Journal of Computer Science

Research Article

Optimized YOLOv8 for Insect Detection in Agricultural Farms Using Edge Vision Systems

Deebalakshmi R., Adhithyaa S., Yoga Vignesh V. and Balaji Ganesh Rajagopal

School of Computing, SRM Institute of Science and Technology Tiruchirappalli, India

Article history
Received: 28-03-2025
Revised: 14-05-2025
Accepted: 16-06-2025

Corresponding Author:
Balaji Ganesh Rajagopal
School of Computing, SRM Institute of Science and Technology Tiruchirappalli, India
Email: balajiganeshrajagopal@gmail.com

Abstract: Farmers started to adopt smart gadgets like automated pest detection and kiosks for their work in light of the AI revolution in agriculture. These kinds of technologies help farmers identify pest infestations early, which allows the farmers to take action accordingly and prevent crop loss. This results in improvement in both yield and sustainability. However, running an advanced model such as YOLOv7 on low-power edge devices remains a challenge, especially in real time. To address the problem, our research brings an innovative Advanced Edge-Based Vision System designed for detecting small insects and classification in agricultural environments. The model built utilizes the Jetson Nano platform, TensorRT optimization, and an enhanced YOLOv8 model. To optimize the performance, the model uses compression techniques, such as INT8 quantization, and is accelerated by TensorRT. The YOLOv8 model is trained on a specialised insect dataset, and this optimization technique results in a tenfold reduction in memory usage, yet performs efficiently and also improves the inference speed using Jetson Nano by achieving a processing rate of 45 Frames Per Second (FPS). Despite these optimizations and memory reduction, the model serves at its best where the F1-scores exceeding 95% ensure accurate detection of the pest. This study demonstrates the successful deployment of lightweight, high-performance vision models on edge devices, enabling real-time, accurate pest detection. The findings contribute to advancing precision agriculture by making intelligent pest management more accessible and efficient.

Keywords: Model Optimization, Jetson Nano, Inference Engine, Quantization, Precision Agriculture, TensorRT

Introduction

Farmers experience various challenges due to pest infestations on their agricultural farms. The major problem is the financial strain for the small-scale farmers. Insects can lead to substantial crop losses, which raises questions about food security and the livelihoods in urban areas. The high expense of buying pesticides will be a financial burden for the farmers, and also not advisable for using it because of its toxicity, affecting the environment, soil, and water pollution, which affects the ecosystem. Additionally, exposure to pesticides poses health risks to farmers and their families. Poor handling and application can lead to both immediate and long-term health issues, affecting the well-being of farmers and agricultural consumers. This gap will hinder opportunity and lead to poverty. Developing this edge vision system offers an effective alternative to pesticides for pest control. Our design provides real-time monitoring and early detection of insects. Our technology empowers farmers, especially small-scale and subsistence farmers, to make decisions that can reduce pesticide usage and crop loss, as well as enhance their economic situation. Therefore, implementing intelligent edge vision systems for insect detection can alleviate many of the economic and social challenges faced by farmers. Insect infestations can significantly impact crop yields. Early detection is a powerful tool that can help farmers manage these issues. By reducing chemical pesticide usage, farmers can increase crop yields, reduce production costs, and enhance overall farm productivity. This technology also benefits consumers by reducing pesticide residues and promoting a healthier ecosystem. Precision agriculture is revolutionizing crop health by analyzing pest behavior, thereby enhancing pest management techniques. This technology benefits rural communities by increasing crop yields and reducing pesticide costs, while also promoting sustainable farming

practices and product safety, thereby attracting policy support and incentives. YOLOv8 was chosen due to its lightweight architecture, speed, and high detection accuracy, which makes it particularly suitable for real-time applications on edge devices. Compared to earlier YOLO variants, YOLOv8 offers better bounding box regression and classification, making it ideal for detecting small, fast-moving insects in complex farm environments.

Our work makes several key contributions toward deploying efficient pest detection systems in agricultural edge environments. First, we optimize the YOLOv8 architecture specifically for enhanced pest detection and classification under real-farm conditions. Second, we systematically validate multiple image augmentation techniques to improve model robustness and performance on diverse, real-time field data. Third, we achieve a notable increase in inference speed, measured in improved frames per second, through model recompilation and optimization. Finally, we integrate these advancements into a complete, end-to-end pipeline that is scalable, power-efficient, and readily deployable on resource-constrained edge devices such as the Jetson Nano.

As shown in Fig. 1, the system architecture integrates edge processing with deep learning inference.

Fig. 1: The assembled prototype

Related Work

Research on edge computing frameworks for pest detection (Cardoso *et al*., 2022; Diya *et al*., 2023; Rajak *et al*., 2023) demonstrates how decentralizing computation can enhance real-time identification, reduce latency, and support timely pest management. Complementary studies including deep learning methods for soybean insect pest counting (Ademola *et al*., 2022) and smart monitoring using optical fiber sensors (Domingues *et al*., 2022) further validate the effectiveness of automated detection systems. Collectively, this body of work establishes edge computing as a viable approach for improving the speed and responsiveness of precision agriculture applications. However, these studies often lack an in-depth examination of two critical deployment factors: the scalability and power efficiency of the proposed frameworks. Addressing these aspects represents a significant opportunity for further research and optimization.

A comparative analysis of YOLO and Faster R-CNN for insect detection in tomato crops (Diya *et al*., 2023; Kang *et al*., 2023) evaluates the relative merits of each model, concluding that YOLO offers superior real-time performance and accuracy. Parallel research into IoT-enabled pest monitoring networks (Sadowski & Spachos, 2018) and comprehensive reviews of IoT-based precision agriculture (Huang *et al*., 2018) further illustrate the growing role of automated detection systems. While these studies advance model selection and system integration, they often omit a critical dimension for field deployment: hardware optimization. This gap underscores the need for further investigation into efficient, edge-optimized implementations of high-performing models like YOLO.

Review articles on machine learning for plant disease identification and categorization (Diya *et al*., 2023; Makkena *et al*., 2023) provide a foundation relevant to automated pest detection. Complementary research into smart agricultural sensors (Tetila *et al*., 2020) and novel deep learning models optimized for edge deployment (Tang & Zhang, 2023) further highlights the growing integration of IoT and edge computing in precision agriculture. Collectively, these works advance machine learning applications for early disease detection, production optimization, and loss minimization. Building on this trajectory, a promising future direction involves the deeper integration of real-time edge computing with advanced machine learning to create more robust and responsive systems for agricultural pest and disease monitoring.

Research on low-power deep learning for edge devices (Shruthi *et al*., 2019; Albanese *et al*., 2022; Shi *et al*., 2022) provides valuable strategies for minimizing energy consumption in resource-constrained AI systems. While these works offer important overviews of model optimization and efficient inference, they lack a dedicated focus on agricultural environments. This limits their applicability and prominence as tailored solutions for the diverse and demanding conditions of real-world agri-tech applications.

Recent research (Baghbanbashi *et al*., 2023; Droukas *et al*., 2023) introduced a novel attention mechanism for rice pest detection, designed to address challenges such as complex backgrounds and small pest sizes. By dynamically adjusting attention weights, the model focuses computational resources on a limited subset of relevant pests. Performance was further enhanced through a multi-scale feature fusion network and a knowledge-condensed network optimized for edge deployment. This integrated architecture enabled rich feature extraction,

allowing the model to outperform advanced benchmarks, including YOLO, EfficientDet, RetinaDet, and MobileNet, in pest detection accuracy.

Moradeyo *et al*. (2023) proposed an algorithm for detecting potato late blight disease using a neural network-based approach. Designed to operate in uncontrolled field environments with complex backgrounds, the system first applies image preprocessing techniques such as decorrelation stretching to enhance color differentiation in input leaf images. Fuzzy C-means clustering is then used to segment regions of interest. Finally, a trained neural network classifies the affected regions by distinguishing them from similarly colored and textured backgrounds.

Yesuf and Assefa (2023) evaluated model compression techniques, including pruning, 16-bit quantization, and knowledge distillation, on a DenseNet architecture using the CIFAR dataset and Jetson Xavier hardware to optimize for embedded systems. Broader context for deployment strategies is provided by survey works, such as the analysis of robotic harvesting systems by Kargar *et al*. (2022) and comprehensive studies on neural network compression. However, these studies often omit a critical practical metric: inference speed. To address this gap, our analysis directly compares runtime efficiency alongside accuracy. Table 1 presents this performance comparison, benchmarking YOLOv8 against alternative models on key deployment metrics.

Table 1: Comparative review of various literature related to the proposed model

Ref. No.	Algorithms/ Learning Models	Major Contribution	Areas for Improvement
Albanese *et al*. (2021)	DNN	Use of DL models to identify pest infestation from photos obtained from low-power embedded sensors. Energy harvesting integration ensures extended battery life.	Wireless Communication Energy Harvesting at different weather conditions
Shin and Oh (2022)	R-CNN, CRA-NET	Design of Embedded Systems with DL Models Introduced the method of Novel model compression for low-power devices	Very limited rice dataset Data imbalance of certain rice crop varieties
Shruthi *et al*. (2019)	K-means clustering	Challenges like complex backgrounds and lighting are considered Employed color enhancement and clustering for segmentation	Scalability and resilience of the FCM clustering Algorithms applicability of algorithms in practical scenarios
Moradeyo *et al*. (2023)	R-CNN, Fast R-CNN, RPN, YOLOv3, SSD	Issues like Model adaptability and data labeling shortages are mitigated	Pipeline for embedded device deployment
Wang *et al*. (2023)	CNN	Utilized a Deep Convolutional Neural Network (CNN) with novel layer fusion and normalization functions	Limited dataset with limited classes, Deployment on embedded systems
Baghbanbashi *et al*. (2023)	Faster R-CNN, YOLOv3, SSD, Cascade R-CNN, Mask R-CNN	integrates attention mechanisms and a multi-scale feature fusion network for improved accuracy, as well as a specialized knowledge distillation network for inference on edge devices.	Efficacy of the YOLO-GBS in multi-class datasets.
Shruthi *et al*. (2019)	K-means clustering	GLCM texture features Optimized for Sugarcane leaf diseases.	Improved feature selection strategies Alternative classification algorithms for larger datasets.
Huang *et al*. (2018)	CNN, RCNN, faster RCNN, R-FCN, SSD, YOLOv3.	Enhanced YOLO V3 algorithm with faster detection times and improved accuracy for agri robots.	A large-scale dataset in different weather conditions.
Ali *et al*. (2023)	Faster-RCNN approach based on MobileNet	Achieving an accuracy of 82.43% on the IP102 dataset while addressing challenges such as image distortions and enhancing the generalization ability of pest recognition in agricultural applications.	It does not address the performance of the Faster-PestNet model on more diverse or larger datasets.

Table 1: Continued

Chithambarathanu and Jeyakumar (2023)	DCNN, CNN, LSTM, DBN	Emphasizing the need for automated systems to enhance agricultural productivity and reduce human error in pest identification.	There is a lack of standardized datasets, dependence on human expertise, and unclear evaluation.
Dong *et al*. (2024)	YOLOv5	Utilizing innovative methods like Multi-Level Spatial Pyramid Pooling and Content-Aware ReAssembly of Features to enhance accuracy while reducing parameters significantly.	Lacks extensive validation on diverse and large datasets, which raises concerns about the generalizability of the PestLite model's performance.
Yang *et al*. (2023)	YOLOv7	Achieving 93.23% accuracy through advanced techniques like deformable convolution and dynamic attention mechanisms, significantly enhancing pest recognition and control efficiency.	The reliance on a limited dataset of 782 images, which may not fully represent the diversity of tea tree pests, could affect the model's generalization.
Wang and Fu (2024)	RTMDet, DCNN	The model is fine-tuned with the YOLOv7 model, and some unwanted layers are pruned.	The VoVGSCSP module improves inference speed and mAP, which could limit the overall performance enhancement due to increased network depth and resistance to dataflow.
Anwar and Sarfaraz (2023)	CNN	Uses transfer Learning with pre-trained CNNs like VGG16, VGG19, and ResNetv50.	Reliance on a specific dataset, lack of detailed comparisons with state-of-the-art models, absence of robustness testing, and potential overfitting issues.
Venkatasaichandrakanthand, and Iyapparaja (2023)	CNN, MFO, and EViTA	This model effectively utilizes dual-branch segment representations to enhance feature extraction from pest images, leading to superior performance compared to existing models.	The datasets used for training and validation may not be sufficiently large or diverse to fully validate the model's effectiveness.
Sushma *et al*. (2024)	MFO,EviTA, CNN	Improves pest detection and classification in peanut crops by integrating dual-branch transformer architectures and advanced feature extraction techniques, leading to superior accuracy compared to traditional CNN models and existing vision transformer architectures.	Improves pest detection and classification in peanut crops by integrating dual-branch transformer architectures and advanced feature extraction techniques, leading to superior accuracy compared to traditional CNN models and existing vision transformer architectures.

Numerous studies have sought to optimize the performance and scalability of deep learning models for object detection and classification across diverse agricultural applications. While much of this research focuses on improving standard accuracy metrics, significantly less attention has been paid to optimizing these models for practical deployment on resource-constrained edge devices. This work addresses that gap by focusing on both the materials (datasets) and methods necessary to create an efficient, production-level insect detection model for edge deployment. Our goal is to highlight the pathway toward a scalable, end-to-end deep learning pipeline that can serve as a reliable inference engine within smart agricultural systems.

Materials and Methods

An Intelligent Edge Vision System for insect detection in agricultural settings is proposed in this work. The system leverages the power of the Jetson Nano edge device along with the TensorRT compiler over a customized YOLOv8 deep learning model. Data collection involves acquiring insect and non-insect images and augmenting the dataset. Model training comprises fine-tuning the YOLOv8 model on the custom dataset. The model is then optimized using TensorRT. Real-time insect detection is achieved by integrating the camera feed and deploying the model on the Jetson Nano, enabling precise and timely pest control in agriculture. The deployment of the Intelligent Edge Vision System for

insect detection in agriculture using Jetson Nano, TensorRT, and YOLOv8 showcased impressive outcomes in terms of model memory requirements, inference timing, and concurrency rate. NVIDIA's Jetson Nano is a small, low-cost computer designed for edge AI applications.

TensorRT is an optimization tool developed by NVIDIA for deep learning inference on GPUs. It is used to optimize trained deep learning models, making them faster and more efficient for real-time inference. In the edge vision systems presented, TensorRT has been employed to accelerate the inference of AI models for tasks like crop monitoring, disease detection, and pest control. YOLO (You Only Look Once) is a popular real-time single-shot object detection model used in a variety of computer vision tasks. However, many computer vision tasks lack the inclusion of the environmental setting and scalability for end-to-end production. YOLOv8 is an improved version with enhanced performance. This work has been utilized for tasks such as insect detection, crop disease identification, and weed detection, enabling precision agriculture practices.

Dataset Curation

The methodology describes the development of an Intelligent Edge Vision System for detecting agricultural pests using a dataset from the National Bureau of Agricultural Insect Resources (NBAIR), which includes 40 pest types found in field crops. Image pre-processing begins by converting RGB images to grayscale for edge detection and noise reduction using Canny edge detection. Bounding boxes are created around the insects, defined by four coordinates (x, y, w, h). Only bounding rectangles larger than 100 pixels in width and height are used for extracting Regions of Interest (ROIs) from the original RGB images. The extracted insect images are resized to 227×227 pixels for consistency. To increase model robustness, geometric transformations such as scaling, rotation, flipping, and transposing are applied, expanding the dataset. This augmentation strategy enhances dataset diversity while maintaining computational efficiency. Figure 2 shows pre-processed insect images, and Table 2 provides details about the species in the NBAIR dataset. The dataset consists of over 5481 annotated images covering mainly five classes: Asian Lady Beetle, Lady Bug, Mealy Bug, Pyrilla Perpusilla, and Stink Bug. Images were collected under different lighting and background conditions to ensure diversity and generalization.

Dataset Description:

Table 2: NBAIR Insect samples

Insect Name	No. of Samples	No. of Test Samples
Asian Lady beetle	876	300
Ladybug	503	300
Mealy bug	802	300
Pyrilla perpusilla	1,099	300
Stink bug	701	300
Total	3981	1500

Fig. 2: Forty Insect classes in the NBAIR insect dataset (Cropped and Preprocessed)

It is found that the number of samples for each of the five different classes of insects is not balanced. Imbalance occurs when certain classes have significantly fewer samples compared to others, leading to biased model training and poorer performance on the underrepresented classes. When there is an imbalance in the number of samples across different classes in the dataset, image augmentation is required to make the number of samples equal across the different classes of images. The various image augmentation techniques, such as gray scale conversion, color saturation, blurring, bounding box crop after zoom, and bounding box after rotation, are applied to the original dataset with 3981 samples across five different classes. On application of various augmentation techniques, 1500 samples are generated for each class of images, and the number of images over each class becomes balanced.

Grayscale conversion has been applied to a subset of 15% of images in the dataset at random and helps the model focus more on texture and structure rather than color information.

Color Saturation (-25 to +25%) adjustment alters the intensity of colors in an image, which helps the model to learn and recognize objects under different lighting conditions and color saturations, improving its generalization ability.

Image blurring (upto 2.5 px) reduced the sharpness of an image by averaging the pixel values within a small neighborhood (5x5), which introduced varying degrees of blurriness, forcing the model to focus on more abstract features rather than fine details.

Bounding Box Crop after Zoom (0 to 20%) involved zooming into an image and then cropping it based on the bounding box of the object of interest, which helped the model to learn and focus on the relevant object despite changes in scale and perspective.

Bounding Box Rotation (-15° to +15°) applied rotational transformations from -15 degrees to +15 degrees to the samples, which helped the model to be invariant to object rotations, enabling it to recognize objects from different viewpoints.

These kinds of image augmentation help to prevent the learning model from being biased towards the majority class and enable it to learn discriminative features for all classes. Table 3 summarizes the different augmentation techniques applied to the dataset.

Optimized Yolo v8 Architecture for Edge Vision Systems

YOLOv8 has emerged as a cutting-edge object detection system, offering superior speed and accuracy compared to its predecessors, making it highly effective in domains like livestock monitoring, PCB inspection, underwater fish identification, and insect detection in agriculture. Its anchor-free architecture significantly enhances real-time detection by eliminating the need for predefined anchor boxes, which improves both detection speed and reliability, especially for small or irregular objects. The multi-scale prediction capability of YOLOv8, powered by the C2f module and anchor-free detection head, ensures that the model can detect objects at various sizes and scales. In this proposed work, we focus on optimizing the detection layers P3, P4, and P5 since these layers handle different feature map resolutions crucial for detecting small objects like pests.

To tailor the YOLOv8 for pest detection, we modify the architecture by replacing the P3, P4, and P5 layers with custom CNN layers. Due to this replacement, we removed the P1 and P2 layers, which are very ineffective for detecting smaller objects. By removing them, the model architecture only focuses on high-resolution images for feature mapping, which better captures pest details. We then set the stride for P3, P4, and P5 layers to 1 (Fig. 3). By doing this, we are enhancing the model's sensitivity to smaller objects to improve detection accuracy, such as the pest size, colours, and textures. By doing this, there will be an increase in the model size, so we are applying post-training INT8 quantization to compress the weight memory and, at the same time, maintain the performance. This ensures the model optimization and remains efficient for real-time pest detection on edge devices.

Hyperparameter tuning was conducted to improve model accuracy, including learning rate adjustments, anchor box optimization, and input image resizing. To validate YOLOv8's suitability, we also experimented with Faster R-CNN and SSD models, which underperformed in speed and accuracy on edge hardware compared to YOLOv8.

Table 3: Description of different augmentation techniques applied on each class of images

Insect Name	No. of Samples	Augmentation Techniques	Samples after Augmentation
Asian Lady beetle	876	• Grayscale: 15% of images	1500
Ladybug	503	• Color Saturation: -25 to +25%	1500
Mealy bug	802	• Blur: Up to 2.5px	1500
Pyrilla perpusilla	1099	• BB Crop after Zoom: 0 to 20%	1500
Stink bug	701	• BB Rotation: -15° to +15°	1500

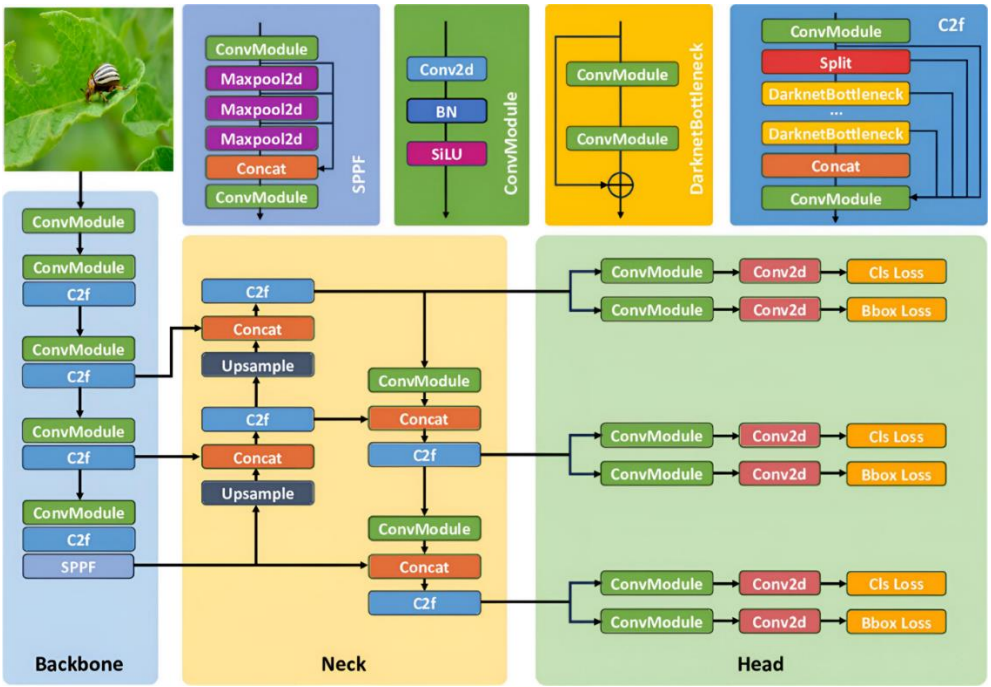

Fig. 3: Optimized Yolov V8 Architecture

Model Compression for Edge Deployment

Modified Backbone of YOLOv8

While testing YOLOv8 for pest detection in low-resolution images, we noticed a key limitation the backbone wasn't capturing enough detail for small pests. To overcome this, we replace the standard backbone with more lightweight but also more efficient feature extractors like MobileNetV3 and EfficientNet (Fig. 4). By changing this, we bring the perfect balance between accuracy and computation efficiency, ensuring even low-resolution images could also extract features accurately. By improving how the backbone processes visual data, we strengthened the detection layers, allowing them to make more precise predictions.

To define the model's input, we used Df × Df × M, where M represents the number of input channels. But here's where things get tricky traditional convolution operations require a significant amount of computation, following the formula (DP × DP × M) × (DK × DK × N). In this equation, DP represents the input's spatial dimensions, DK is the kernel size, and N is the number of output channels. The result? A high computational load can be a major challenge when running real-time object detection on resource-limited edge devices. By optimizing the backbone and refining feature extraction, we ensured that our model remained both efficient and highly accurate, making it well-suited for real-time pest detection in agricultural settings.

Modified Convolution Module

During the research, we mainly focused on enhancing the CNN modules in the detection layer of YOLOv8 (P3, P4, P5) to increase the accuracy of detecting low-resolution images. The key feature we modified in the research is by replacing the standard CNN layer with a custom-trained one, which focuses on the detection of small objects (Fig. 5). Through testing, we found that P1 and P2 mainly focus on detecting larger objects and contribute less to detecting small-sized pests. To improve the model, we pruned these two layers using Depth Separable Convolution (DSC), which reduces the unnecessary computation while retaining detection accuracy. Pruning these layers not only improves memory efficiency but also makes the model computationally lightweight yet more powerful. It also detects pests of different sizes, shapes, and colors. Additionally, we fine-tuned the stride configuration of P3, P4, and P5, setting it to 1 to boost sensitivity to small objects. This adjustment led to more precise detections, even in lower-resolution images. To optimize the model, we applied the INT8 quantization technique, which significantly reduces the memory usage of the model without affecting the accuracy of detection. These enhancements made the model highly efficient on deploying on edged devices, such as drones and agri monitoring systems. By strategically combining DSC pruning and quantizing, we developed a lightweight yet powerful model, especially designed for real-time applications. The DSC ratio can be computed using Eq. (1):

$$DSC\ ratio = \frac{M \times N \times D_k \times D_k}{M \times D_k \times D_k + M \times N} \quad (1)$$

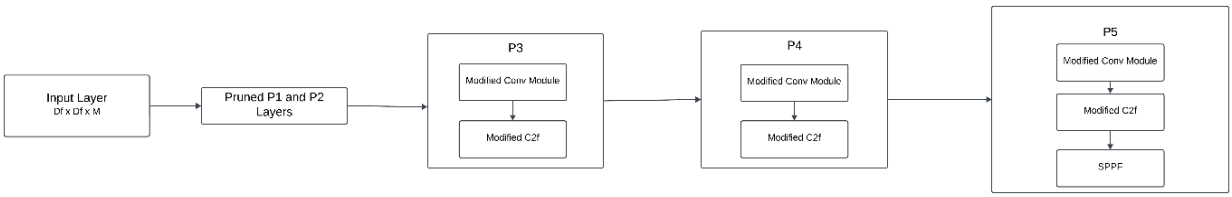

Fig. 4: Modified Backbone of YOLOv8

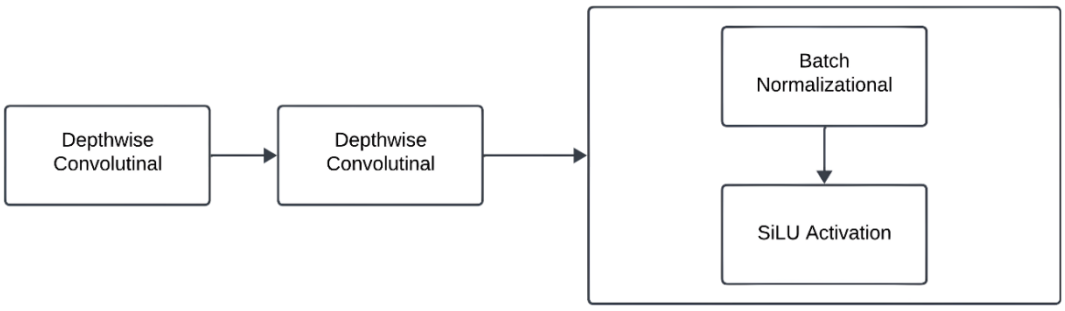

Fig. 5: Modified Convolution Module

Modified C2f Module

To improve the accuracy of pest detection, we enhanced the C2f module, which is essential for integrating high-level features with contextual information. A key modification involved increasing the number of bottleneck layers to five, enabling deeper feature extraction and more effective multi-scale information fusion (Fig. 6). In this structure, one segment of the feature map undergoes multiple bottleneck transformations for thorough processing, while another segment remains unchanged to retain crucial structural details. This upgraded C2f module facilitates a more comprehensive understanding of the detection environment, enhancing the model's ability to accurately identify insects and their bounding boxes. Additionally, by optimizing the integration of high-level features with contextual details, we ensure smoother gradient flow, which is vital for efficient training. These improvements significantly boost the model's performance, particularly on edge devices with limited computational resources, thereby making real-time pest detection more efficient and reliable.

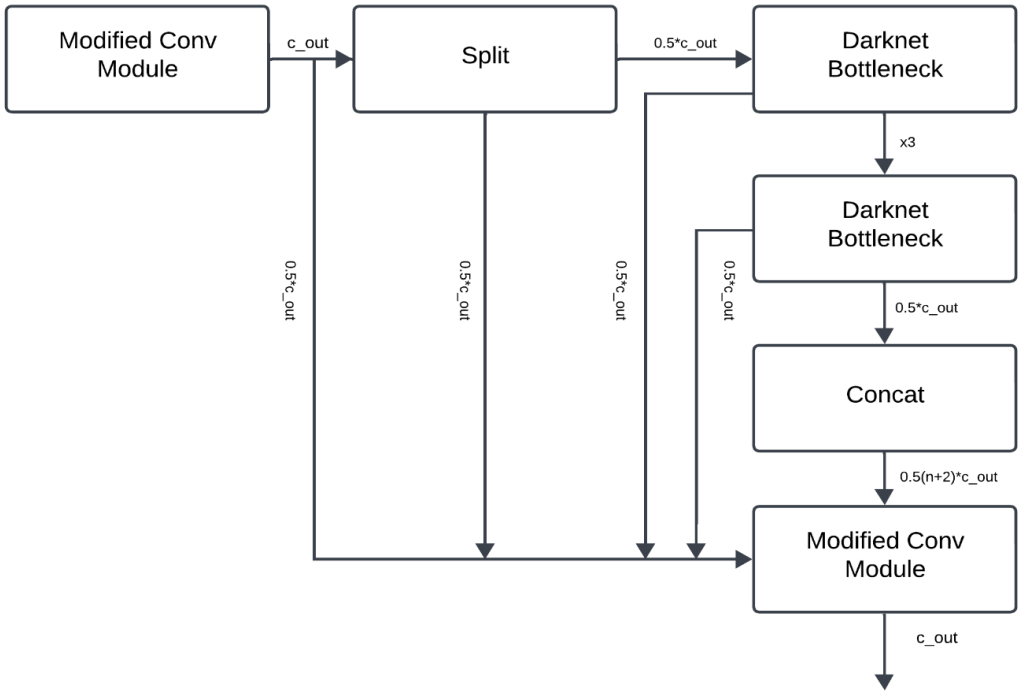

Fig. 6: Modified C2f Module

Results and Discussion

The application of intelligent edge vision systems, deep learning techniques, and optimization with TensorRT can yield significant benefits in pest detection for agricultural farms. The results obtained from such a system can have a substantial impact on agricultural practices and crop management. The optimized model proposed in this work is implemented in an NVIDIA RTX3060 GPU workstation with 12GB of GPU RAM in a Python programming environment with PyTorch and TensorRT libraries. The model was trained for 100 epochs and other hyperparameters as listed in Table 4. The full training was completed using the annotated custom curated dataset of 4.5 gigabytes for a total duration of 13.5 hours, and a PyTorch weights file of size 210 MB was outputted. The PyTorch file was given as input to the ONNX file to produce the TensorRT Inference Engine. The multi-class cross-entropy function is used to arrive at the training loss.

Table 4: List of various Hyperparameters used during the training

Hyperparameters	Values Set
Batch Size	32
Initial Learning Rate (LR)	0.0001
Burn-in Steps (Warm-up)	4000
LR Decay	Cosine
Input Image Size	[227,227]
n (number of bottleneck layers)	3
IoU Threshold	0.5
Confidence Threshold	0.45
Optimization Function	Adam
Epochs	250
Train, Test Split	80:20

Object Localization Loss using Complete Intersection Over Union (CIoU): The bounding box prediction accuracy was determined using Complete Intersection Over Union (CIoU) loss, which calculates the difference in the coordinates of predicted bounding boxes (*b_pred*) and the ground truth bounding boxes (*b_gt*), which are acquired from the curated Insect dataset. The area of intersection is given by A ∩ B = WO x HO, where WO and HO are the minimum values of overlap between the predicted boxes and ground truth boxes in terms of width and height, respectively. Similarly, the Area of Union is given by A ∪ B = Aa + Ab - (A ∩ B), where Aa and Ab are the areas of the predicted box and ground truth box. The final Intersection over Union (IoU) is calculated using the ratio of the Area of Intersection to the Area of Union. The loss function of the model is given in (2).

Confidence Loss: The confidence loss distinguishes between objects present in the image (foreground) and background regions. Confidence for Objects measures the difference between the predicted confidence score and Idle confidence score that contains the Insects:

$$CIoU\ Loss = 1 - IoU + \frac{\rho^2(b_{pred}, b_{gt})}{c^2} + \alpha v \quad (2)$$

$$CIoU\ Loss = DIoU\ + \alpha v \quad (3)$$

For IoU ≥ 0.5, which means overlapping is satisfactory. Now, for more accurate results, we consider this aspect ratio.

Classification Loss: Binary cross-entropy loss is used to quantify the error between the model's predictions and the true class labels. It is calculated as the Loss between the predicted class probabilities and a one-hot encoded vector of the true class of the matched insect class. This loss function penalizes the model for inaccurate classifications among the five classes of insects. The different loss values obtained during the training and validation phase of the model are presented in Figure 7.

The training and validation loss graphs in Figure 7 provide invaluable insights into the performance, generalization, and convergence of the proposed Intelligent edge vision system for insect detection in agricultural farms. Sample training batches with Insects at different scales, occlusion is presented in Figure 8, and a sample test image set is available in Figure 9. These graphs offer a visual representation of the model's training progress, showcasing how various components such as box score, objectness score, and classification score evolve over 100 epochs during the training process. Similarly, the validation graphs provide a means to evaluate the model's performance on unseen data, offering a glimpse into its generalization capabilities.

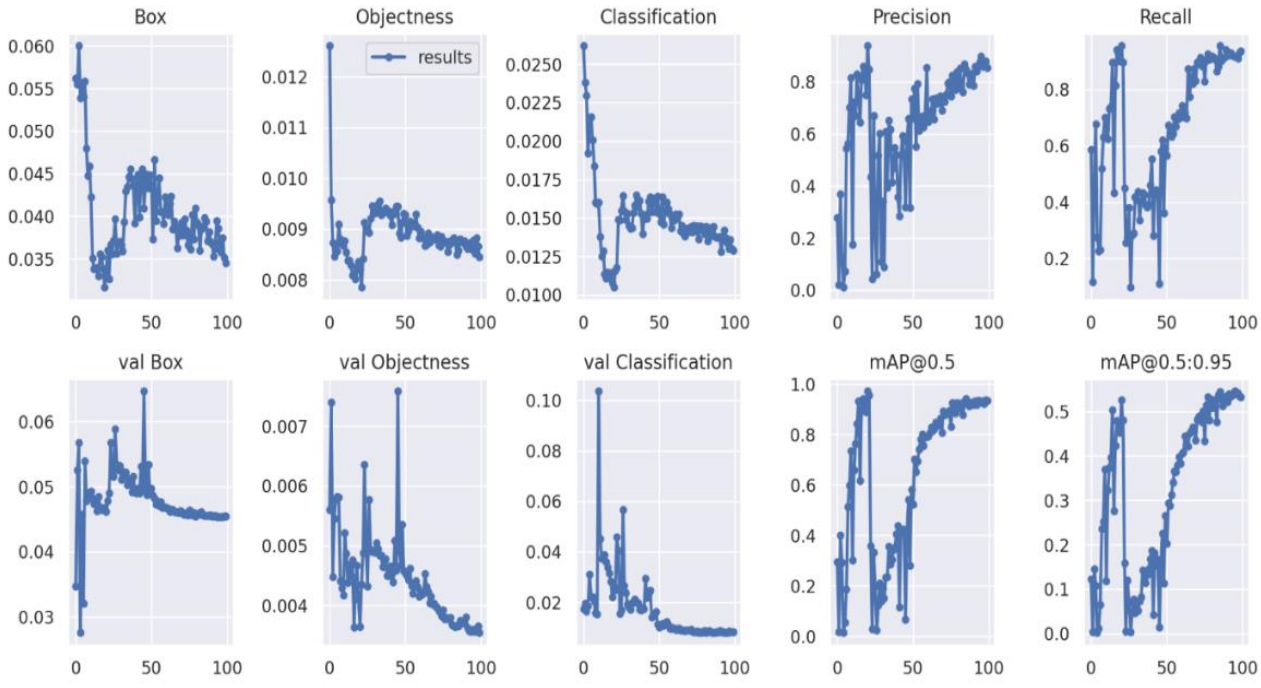

Fig. 7: Training (Top row) and Validation loss (Bottom row) for the optimized YOLOv8 model with the augmented Insects dataset

Fig. 8: Training samples of Insects at different scales, with occlusion and multiple insects within the same Image sample

Fig. 9: Results of test cases with Predicted labels (left) and True labels (right)

Performance Evaluation

The custom-trained model's performance is quantified using the Precision, Recall, F1-score metrics, and the Confusion matrix. The results of the various performance metrics are depicted in Figure 10.

The F1 score graph demonstrates the harmonic mean between precision and recall, offering a comprehensive view of the model's overall performance in insect detection within agricultural farms using Intelligent Edge Vision Systems. F1-score provides a balanced assessment, taking into account both the ability of the model to correctly identify insects (precision) and its capability to capture all instances of insects (recall).

The comparative analysis of various models in the literature with our proposed model is presented in Table 5. The proposed models' F1-score made a striking balance between the precision and recall metrics for all five classes. It is also worth mentioning that the class stink bug underperforms due to the model's inability to separate the background of the image from the foreground image of the stink bug insect. This particular class of insect, the 'stink bug', has pulled down the overall performance of the model, whereas the rest of the insect classes are being classified by the model well. This has also been evident from the Precision-Recall Map and Confusion matrix as shown in Figure 11. Similarly, from the precision curve, it is found that the class of stink bug insect's precision values dropped suddenly for a few epochs, and the model regained its learning parameters and uplifted the confidence to reduce the number of false positives for the stink bug insect class.

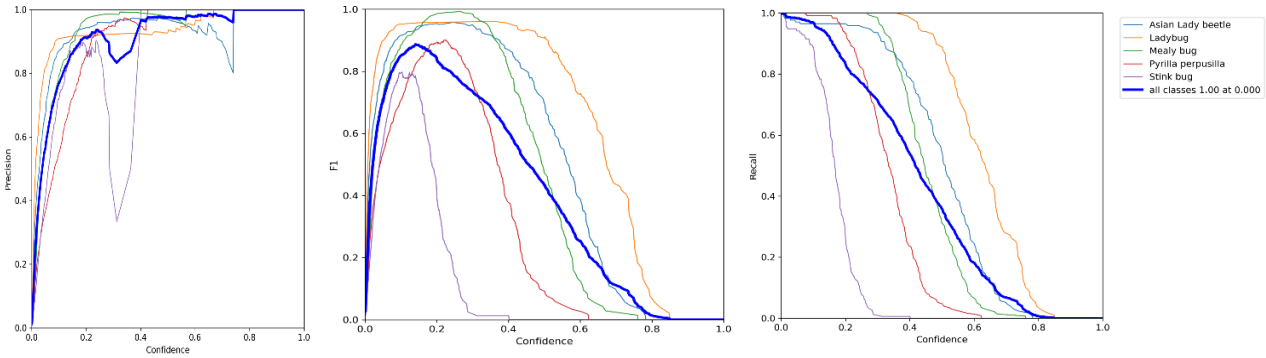

Fig. 10: Precision (Left), F1-Score (Middle) and Recall (Right) metrics of the model

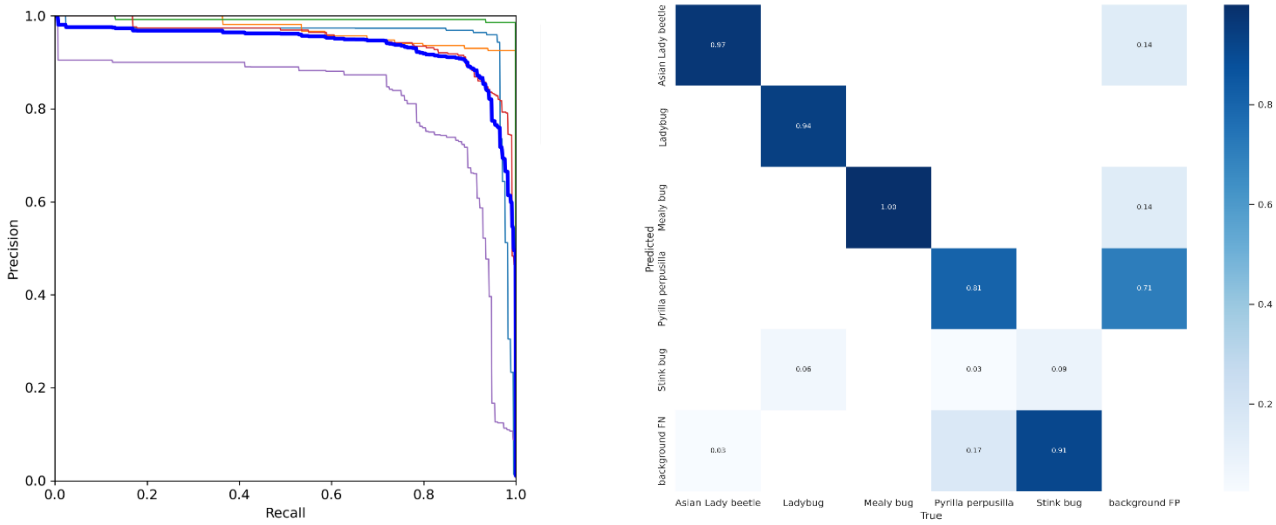

Fig. 11: Precision-Recall Curve (Left) and Confusion Matrix (Right) for the proposed model

Table 5: First 5 losses of CIoU loss function

Epoch	Asian Lady Beetle	Ladybug	Mealy	Pyrilla Perpusilla	Stink Bug
1	0.500	0.436	0.759	0.294	0.348
2	0.961	0.508	0.629	0.952	0.634
3	0.786	0.881	0.448	0.702	0.898
.	.	.	.	.	.
.	.	.	.	.	.
248	0.973	0.262	0.635	0.309	0.226
249	0.971	0.980	0.925	0.960	0.937
250	0.882	0.989	0.699	0.557	0.693

Spanning five distinct classes, including Asian Lady Beetle, Ladybug, Mealy Bug, Pyrilla Perpusilla, Stink Bug, and background (representing false positives), the matrix meticulously dissects the model's performance across these categories. Notably, the YOLOv8 model exhibits remarkable accuracy in identifying Asian Lady Beetle and Ladybug instances, boasting high true positive rates of 0.97 and 0.94, respectively. Similarly, the model demonstrates flawless recognition of Mealy Bug instances, achieving a perfect true positive rate of 1.00. However, the challenge emerges with the Pyrilla Perpusilla and Stink Bug classes. While the model achieves a commendable true positive rate of 0.81 for Pyrilla Perpusilla, it falters in the detection of Stink Bug instances, failing to register any true positives and instead producing a high proportion (0.91) of false positives, indicating misclassification as other classes or background noise. This misclassification is because of the model’s inability to differentiate between the insect class stink bug with the image’s background. These findings

underscore the need for further refinement and optimization of the YOLOv8 model, particularly in improving its ability to accurately identify and classify Stink Bug instances. While the model achieves high accuracy, steps were taken to reduce overfitting, including image augmentation, dropout regularization, and cross-validation. The potential for bias due to dataset imbalance was mitigated by ensuring class representation during training.

Edge Deployment and Inferencing Results

The proposed model is recompiled for deployment at a Jetson Nano 2 GB device with 128 NVIDIA Maxwell GPU cores, which is one of the very optimal and cost-effective edge platforms for optimal inferencing of the proposed vision model in Agricultural farms. TensorRT compiler provided recompilation of the existing PyTorch model with post-training Quantization, Tensor-Layer Fusion to optimize the proposed model for the Jetson Nano architecture, resulting in performance improvements in terms of concurrency rate, increased frames per second, improved latency time with reduction in the memory footprints required for model deployment as stated in Table 6. The model has been recompiled for three different quantization levels, viz FP32, FP16, and INT8, to compare the inference metrics such as Frames per second, Latency time, and reduced memory requirements. The TensorRT library produced two files with respect to the proposed model, viz. Weights file (.wts) and Engine file (.engine). The .wts file stores the INT 8 quantized weights of the model. The .engine file is the optimized, executable model used for inference on the Jetson Nano with TensorRT. The engine incorporates the information from the .wts file along with additional optimizations such as Layer-Tensor fusion and kernel selection. In the proposed work, five consecutive layers of the convolution function to c2f module are fused to reduce the number of tensor operations, thereby the memory footprints are reduced.

Table 7: Comparative Analysis of various parameters after deployment and Inferencing at Jetson Nano

Model Type	Quantization	Model File Size	Latency	Inferencing Rate (FPS)	Libraries
Pytorch (.pth)	FP32	85.1 MB	42.8 ms	28 fps	CUDA 12
Weights (.wts)	FP16	46.8 MB	18.5 ms	40 fps	TensorRT 8
Engine file (.engine)	INT 8 + Layer-Tensor Fusion	21.5 MB	9.2 ms	65 fps	TensorRT 8

Conclusion

In the proposed work, an Intelligent Edge vision system for insect detection in agricultural farms using YOLOv8 models, TensorRT compiler, and Nvidia Jetson Nano is presented. The work substantiates the optimized YOLOv8 model for deployment at the Jetson Nano device using INT8 quantization and LT fusion, which has been proven to be significant for the NBAIR Insect dataset. It has been inferred that serious Image Augmentation techniques improved the performance of the model in terms of Precision and Recall parameters. The post-training recompilation has significantly reduced the model size through INT8 quantization and Layer-Tensor Fusion. These two model optimization techniques yielded better results in terms of reduced Model size, Latency time, and improved FPS rate. The work also observed that the c2f module in the YOLOv8 models and their Tensor's fusion using TensorRT compiler has made the model compression become a practical solution for deployment at Jetson Nano. However, the model failed to learn the distinguishing features of the stink bug, primarily due to low intensity contrast between the insect and its background, which hindered effective feature extraction. The work can be further directed towards preparing an optimized object classification model for fusing multi-modal image data at night and under different lightning conditions at the Agricultural farms. Future work may include integration with drone-based imaging and model refinement using domain adaptation techniques for varied crop types.

Acknowledgment

We sincerely thank the management SRM Institute of Science and Technology Tiruchirappalli for their support and resources. We also acknowledge the valuable inputs from agricultural experts and farmers during data collection.

Funding Information

This research was funded by the authors on their own.

Author's Contributions

Deebalakshmi R.: Conducted performance analysis, benchmarking, and statistical validation.

Adhithyaa S.: Performed the literature review, updated references, and managed formatting and journal compliance.

Yoga Vignesh V.: Developed the methodology, algorithms, and system architecture.

Balaji Ganesh Rajagopal: Drafted the manuscript and coordinated revisions and submission. All authors reviewed and approved the final manuscript.

Ethics

This article does not contain any studies with human participants or animals performed by any of the authors. All procedures were conducted in an ethically responsible manner, following all relevant institutional and national regulations. The research adhered to the principles of honesty, integrity, and objectivity in data collection and reporting.

References

Ademola, O. A., Eduard, P., & Mairo, L. (2022). Ensemble of Tensor Train Decomposition and Quantization Methods for Deep Learning Model Compression. *2022 International Joint Conference on Neural Networks (IJCNN)*, 1–6. https://doi.org/10.1109/ijcnn55064.2022.9892626

Albanese, A., Nardello, M., & Brunelli, D. (2021). Automated Pest Detection With DNN on the Edge for Precision Agriculture. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *11*(3), 458–467. https://doi.org/10.1109/jetcas.2021.3101740

Albanese, A., Nardello, M., & Brunelli, D. (2022). Low-power deep learning edge computing platform for resource constrained lightweight compact UAVs. *Sustainable Computing: Informatics and Systems*, *34*, 100725. https://doi.org/10.1016/j.suscom.2022.100725

Ali, F., Qayyum, H., & Iqbal, M. J. (2023). Faster-PestNet: A Lightweight Deep Learning Framework for Crop Pest Detection and Classification. *IEEE Access*, *11*, 104016–104027. https://doi.org/10.1109/access.2023.3317506

Anwar, Z., & Masood, S. (2023). Exploring Deep Ensemble Model for Insect and Pest Detection from Images. *Procedia Computer Science*, *218*, 2328–2337. https://doi.org/10.1016/j.procs.2023.01.208

Baghbanbashi, M., Raji, M., & Ghavami, B. (2023). Quantizing YOLOv7: A Comprehensive Study. *2023 28th International Computer Conference, Computer Society of Iran (CSICC)*, 1–5. https://doi.org/10.1109/csicc58665.2023.10105310

Cardoso, B., Silva, C., Costa, J., & Ribeiro, B. (2022). Internet of Things Meets Computer Vision to Make an Intelligent Pest Monitoring Network. *Applied Sciences*, *12*(18), 9397. https://doi.org/10.3390/app12189397

Chithambarathanu, M., & Jeyakumar, M. K. (2023). Survey on crop pest detection using deep learning and machine learning approaches. *Multimedia Tools and Applications*, *82*(27), 42277–42310. https://doi.org/10.1007/s11042-023-15221-3

Diya, V. A., Nandan, P., & Dhote, R. R. (2023). IoT-based Precision Agriculture: A Review. *Proceedings of Emerging Trends and Technologies on Intelligent Systems*, 373–386. https://doi.org/10.1007/978-981-19-4182-5_30

Domingues, T., Brandão, T., Ribeiro, R., & Ferreira, J. C. (2022). Insect Detection in Sticky Trap Images of Tomato Crops Using Machine Learning. *Agriculture*, *12*(11), 1967. https://doi.org/10.3390/agriculture12111967

Dong, Q., Sun, L., Han, T., Cai, M., & Gao, C. (2024). PestLite: A Novel YOLO-Based Deep Learning Technique for Crop Pest Detection. *Agriculture*, *14*(2), 228. https://doi.org/10.3390/agriculture14020228

Droukas, L., Doulgeri, Z., Tsakiridis, N. L., Triantafyllou, D., Kleitsiotis, I., Mariolis, I., Giakoumis, D., Tzovaras, D., Kateris, D., & Bochtis, D. (2023). A Survey of Robotic Harvesting Systems and Enabling Technologies. *Journal of Intelligent & Robotic Systems*, *107*(2), 21. https://doi.org/10.1007/s10846-022-01793-z

Huang, R., Pedoeem, J., & Chen, C. (2018). YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers. *2018 IEEE International Conference on Big Data (Big Data)*, 2503–2510. https://doi.org/10.1109/bigdata.2018.8621865

Kang, H., Ai, L., Zhen, Z., Lu, B., Man, Z., Yi, P., Li, M., & Lin, L. (2023). A Novel Deep Learning Model for Accurate Pest Detection and Edge Computing Deployment. *Insects*, *14*(7), 660. https://doi.org/10.3390/insects14070660

Kargar, A., Wilk, M. P., Zorbas, D., Gaffney, M. T., & Q'Flynn, B. (2022). A Novel Resource-Constrained Insect Monitoring System based on Machine Vision with Edge AI. *2022 IEEE 5th International Conference on Image Processing Applications and Systems (IPAS)*, 1–6. https://doi.org/10.1109/ipas55744.2022.10052895

Makkena, Y. C., Tella, R. R., Parekh, N., Saraf, P. K., Annu, Shukla, H., Matathammal, A., Danda, S. C. S., Chandrahas, P., Jadhav, A. R., Tammana, P., Kondepu, K., & Pachamuthu, R. (2023). Experience: Implementation of Edge-Cloud for Autonomous Navigation Applications. *2023 15th International Conference on COMmunication Systems & NETworkS (COMSNETS)*, 579–587. https://doi.org/10.1109/comsnets56262.2023.10041370

Moradeyo, O. M., Olaniyan, A. S., Ojoawo, A. O., Olawale, J. A., & Bello, R. W. (2023). YOLOv7 Applied to Livestock Image Detection and Segmentation Tasks in Cattle Grazing Behavior, Monitor and Intrusions. *Journal of Applied Sciences and Environmental Management*, *27*(5), 953–958. https://doi.org/10.4314/jasem.v27i5.10

Rajak, P., Ganguly, A., Adhikary, S., & Bhattacharya, S. (2023). Internet of things and smart sensors in agriculture: Scopes and challenges. *Journal of Agriculture and Food Research*, *14*, 100776. https://doi.org/10.1016/j.jafr.2023.100776

Sadowski, S., & Spachos, P. (2018). Solar-Powered Smart Agricultural Monitoring System Using Internet of Things Devices. *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, 18–23. https://doi.org/10.1109/iemcon.2018.8614981

Shi, Y., Yang, K., Yang, Z., & Zhou, Y. (2022). *Model compression for on-device inference*. 71–82. https://doi.org/10.1016/b978-0-12-823817-2.00015-2

Shin, H., & Oh, H. (2022). Neural Network Model Compression Algorithms for Image Classification in Embedded Systems. *Journal of Korea Robotics Society*, *17*(2), 133–141. https://doi.org/10.7746/jkros.2022.17.2.133

Shruthi, U., Nagaveni, V., & Raghavendra, B. K. (2019). A Review on Machine Learning Classification Techniques for Plant Disease Detection. *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 281–284. https://doi.org/10.1109/icaccs.2019.8728415

Sushma, D. S., Mohammed, A., Aravind, M., Jayanth, A. B., & Rakshith, K. K. (2024). Pest Detection and Classification in Peanut Crops. *International Research Journal on Advanced Engineering and Management (IRJAEM)*, *2*(05), 1372–1379. https://doi.org/10.47392/irjaem.2024.0189

Tang, Y., & Zhang, S. (2023). A Smart Real-Time Monitoring Method of Vegetable Diseases and Insect Pests Based on Optical Fiber Sensor. *Journal of Testing and Evaluation*, *51*(3), 1277–1294. https://doi.org/10.1520/jte20220052

Tetila, E. C., Brandoli Machado, B., Menezes, G. V., de Souza Belete, N. A., Astolfi, G., & Pistori, H. (2020). A Deep-Learning Approach for Automatic Counting of Soybean Insect Pests. *IEEE Geoscience and Remote Sensing Letters*, *17*(10), 1837–1841. https://doi.org/10.1109/lgrs.2019.2954735

Venkatasaichandrakanthand, P., & Iyapparaja, M. (2023). Pest Detection and Classification in Peanut Crops Using CNN, MFO, and EViTA Algorithms. *IEEE Access*, *11*, 54045–54057. https://doi.org/10.1109/access.2023.3281508

Wang, W., & Fu, H. (2024). A Lightweight Crop Pest Detection Method Based on Improved RTMDet. *Information*, *15*(9), 519. https://doi.org/10.3390/info15090519

Wang, X., Zhang, S., Wang, X., & Xu, C. (2023). Crop pest detection by three-scale convolutional neural network with attention. *PLOS ONE*, *18*(6), e0276456. https://doi.org/10.1371/journal.pone.0276456

Yang, Z., Feng, H., Ruan, Y., & Weng, X. (2023). Tea Tree Pest Detection Algorithm Based on Improved Yolov7-Tiny. *Agriculture*, *13*(5), 1031. https://doi.org/10.3390/agriculture13051031

Yesuf, M. M., & Assefa, B. G. (2023). Model Compression Techniques in Deep Neural Networks. *Pan-African Conference on Artificial Intelligence*, 169–190. https://doi.org/10.1007/978-3-031-31327-1_10