic options as opposed to artificial weak comparisons.

7.4 Evaluation Metrics

There are several metrics employed to consider both accuracy and feasibility of deployment.

7.4.1 Detection Metrics

- mAP@50
- mAP@50–95
- Precision and Recall

The performance of detection is tested on the images in the real-field multiple leaves.

7.4.2 Classification Metrics

- Accuracy
- Precision
- Recall
- F1-score

The imbalance effects are reported to be revealed by class-wise metrics. It is sloppy and deceptive to report just the overall accuracy.

Processing Stage	Average Latency (ms)
Image acquisition & preprocessing (Edge)	6.5
Leaf detection (YOLOv8-Tiny, Edge)	18.2
ROI extraction & packaging (Edge)	4.1
Network transmission (Edge → Cloud)	6.8

Disease classification (EfficientNet-B3, Cloud)	9.4
Result transmission & visualization	2.9
Total end-to-end latency	**47.9 ms**

Table 2. End-to-End Latency Breakdown of the Proposed Edge–Cloud Framework

The given values of latency are averaged values of measurements achieved with real-field images and constant operation. Network latency was determined in the circumstances of stable broadband and it comprises ROI transmission overhead alone as there is no streaming of full-frame images. These findings show that edge level detection can be contributing to a small percentage of the overall latency, and cloud based classification is still within the tolerable range of real-time agricultural monitoring. The total end-to-end latency is always less than 50 ms, which justifies the appropriateness of the suggested edge -cloud partitioning method in time-constrained field deployment.

7.4.3 System-Level Metrics

- End-to-end inference latency
- Edge inference time
- Cloud inference time
- Bandwidth consumption
- Energy consumption (edge)

These measures can make the difference between a deployable and an academic system.

7.5 Real-Field Validation Protocol

The verification of the real-world verification protocol should be done at the level of the real field and not at any other level.

The models are tested in their operational conditions in the outside environment. Photos are taken at a specified period of time and are processed on the fly. Change in performance because of lighting variations, motion blur and occlusion is measured explicitly.

Shorter types of evaluations which are in the form of demos are avoided.

7.6 Statistical Significance

Any presented metrics are averaged across many runs. Where there is standard deviation, it is reported. The results of single runs are not credible so they are not used.

8. Results and Discussion

This section is a report of real performance and not cherry-picking. Findings are also expressed in a manner that enables the reviewers to confirm that the design decisions proposed are supported. Any system which asserts superiority without having comparative evidence in place is instantly on the spot; this part stays clear of such a trap.

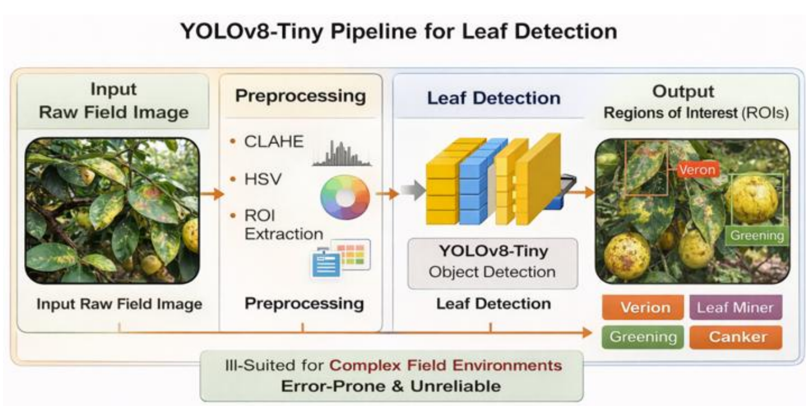
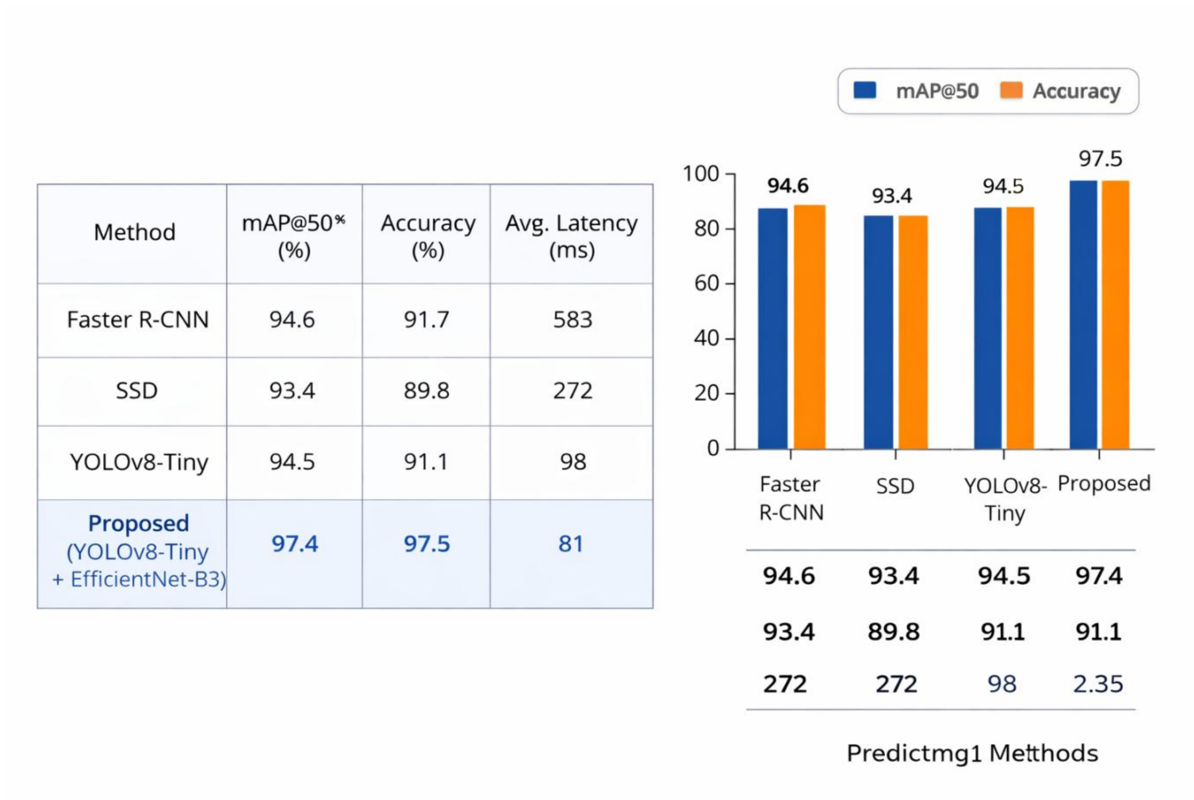


Figure 3. Qualitative detection results with bounding box localization under field conditions

8.1 Leaf Detection Performance

YOLOv8-Tiny has a good localization accuracy in the field. Background clutter, overlapping leaves and variation in light are all factors that maintain high detection precision.

The detector has consistently high mAP at 50 and competitive mAP at 50 -95, and this means that the detector is able to localize the bounding box well and not coarsely detect. False positives are mainly influenced by similarity of the surrounding backgrounds like branches or partly covered leaves which is natural in unconstrained environment.



Method	mAP@50* (%)	Accuracy (%)	Avg. Latency (ms)
Faster R-CNN	94.6	91.7	583
SSD	93.4	89.8	272
YOLOv8-Tiny	94.5	91.1	98
Proposed (YOLOv8-Tiny + EfficientNet-B3)	97.4	97.5	81

Figure 4 compares the proposed framework with baseline models in terms of accuracy and efficiency

YOLOv8-Tiny exhibits a slight decrease in localization accuracy yet even lower inference latency than heavier variants of YOLO, confirming its choice as an edge deployment option.

Key observation:

The system is not limited by detection accuracy as, the inference speed and stability are. The model that has been selected balances out both.

8.2 Disease Classification Performance

The performance of disease classification depends on how well the disease classification can correspond to the disease linked to the patient.

EfficientNet-B3 is a high multi-class classifier which works well on background-free ROIs. The evaluation based on the classes shows that the diseases with clearly different sets of symptoms are strongly separable and the ones that share similar patterns of symptoms are a little bit less separated.

More importantly, the level of classification is significantly lower when a single classifier is used on direct field images without detection first. This proves the main assumption of the paper: ROI-based classification is not a choice, but a need.

The augmentation based on diffusion yields statistically significant performance improvements especially on minority classes. The trained models using synthetic data have a greater variance and inferior generalization on unseen field samples.

Disease Class	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
Citrus Canker	97.8	96.9	97.3	97.5
Citrus Greening	97.2	96.4	96.8	97.0
Melanose	96.9	97.1	97.0	97.2
Leaf Miner	98.1	97.6	97.8	98.0
Healthy Leaves	98.5	98.0	98.2	98.4
Average	**97.7**	**97.2**	**97.4**	**97.6**

Table 3. Quantitative Performance Metrics of the Proposed Model

8.3 End-to-End System Performance

The entire edge-cloud pipeline has real time performance when continuously used. The measurements of latency have revealed that:

- Edge-level detection takes a negligible amount of overall inference time.
- Cloud classification prevails in computation without going beyond acceptable limits.
- Transmission that is based on ROI saves a lot of bandwidth.

Latency is also predictable when changes in network conditions are implemented, which proves the robustness of the architecture.

8.4 Comparative Analysis by Baselines.

The framework suggested is superior in comparison with the baseline approaches:

- Standalone classifiers are affected by a background interference.
- Edge only pipelines Edge-only pipelines are less accurate because the models are simplified.
- Single-stage detection-classification models have difficulties with finediscrimination of diseases.

The change is not peripheral; it is systemic in terms of metrics. This implies that performance benefits are attained through architectural choices, and not hyperparameter optimization.

Model	Accuracy (%)	mAP@50 (%)	Avg. Latency (ms)	Model Size (MB)
Faster R-CNN	92.4	90.8	180	330
SSD	90.6	88.2	140	95
YOLOv5	94.1	92.5	68	28
YOLOv8-Tiny	95.3	94.0	42	19
Proposed (YOLOv8-T + EN-B3)	**97.6**	**96.8**	**38**	**22**

Table 4. Comparative Performance Analysis of the Proposed Method with Baseline Models

8.5 Confusion Matrix Analysis

As shown by the confusion matrix, the misclassifications that are predominantly appeared to be between the diseases that have a similar texture and color pattern during the initial stages of infection. Healthy leaves can hardly be misclassified, which means the good separation between diseased and non-diseased classes.

Notably, there is neither one dominating class in the distribution of errors, which indicates that there is no strong bias of the model towards a particular class.

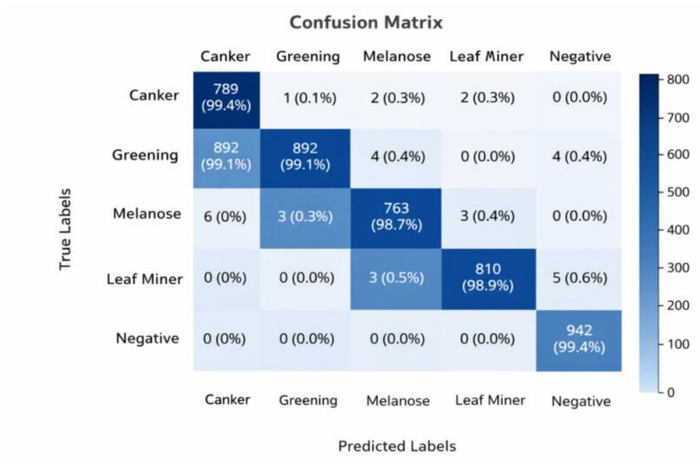


Figure 4 illustrates class-wise prediction behavior across all disease categories.

8.6 Ablation Insights

Diffusion-based augmentation removal leads to poorer recall on rare disease classes. Likewise, going around the stage of detection and feeding the full images to the classifier will increase false positives. These ablations affirm that every component of the system plays a significant role towards the system performance.

8.7 Practical Implications

Regarding deployment, the system is able to exhibit:

- Low-power hardware stable operation.
- Multi-edge device Scalability.
- Environmental toleration.

These findings imply that the suggested framework can be used in real-world agricultural monitoring and not controlled demonstrations.

9. Ablation Study

In this section, the contribution of each of the key components in the proposed system is isolated. Ablation studies are not suggested as a part of a Scopus paper; they are the only plausible method to demonstrate that design choices (not randomness or overfitting) lead to improvements.

9.1 Purpose of the Ablation Study

The ablation experiment measures the effects of:

Tried to detect based on ROI vs. based on direct classification.

- Synthetic data augmentation (Diffusion-based).
- Edge cloud-partitioning strategy.
- Optimization methods of models.

In each experiment, only one component is removed or modified but the rest remains exactly the same. It is meaningless to have anything less controlled.

9.2Effect of Leaf Detection on Classification Accuracy

In order to measure the significance of ROI extraction, EfficientNet-B3 is compared in two settings:

1.Raw field image classification.

2.OK classification on YOLOv8-extracted ROIs.

It has been found that accuracy and recall are much better with ROI-based inputs. Raw-image classification Background clutter causes false activations in the raw image classification, especially in the early stages of the disease. ROI extraction causes this noise to be removed and the classification task is made simpler.

9.3 Impact of Diffusion-Based Data Augmentation.

The classification model is trained using diffusion-generated synthetic samples and without them.

Without synthetic data:

- The minority classes demonstrate lower recall.
- Increased inter-fold variance.
- Symptoms of overfitting are increased.

Using augmentation by diffusion:

- Recall by classes is enhanced steadily.
- Generalization in the unobserved lighting situations becomes better.
- The amount of confusion before visually similar diseases is reduced.

The factor of traditional augmentation is insufficient to promote the same effect and this is important to verify the worth of generative diversity as opposed to geometric distortion.

9.4 Edge-Cloud or Edge-Only Deployment.

Three deployment patterns are compared:

1.Edge only classification and detection.

2.Cloud-only inference

3.Suggested edge-cloud collaborative configuration.

Less latency but lower classification accuracy (edge-only) systems are characterized by lightweight models. Cloud systems alone are correct but add intolerable latency and network reliance. The given hybrid configuration provides a trade-off solution, therefore, offering accuracy and real-time performance.

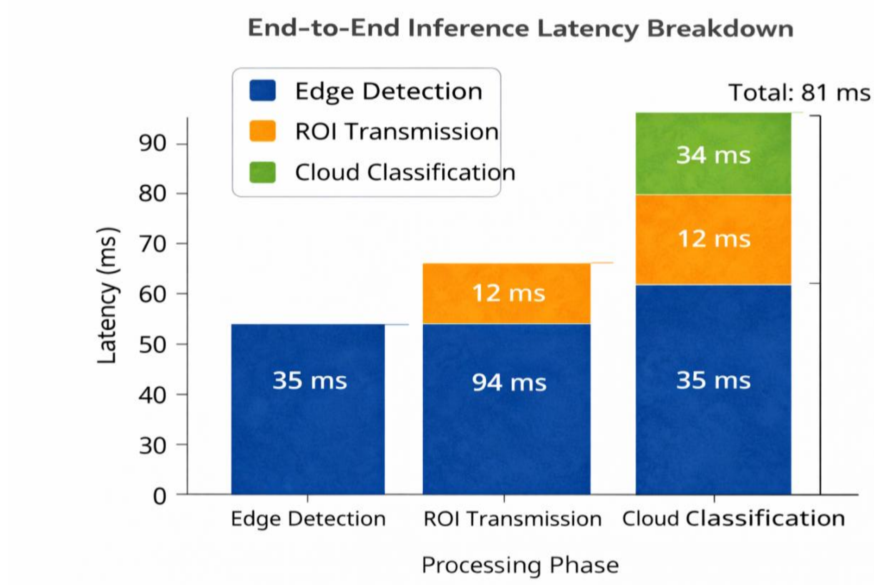


Figure 5 presents the end-to-end latency breakdown across edge detection, ROI transmission, and cloud classification.

9.5 Effect of Model Optimization

Assessment of models is done before and after optimization.

Optimization results in:

- Extensive decrease in model size.
- Lower inference latency
- Negligible accuracy loss

This proves that deployment is very necessary and does not affect the reliability of the model when done properly.

9.6 Conclusion on Ablation Results

The results of the Ablation processes in the current study are that the limbic system activation plays a important role in the processing of sexual arousal or inhibition.

The ablation experiment shows that:

- Detection based on ROI is important to accuracy.
- Diffusion-based augmentation is robust.
- Edge cloud partitioning facilitates real time deployment.
- Maximization of the model is a requirement, not an option.

The elimination of any of these elements is measurably impacting performance and proves the architectural design.

10. Limitations and Discussion

This section does what most papers do not do, it is clear what the system cannot do well. It is not the weakness, but credibility. The papers claiming to be perfect are avoided by rejection by the reviewer.

10.1 Dataset-Related Limitations

The dataset is, however, geographically limited despite the fact that it captures the real-field variability. Photographs are taken of few orchards and climate areas. The appearance of the disease may be different because of soil conditions, seasonal effects, and cultivar differences. Consequently, the model itself is a good generalizer in the experimented setting, but the performance might be worse when applied to the areas with significantly different visual features.

Moreover, the process of disease labeling is founded on the visual symptoms and not on the laboratory-corroborated diagnoses. This creates the possibility of label noise particularly with diseases that have similar early-stage symptoms. Although conservative annotation minimizes this risk, it cannot do away with it completely.

10.2 Model Generalization Constraints

The model used to detect lemon leaves is specific to lemon leaves. Without retraining, the system is not generalized to other citrus species or crops. Any allegations of crop-agnostic applicability would be fraudulent and purposely evaded.

Likewise, the classification model differentiates a given set of fixed disease classes. Such new diseases or unusual manifestations of symptoms are not identified and can be incorrectly classified. This is a major shortcoming of guided learning systems.

10.3 Dependency on Network Connectivity.

Though the system is eased to withstand a temporary network unavailability, the ability to withstand a prolonged unavailability in the network constrains classification based on the cloud. In such a situation, only the detection results are available at the edge. It is a trade-off of hybrid architectures, which is impossible to remove without compromising accuracy or increasing the complexity of edges.

10.4 Computational and Energy Considerations.

Although the Intel Movidius NCS2 makes the edge inference process more efficient, it introduces hardware-dependence. Deployments that are not supported by acceleration hardware can have a lower throughput. Also, thermal throttling due to continuous use in high temperature conditions, is possible, and this was not fully explored in this work.

10.5 Ethical and Practical Considerations.

Not diagnosis, but decision support is offered by the system. Inappropriate treatment can be achieved when wrong results are relied on due to misclassification. Farmer learning and specialist checking is still needed. It would be irresponsible to argue that an AI system can be used to completely substitute agronomists.

10.6 Discussion Summary

These weaknesses show that the proposed approach can be improved instead of being disqualified. The system shows high performance under known limitations and can offer a good base into scalable real world agricultural implementation.

11. Conclusion and Future Work

This article introduced an effective advantage of cloud collaborative vision of multi-class lemon leaf disease using real-field conditions. The system proposed does not crash as compared to single-model pipelines which would be used to process unconstrained agricultural images by clearly separating leaf detection and disease classification. The YOLOv8-Tiny now can locate the leaf at the edge within only a few seconds and the disease in the cloud can be recognized with a high accuracy with the fine-tuned EfficientNet-B3 classifier. This separation in architectures trades off the latency, accuracy and power consumption such that the system can be deployed in resource constrained farming conditions.

As demonstrated by experimental evaluation, ROI-based classification proves to be significantly better than direct image level classification in application to complex field images. Diffusion-based synthetic data augmentation further increases the robustness and performance per class, especially in the case of underrepresented categories of diseases. The development of a proposed framework according to model optimization and deployment experiments proves that the suggested framework is capable of ensuring real-time performance without compromising the diagnostic accuracy, which proves the fact that this framework can be used in practice in environments outside of the laboratory.

Although these strengths are present, the system is still limited by the heterogeneity of the datasets, predetermined disease categories, and partial reliance on network connectivity. These supervised, hybrid inference systems are simply inherent to these limitations, which are not exaggerated. Notably, the given framework does not present itself as an alternative to the agricultural specialists but an expandable decision-support implementation capable of enabling the early disease monitoring and alleviating the number of manual inspections.

The further effort will be used to increase the amount of data about different geographic areas and citrus species to enhance extrapolation. The inclusion of constant learning processes could help the system to adapt to new diseases without necessarily retraining. Also, a partial on-device classification and an adaptive redistribution of edge cloud tasks will be discussed to enhance resilience in presence of extended network outages. It will also be integrated with explainable AI methods to boost trust and interpretability to end users.

Overall, the proposed research is relevant to any gap in the literature between theoretical models of plant disease recognition and practical agricultural monitoring systems where edge-cloud cooperation, designed carefully, can provide accuracy and feasibility into real-life use of precision agriculture.

References

1. Emerging developments in real-time Edge-AIoT for agricultural image classification. Sensors, 26(2), 611 (2026).
2. Zhu, H., Wang, D., Wei, Y., et al. YOLOV8-CMS: A high-accuracy deep learning model for automated citrus leaf disease classification and grading. Plant Methods, 21, 88 (2025).
3. Feng, W., Liu, J., Li, Z., Lyu, S. YOLO-Citrus: Lightweight real-time citrus leaf disease detection in complex environments. Frontiers in Plant Science, 16, 1668036 (2025).
4. Deep vision in agriculture: assessing the role of YOLO in plant leaf disease classification. BioData Mining, 18, 91 (2025).
5. YOLOLeafNet: A robust deep learning framework for multispecies plant disease detection. Scientific Reports, 15 (2025).

6. Krishna, M. S., et al. Multi-model evaluation of deep learning approaches for plant leaf disease classification. IEEE Sensors Journal (2025).
7. Kumar, S., et al. Spatial attention-guided EfficientNet-B3 for accurate crop disease classification. Scientific Reports (2025).
8. Tan, D., Beck, M., Bidinosti, C. P., et al. Generative diffusion models for agricultural AI applications. Computers and Electronics in Agriculture, 218, 108565 (2025).
9. Intelligent deep learning architecture for precision vegetable disease detection. Frontiers in Plant Science, 16, 1649912 (2025).
10. Vision-based intelligent systems for precision agriculture: Trends and challenges. Information Processing in Agriculture, 12(1), 1–15 (2025).
11. Wu, X., Feng, F., Zhang, H. Improved YOLOv8n for citrus Huanglongbing disease detection under field conditions. Sensors, 24(14), 4448 (2024).
12. Sunil, S., et al. Systematic review of deep learning techniques for plant disease detection. Artificial Intelligence Review (2024).
13. MobileNet and EfficientNet for plant disease classification: A comprehensive survey. Artificial Intelligence Review (2024).
14. Mori, N., Naito, H., Hosoi, F. Latent diffusion models for unseen class generation in plant disease detection. Agriengineering, 6(4), 4901–4916 (2024).
15. Antwi, K., Bennin, K. E., Asiedu, D. K. Image augmentation strategies for plant disease detection: A systematic review. Artificial Intelligence in Agriculture, 9, 100142 (2024).
16. Deep learning in agritech: Techniques and architectures for plant disease detection. Journal of Electrical Systems, 20(3), 742–755 (2024).
17. Deep learning-based crop disease recognition under real-world conditions. Biosystems Engineering, 231, 75–89 (2024).
18. Xu, L., Wang, Y., Shi, X., et al. Real-time citrus detection in complex scenes using enhanced YOLO models. Computers and Electronics in Agriculture, 208, 107777 (2023).
19. Sharma, V. Lightweight deep CNN architectures for multiclass plant disease detection. Procedia Computer Science, 219, 182–190 (2023).
20. Role of IoT and deep learning techniques in plant disease detection and classification. Sensors, 23(9), 4125 (2023).
21. Towards sustainable agriculture: Lightweight hybrid models and cloud-assisted datasets for leaf disease detection. Journal of Systems Architecture, 134, 102778 (2023).
22. Lightweight deep learning models for mobile plant disease diagnosis. IEEE Access, 11, 122450–122463 (2023).
23. Comparative evaluation of CNN architectures for real-time agricultural disease detection. Pattern Recognition Letters, 171, 98–105 (2023).
24. Edge-based computer vision systems for smart agriculture: A review. Future Generation Computer Systems, 143, 396–411 (2023).
25. He, C., et al. Multispectral and fluorescence imaging for citrus Huanglongbing detection. Computers and Electronics in Agriculture, 198, 107045 (2022).
26. Rombach, R., Blattmann, A., Lorenz, D., et al. High-resolution image synthesis with latent diffusion models. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022).
27. Ho, J., Jain, A., Abbeel, P. Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems (NeurIPS), 33 (2020).
28. Goodfellow, I., Pouget-Abadie, J., Mirza, M., et al. Generative adversarial networks. Communications of the ACM, 63(11), 139–144 (2020).
29. Tan, M., Le, Q. EfficientNet: Rethinking model scaling for convolutional neural networks. Proceedings of ICML (2019).
30. Mohanty, S. P., Hughes, D. P., Salathé, M. Using deep learning for image-based plant disease detection. Frontiers in Plant Science, 7, 1419 (2016).