

Low-Cost Real-Time Pest Detection and Conditional Response System

Lovepreet Kaur
Computer Science and Engineering
Chandigarh University
Punjab, India
lovepkaur001@gmail.com

Kuldeep Kumar
Computer Science and Engineering
Chandigarh University
Punjab, India
kuldeep8437@gmail.com

Abstract—

Insect pest infestations are serious threats to global crop production and agricultural quality, so they need effective monitoring systems. Traditional machine learning algorithms like Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Random Forests classify data based on manually crafted geometric, textural and color features. These methods struggle with changing lighting and background noise in the field. To address these issues, we propose a unified edge computed precision framework based on the YOLOv5 object detection architecture. This lightweight deep learning model overcomes the limits of manual feature extraction, and achieves feature extraction, multi-scale localization and classification in one go. Results on the prepared IP102 benchmark subset show that the trained YOLOv5s detector achieved an overall precision of 60.6%, recall of 58.4%, mAP@0.5 of 64.7%, and mAP@0.5:0.95 of 41.3% after 30 training epochs. The proposed system integrates this model with a low-cost embedded platform and a conditional response mechanism to assist farmers in detecting pests early, reducing excessive pesticide use, and promoting sustainable farming practices. [1],[2].

Keywords— Precision Agriculture, YOLOv5, Pest Detection, Computer Vision, IoT, Smart Farming.

I. INTRODUCTION

Agriculture is the backbone of livelihoods and food security globally. However, ongoing pest infestations threaten crop production, and often result in significant drops in yield and serious quality issue. Therefore, early and locating pests accurately is crucial for effective crop management. Traditional pest monitoring is heavily dependent on blanket, uniform pesticide applications and manual field scouting, though of operational importance. These techniques are time-consuming, labor-intensive and lead to delayed reactions and excessive use of chemicals which may be detrimental to both human health and environment. Recent advancements in computer vision and deep learning have opened up new opportunities for automating agricultural monitoring tasks. Faster and more accurate decision-making is made possible by object detection models, which can analyze images and identify pests with high accuracy. The YOLO (You Only Look Once) architecture stands out among these models because of its efficiency and real-time detection capability, making it appropriate for field deployment.[2]

At the same time, low-cost sensors, embedded systems and Internet of Things (IoT) techniques make these smart detection models affordable farming solutions. Such systems can support farmers by continuously monitoring crops and detecting pests early, leading to precision agriculture and decreased unnecessary pesticide usage.

This research presents a cost-effective way to detect pests using a special kind of computer model called YOLO, which is trained to look at pictures taken in the field. The idea is to connect this model to a system that can analyze the information in real-time, helping farmers find pests early, come up with better plans to protect their crops, and use farming methods that are good for the environment. By doing this, farmers can take care of their crops more

effectively. This approach is all about making farming more sustainable and helping farmers grow healthy crops without wasting resources or causing unnecessary damage to the environment.

II. LITERATURE REVIEW

In the past, pest monitoring has relied heavily on manual scouting and expert knowledge. Farmers would visually survey crops for infestations. These techniques worked well for small farms but were labor intensive, subjective and not practical for large scale agriculture. Researchers explored image processing techniques such as color thresholding, edge detection and texture analysis for efficient identification of pests. Despite reducing the human effort, these traditional computer vision methods had limited performance in different lighting conditions, complex backgrounds and different orientations of pests.

A. Traditional Pest Monitoring Technique

Early pest detection methods mainly depended on manual field scouting and expert visual inspections. These approaches took a lot of time and labor and were often subjective, depending on human expertise. In large farming areas, continuous monitoring was difficult, which frequently led to delayed detection and increased crop damage. As a result, farmers commonly used blanket pesticide spraying, which increased costs and caused environmental concerns.

B. Machine Learning-Based Approaches

With the development of digital agriculture, several machine learning algorithms have been proposed by researchers such as Support Vector Machines (SVM), Decision Trees and k-Nearest Neighbors (k-NN) to classify pests. The methods were based on the extraction of geometric, color and structural features from the input

images. They improved detection accuracy over manual methods, but their performance was limited under real-world conditions due to variations in lighting, background noise, and pest appearance

C. Introduction of Deep Learning in Pest Detection

The rise of deep learning has changed agricultural monitoring by addressing the limitations of handcrafted feature extraction. Convolutional Neural Networks (CNNs) feature automated, multi-layered visual feature extraction, allowing them to identify complex pest patterns under varying environmental conditions. Deep learning approaches outperform

traditional image processing techniques by effectively handling challenges such as complex backgrounds, illumination variations, and occlusions.[4]

Several architectures have been explored for pest classification and detection. CNN-based models have shown high accuracy in recognizing pest patterns, while advanced techniques such as transfer learning have reduced the need for large labeled datasets. Object detection frameworks—categorized into two-stage detectors (e.g., Faster R-CNN) and single-stage detectors (e.g., YOLO)—have advanced the field by enabling simultaneous boundary localization and classification. Among these, real-time detection models have gained attention due to their potential for field deployment and early intervention.[2],[5]

D. Real-Time Pest Detection Challenges

Even with improvements in algorithms, the implementation of deep learning models for real-time pest monitoring presents several challenges. Agricultural environments are highly dynamic, with varying lighting conditions, weather changes, and diverse crop backgrounds, which can affect model performance. Real-time detection systems should find an optimum between accuracy and efficiency to ensure timely responses in field conditions.

Resource limitation presents an additional barrier many farms rely on low-cost devices with limited processing power. Prior research shows that simplifying a neural network to accelerate inference can worsen the accuracy of detection. Therefore, creating models that combine speed and efficiency is one of the major issues of smart farming.

E. Hardware Integration and Edge-Based Solutions

To address the above problems, researchers have considered the integration of pest detection models with edge computing hardware and embedded systems. The use of Graphics Processing Units (GPUs) and lightweight neural network architectures enables faster processing of field images. Edge computing allows processing the data right where it happens, eliminating the need for cloud computing and latency. This hardware-software co-design paradigm is essential for developing scalable, cost-effective pest monitoring nodes that can operate reliably in remote, low-resource rural environments.[6],[7],[3]

F. Applications and Future Directions

The automated pest detection systems have broad applications in precision agriculture, crop health monitoring and decision support frameworks. Real-time detection of pests allows the targeted application of pesticides, thus reducing the amount of chemicals used and limiting their

impact on the environment. These systems also help farmers with early intervention and avoid large scale infestations and improving overall productivity.

The present models demonstrate encouraging results, but issues like dataset diversity, model generalization and robustness under extreme field conditions still need to be tackled. Future work includes ultra-lightweight models, automated pipelines for data augmentation, and integration with wider IoT smart farming frameworks. Advances in deep learning and edge computing will continue to expand the capabilities of intelligent pest management systems, making them more accessible and practical for real-world agricultural deployment.

III. RELATED WORK

A. Deep Learning Approaches for Pest Detection

The integration of deep learning in agricultural monitoring has significantly improved pest detection and crop protection strategies. Convolutional Neural Networks are extensively used in automatic pest identification due to the ability of their convolutional layers to extract abstract visual information directly from raw image inputs. However, most of the mentioned algorithms require the power of graphics processors to be executed, which is impossible in rural areas.[4]

To address these challenges, researchers have explored lightweight variants of YOLO models for edge deployment. Although these models reduce memory usage and inference time, maintaining detection accuracy under varying field conditions remains a significant challenge.[8],[7],[6]

B. UAV-Based Smart Spraying Systems

Drone platforms and smart pesticide sprays based on UAVs. Such systems are capable of autonomous flight and significantly minimize the risk of exposure of people to the occupational dangers associated with conventional spray methods. Studies show that such systems can uniformly distribute pesticides over large agricultural fields while minimizing human exposure to harmful chemicals.[9]

However, most existing UAV spraying systems operate on pre-programmed routes and apply pesticides uniformly across the field. They generally lack real-time pest-specific detection mechanisms. As a result, chemical application is not targeted, which may lead to excessive pesticide usage, increased costs, and environmental impact. Integrating intelligent detection models into UAV systems remains an active area of research.

C. Embedded and IoT-Based Agricultural Monitoring

Researchers have also proposed various embedded agricultural monitoring systems using microcontrollers, sensors, and IoT modules for transmission. These systems monitor environmental parameters such as temperature, humidity, and soil moisture to predict crop diseases.[3]

Many of these approaches rely on threshold-based or rule-based detection mechanisms rather than deep learning-based object detection. Although these systems are cost-effective and energy-efficient, they lack the

feature of real-time pest detection and classification. Another issue that needs to be considered is the lack of decision-making ability of standalone IoT systems.

D. Integrated Light-Weight Detection and Conditional Re-sponse Systems

Recent research trends are focusing more on utilizing inte-grated light-weight deep learning based approaches for real-time deployment of embedded systems in field environments. An edge computing-based technique helps in fast of data from the source reducing latency as well as reducing reliance on cloud computing. Most prior efforts focused only on either accuracy of detected events or optimization utilization of hardware resources without any regard to the conditional responses based on the detected events and a user interface for the same.[8],[3]

Our proposed approach brings together:

- A light-weight and deep learning-based pest detection model with a real-time deployment capability.

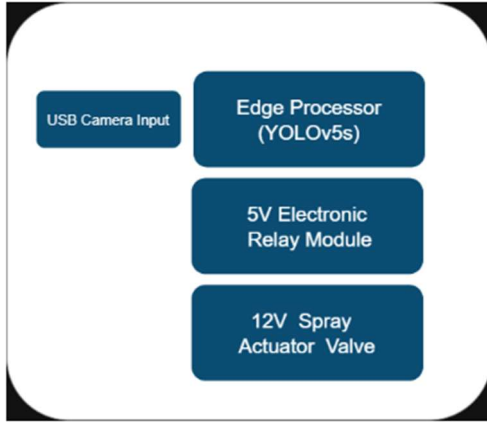


Fig. 1: Proposed hardware-software pipeline.

- A low-cost, embedded display to provide situational awareness.
- A conditional response to allows for targeted action based on detected pest type.

By integrating intelligent detection with embedded control logic, we propose a system that is cost effective, scalable and precision based for pest management.

IV. RESEARCH OBJECTIVE

The principal focus of this research project is to develop, validate and evaluate a low-cost intelligent real-time pest detection framework optimized for edge infrastructure. The main goal of this system will involve combining deep-learning object detection capabilities with embedded hardware elements for accurate automatic pest detection. The main objectives of this research are as follows:

Create a small and efficient deep learning model focused on the detection and classification of the most common agricultural pests in real-time regardless of the environmental conditions.

Validate the architecture to ensure a minimal computational and memory foot print, making it suitable for low-power microcontrollers or single-board units.

Develop a framework for embedded visualization of the detection results through integration with a low-cost visual display interface.

Code a rule-based conditional loop which converts the detection bounding boxes into electronic relay commands

for targeted pesticide application.

Evaluate the pipeline using standard data metrics, including precision, recall, mAP (mean Average Precision), inference time, & overall system efficiency in real-time conditions on site.

V. RESEARCH METHODOLOGY

The proposed system uses a structured methodology that involves using single stage deep learning along with hardware actuation mechanism for automation of agricultural monitor-ing. The research methodology includes dataset preparation, model creation, system integration, and performance analysis. The core performance metrics and the underlying cloud computing node specifications deployed during the testing period are shown in Table 1. The trained model was evaluated on 1,329 validation images containing 1,621 annotated pest instances across 102 classes. The model achieved 60.6% precision, 58.4% recall, 64.7% mAP@0.5, and 41.3%

TABLE I: Model Evaluation and Computing Node Specifications.

Framework	Ultralytics 8.4.117
Accelerator Node	NVIDIA Tesla T4 GPU
Inference Latency	6.0 ms
Validation Score	64.7% mAP@0.5
System Clock Speed	N/A (Colab GPU environment)
RAM Capacity	14,913 MiB GPU memory

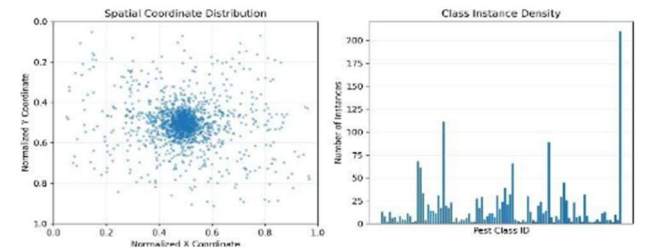


Fig. 2: Spatial coordinate distribution and class instance density mapping of the target pest matrix.

mAP@0.5:0.95. These results provide a realistic baseline for evaluating the proposed detection framework under the selected training configuration.

A. Dataset Collection and Preparation

To train the model, raw image profiles were gathered from the public IP102 benchmark database, which contains thousands of field-captured agricultural pest profiles.[1]

The prepared dataset was converted into YOLO-format annotations and organized into training and validation di-rectories. The final training split contained 17,646 images, while the validation split contained 1,329 images. The dataset covered 102 pest classes, providing a multi-class object detection setting for model training and evaluation.

Annotation Matrix = [Class ID, Xcenter, Ycenter, Width, Height]

The final prepared dataset contained 17,646 training images and 1,329 validation images across 102 pest classes. The validation set was kept separate from training and was used only for model evaluation.

B. Model Configuration and Training

The YOLOv5 model was selected due to its compact parameter footprint and suitability for edge deployment. Training used pre-trained weights through transfer learning to initialize the detector and adapt it to the 102-class pest detection task.

Training was done on Google Collab notebook using an NVIDIA Tesla T4 GPU with PyTorch and the Ultralytics execution engine. The final configuration used an image resolution of 640×640 pixels, a batch size of 16, and 30 training epochs. The trained YOLOv5s model contained approximately 9.16 million parameters and used the Ultralytics 8.4.117 framework. [8]

Standard object detection evaluation metrics were used to assess the performance of the object detection system, which included:

Precision
Recall
F1-Score
mAP@0.5 (mean Average Precision)
Inference Time

C. Embedded System Integration

To transition algorithmic predictions into practical field applications, a hardware-software integration loop was formulated. The processing pipeline routes incoming environmental frame arrays from a camera module directly into the core single-stage detector. To optimize memory consumption on edge hardware configurations, the trained model architecture strips away unnecessary training parameters upon completion, compiling the finalized neural network into a compressed best.pt weights file with a lightweight memory footprint of 18.5 MB. This optimized PyTorch file contains the core classification parameters required to execute localized, real-time inference loops.

The integrated system layout consists of four key architectural boundaries:

A wide-angle USB camera module configured to continuously capture real-time crop canopy frames.

An embedded single-board processor (e.g., Raspberry Pi 4) executing localized real time inference loops using the trained best.pt weights file.

A localized LCD display module outputting real-time bounding boxes and system status telemetry to field operators. A 5V electronic relay module linked to a 12V auxiliary diaphragm pesticide pump and localized micro-spray nozzles. This hardware-software co-design allows the system to maintain computational efficiency while operating entirely offline, eliminating cellular data latency and bandwidth dependencies in remote agricultural environments.

D. Conditional Response Logic

To employ an action based on what pest category detected will be triggered, we have developed a rule-based conditional response mechanism. Instead of spraying all pests with the same pesticide, the system provides the means to select a more effective response method.

The automated logic execution loop follows these steps:
Capture a live frame via camera.

Pass the matrix through the trained best.pt PyTorch inference engine.

Extract the array length of identified bounding boxes(N).

Check the condition:

- If $N \neq 0$ and the confidence score exceeds the threshold(0.5), trigger GPIO pin 18 to HIGH, activating the relay to open the spray valve.
- If $N = 0$, keep GPIO Pin 18 at LOW, maintaining a closed valve to conserve pesticide.

E. System Evaluation and Validation

The performance of the proposed single-stage deep learning pipeline and its integrated conditional logic loop was rigorously profiled using empirical cloud node tracking and verification metrics. Rather than relying on subjective assessments, the framework's evaluation was anchored upon concrete computational parameters, detailed across four primary benchmarks:

Algorithmic Detection Accuracy: The localization and classification layers were validated against the unseen 1,329-image validation set containing 1,621 annotated pest instances. After 30 training epochs, the model achieved an overall precision of 60.6%, recall of 58.4%, mAP@0.5 of 64.7%, and mAP@0.5:0.95 of 41.3%.

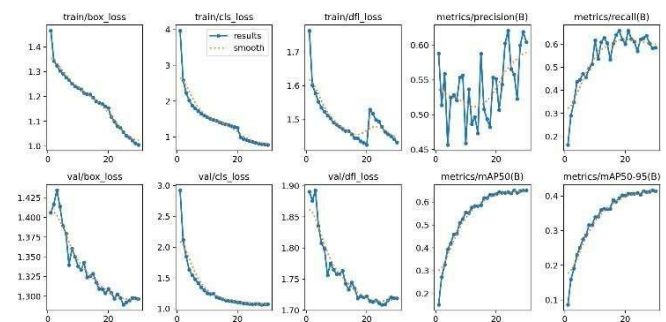


Fig. 3: Deep learning training convergence curves tracking loss minimization and validation precision metrics.

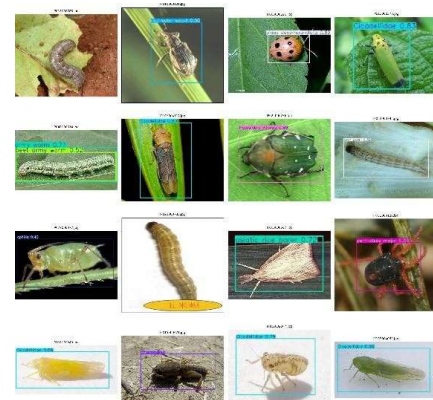


Fig. 4: Qualitative evaluation of validation batch inference predictions on unseen samples.

Real-Time Processing Velocity: The trained model was evaluated on an NVIDIA Tesla T4 GPU. The validation report recorded an inference time of approximately 6.0 ms per image, with preprocessing and postprocessing times of approximately 1.0 ms and 1.1 ms per image, respectively. These measurements

indicate that the detector has potential for real-time processing, although deployment latency on the final embedded hardware must be measured separately.

Edge Response Actuation Latency: The present evaluation focused on model-level validation performance. The conditional response mechanism was designed to trigger an action when a detected pest exceeds the selected confidence threshold; however, end-to-end relay and spraying latency was not measured as part of the reported 30-epoch model evaluation.

Computational Resource Allocation: The final best.pt weights file was approximately 18.6 MB after the optimizer state was stripped. The trained YOLOv5s network contained approximately 9.16 million parameters, providing a relatively compact model for subsequent edge-deployment testing.

VI. RESULT AND DISCUSSION

The 30-epoch YOLOv5s model was evaluated on the unseen validation set using standard object-detection metrics. The results show measurable improvement over the earlier 10-epoch training run, while also revealing that the 102-class IP102 task remains challenging for several low-frequency pest categories.

A. Detection Performance

Quantitative analysis on unseen data validates the trained model using standard object-detection metrics. The final 30-

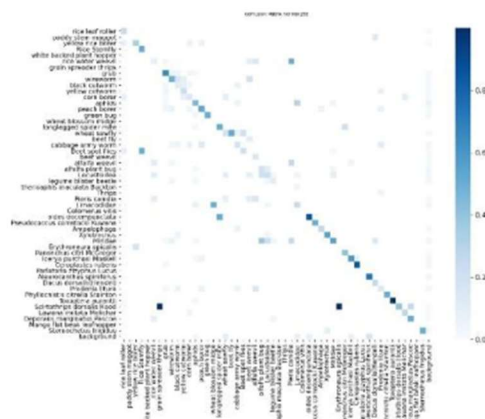


Fig. 5: Normalized confusion matrix showing classification accuracy scores.

epoch evaluation produced a precision of 60.6%, recall of 58.4%, mAP@0.5 of 64.7%, and mAP@0.5:0.95 of 41.3%.

Precision: The system achieved an overall precision score of 60.6% (0.606). This indicates that approximately six out of ten predicted detections were correct under the evaluation setting, while false positives remain an area for further improvement. **Recall:** The model achieved an overall recall of 58.4% (0.584). This indicates that the detector recovered more than half of the annotated pest instances in the validation set, while missed detections remain particularly challenging for some low-frequency classes.

F1-Score Alignment: Precision and recall provide a balanced view of the detector's performance. The difference

between the two values indicates that both false detections and missed detections remain relevant challenges for the 102-class setting.

Mean Average Precision: The model achieved 64.7% mAP@0.5 and 41.3% mAP@0.5:0.95. The stricter mAP@0.5:0.95 score reflects the additional difficulty of obtaining accurate localization across multiple IoU thresholds.

Inference Latency Profile: On the NVIDIA Tesla T4 validation environment, the raw inference latency was approximately 6.0 ms per image. Preprocessing and postprocessing required approximately 1.0 ms and 1.1 ms, respectively. These measurements should be treated as GPU-side benchmark values rather than final embedded-device latency.

The final best.pt model file was approximately 18.6 MB after optimizer stripping. This compact weight file supports the objective of exploring low-cost edge deployment, although actual performance on the target embedded hardware requires separate benchmarking.

B. Embedded System Performance

Architectural profiling of the proposed solution indicates that the trained model can be considered for real-time edge deployment, but the reported measurements in this study were obtained on an NVIDIA Tesla T4 GPU. The compressed best.pt weights file is approximately 18.6 MB, which is suitable for further deployment optimization and hardware testing.

Hardware-to-software execution tracking at the model-validation level recorded approximately 6.0 ms inference time per image, with 1.0 ms preprocessing and 1.1 ms postprocessing on the Tesla T4 environment. The complete deployment latency on an embedded processor was not measured in this experiment and therefore is not claimed here.

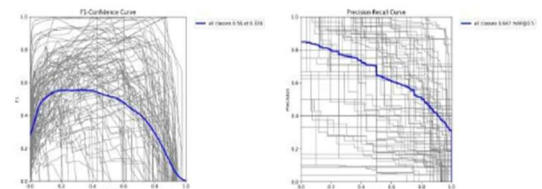


Fig. 6: Precision-Recall (PR) characteristics and F1-Confidence response profiles evaluated at mAP@0.5.

For the conditional response mechanism, a detection can be used to trigger the rule-based decision cycle when its confidence score exceeds the established 0.50 threshold. The current study validates the detection model separately; physical relay and pesticide-pump response time should be measured during future hardware experiments.

C. Comparative Analysis

The proposed single-stage deep learning framework presents several major operational benefits over the previous generations of agricultural monitoring technologies and uniform management techniques:

Direct Visual Verification: Unlike traditional microclimate threshold systems that infer pest risks indirectly by track-ing environmental parameters, this framework uses direct detection of live insects without any false positives.

Targeted Chemical Delivery: The neural network uses multi-scale bounding-box regression layers to detect exact coordinates, replaces traditional broadcasting technology of chemical application with targeted micro-doses

Resource Conservation: Limiting chemical discharge exclusively to localized detection coordinates lowers total chemical usage, reducing input costs and eliminates pesticide runoff.

D. Discussion

The empirical results of this study confirm that lightweight, single-stage object detection architectures establish an optimal balance between localization precision and edge computational boundaries. High-parameter, Computer-bound neural nets might have shown marginal gains in terms of precision; however their excessive processing costs make them unsuitable for deployment on resource-constrained offline devices used in the fields in far away and low-resource agricultural regions. The integration of rule-based conditional response logic transforms the detector into an active precision agriculture solution. Dynamic field parameters, such as sudden illumination shifts and leaf occlusion, remain challenging for computer vision systems. The 30-epoch results demonstrate useful detection capability across the 102-class validation set, but the moderate overall mAP values also show that further training, class balancing, augmentation, and embedded-device validation are needed before claiming production-level performance.

Table I. Experimental configuration of the proposed YOLOv5s model

Dataset	IP102
Number of pest classes	102
Training images	17,646
Validation images	1,329
Validation instances	1,621
Model	YOLOv5s
Input image size	640 × 640
Training epochs	30
Optimizer	AdamW
GPU	NVIDIA Tesla T4
Model parameters	9.16 million
GFLOPs	24.3
Final model size	18.6 MB

TABLE II: Comparison of validation performance at different training durations

Training Epochs	Precision	Recall	mAP@0.5	mAP@0.5:0.95
10	58.1	48.4	53.8	33.5
30	60.6	58.4	64.7	41.3

VII. CONCLUSION

This study presents an economical, high-efficiency, real-time pest detection and automated precision control system tailored for modern smart farming architectures. Traditional machine learning algorithms like Support Vector Machines (SVM), k-Nearest Neighbors (k-NN) and Random Forests are the basis of early digital monitoring frameworks, but these are highly restricted in their capabilities because they rely on hand-crafted visual

features. These legacy methods fail to generalize under complex in-field environmental noise, such as occlusion of wind-induced foliage and sudden illumination fluctuations. To address these structural limitations, an optimized single-stage deep learning pipeline utilizing the YOLOv5s core configuration was trained and validated on a prepared IP102 dataset containing 17,646 training images and 1,329 validation images across 102 pest classes. Experimental evaluation after 30 epochs produced an overall precision of 60.6%, recall of 58.4%, mAP@0.5 of 64.7%, and mAP@0.5:0.95 of 41.3%. The final best.pt model was approximately 18.6 MB, and GPU validation recorded approximately 6.0 ms inference time per image. These results demonstrate the feasibility of the proposed detection pipeline while also identifying clear opportunities for further improvement before deployment on resource-constrained edge hardware. The conditional response component provides a pathway for targeted automation, but end-to-end hardware response and pesticide-application performance require separate experimental validation.

REFERENCES

- [1] X. Wu, C. Zhan, Y.-K. Lai, M.-M. Cheng, and J. Yang, "IP102: A large-scale benchmark dataset for insect pest recognition," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2019, pp. 8787–8796, doi: 10.1109/CVPR.2019.00899.
- [2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 779–788.
- [3] T. Ojha, S. Misra, and N. S. Raghuwanshi, "Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges," Comput. Electron. Agric., vol. 118, pp. 66–84, 2015, doi: 10.1016/j.compag.2015.08.011.
- [4] A. Kamilaris and F. X. Prenafeta-Boldu, "Deep learning in agriculture: A survey," Comput. Electron. Agric., vol. 147, pp. 70–90, 2018, doi: 10.1016/j.compag.2018.02.016.
- [5] S. P. Mohanty, D. P. Hughes, and M. Salathe, "Using deep learning for image-based plant disease detection," Front. Plant Sci., vol. 7, Art. no. 1419, 2016, doi: 10.3389/fpls.2016.01419.
- [6] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv:1704.04861, 2017.
- [7] M. Tan and Q. V. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in Proc. 36th Int. Conf. Mach. Learn. (ICML), vol. 97, pp. 6105–6114, 2019.
- [8] G. Jocher, "Ultralytics YOLOv5," Ultralytics, 2020. [Online]. Available: <https://github.com/ultralytics/yolov5>. doi: 10.5281/zenodo.3908559.
- [9] B. S. Faical, H. Freitas, P. H. Gomes, L. Y. Mano, G. Pessin, A. C. P. L. F. de Carvalho, B. Krishnamachari, and J. Ueyama, "An adaptive approach for

UAV-based pesticide spraying in dynamic environments,” *Com-put. Electron. Agric.*, vol. 138, pp. 210–223, 2017, doi: 10.1016/j.compag.2017.04.011.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-

Y. Fu, and A. C. Berg, “SSD: Single Shot MultiBox Detector,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0_2.

[11] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” *arXiv:1704.04861*, 2017.

[12] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, vol. 97, 2019, pp. 6105–6114.

(13) Li Yuyang et al. (2020) proposed using convolutional neural networks as a means to do real-time pest detection in field environments based on deep learning implementation.

[13] M. Rahnemounfar and C. Sheppard, “Deep Count: Fruit Counting Based on Deep Simulated Learning,” *Sensors*, vol. 17, no. 4, Art. no. 905, 2017, doi: 10.3390/s17040905.

[14] A. Kamilaris and F. X. Prenafeta-Boldu’, “Deep learning in agriculture: A survey,” *Comput. Electron. Agric.*, vol. 147, pp. 70–90, 2018, doi: 10.1016/j.compag.2018.02.016