International Journal of Artificial Intelligence Trends (IJAIT)
Vol. 5, Issue VI; No. 69, June, 2026, pp. 700-715

DEVELOPMENT OF A PEST MANAGEMENT SYSTEM FOR PRECISION AGRICULTURE USING TRANSFER LEARNING AND INTERNET OF THINGS (IOT)

[1]Nnaemeka Frances O.,[2]Ogochukwu Okeke
Department of Computer Science.
Chukwuemeka Odumegwu Ojukwu University, Uli.
Email: [1]obynnaemekaf@gmail.com [2]ogoookeke@yahoo.com

Article Info
Received: 18/11/ 2025
Revised: 12/5/2026
Accepted 3/6/2026
Corresponding Authors
[1*]Email:
obynnaemekaf@gmail.com

Corresponding Author's Tel:
[1*]+2348064778661

ABSTRACT
The aim of this study is design and implementation of pest management system for precision agriculture using integrated transfer learning and internet of things technique. The methodology used is dynamic system development model. The tested for primary data collection is Aninri and the pest considered are weevil, caterpillar, and whitefly. The secondary data source is Kaggle. Then total sample size of data collected is 18138. The data was processed through annotation and labeling using Robowflow tool. The benchmark transfer learning model used is You Only Look Once (YOLOv10), which was improved through the optimization of the spatial pyramid pooling function using average pooling layer and maximum pooling layer. Python programming language was applied to train the model and generate the real-time pest classification solution. Training of the model was done experimentally, considering the benchmark model and then the improved YOLOv10. From the results, it was observed that precision of the model with YOLOv10 recorded 0.88519 while the recall recorded 0.84463. The mAP@0.5 recorded 0.82746 and 0.58272 for the traditional YOLOV-10, while for the new model the precision recorded 0.97519, recall reported 0.93163 and mAP@0.5 recorded 0.92486 and 0.67272 respectively. From these results, it was observed that the new YOLOV-10 in all metrics supersedes the traditional models and the reason was due to the optimization of the SPPF layer which increased the quality and quantity of features extracted, thereby optimizing the model performance in correctly pest detection in the farm.

Keywords: Precision Agriculture; Pest Management; Transfer Learning; IoT; YOLOv10

1. INTRODUCTION

"In the fertile field of agriculture, pests emerge as unwelcome guests, arriving uninvited to disrupt the natural balance of the fields. However, the seasoned farmer, armed with knowledge and patience, turns these intruders into invaluable lessons. They cultivate a symphony of resilience and bounty amidst the ever-changing rhythms of nature". Cheng et al. (2017) defined pests as a range of organisms, such as insects, rodents, fungi, bacteria, viruses, and more, that inflict damage on the health and growth of plants. According to Chen et al. (2020), these pests represent a significant challenge impacting the successful yield of agricultural products worldwide.

According to estimations from the Food and Agriculture Organization (FAO), pests cause an astounding 40% loss in worldwide food production each year (Karar et al., 2021). For this reason, pest management is an important task that forms the basis of this study. Generally speaking, weeds (Gao et al., 2018), insects (Aphids, beetles, and caterpillars), diseases (fungi, bacteria, and viruses), and vertebrates (birds, rodents, and herbivorous animals) are the four main types of pests that Chen et al. (2020) classified with examples. When taken as a whole, Kumar et al. (2017) found that these pests have serious negative consequences on agriculture, including crop losses from damage that lowers yield, negative economic effects on farmers who may suffer financial losses from the high cost of pest control, and decreased productivity.

Additionally, the impact of pests on plants can lead to a shortage of food, as well as negative environmental effects from the spread of pesticides and pest management measures (Mendoza et al., 2023). Furthermore, these pests can spread quickly throughout the farm, meaning that by the time farmers realize they have a problem, the effects have already

become severe and, for the most part, uncontrollable (Johnson, 2020). Conventional methods for controlling pests include early pest discovery, followed by the use of a control measure. Using drones, robots, and satellites for remote sensing, pheromone traps (which trap pests using chemicals), visual inspection (which involves the farmer performing a human check), and the use of smart sensors (special motion sensors, infrared sensors) to identify the presence of pests on the farm (Kumar et al., 2017).

The control strategies include using pesticides, which are chemicals applied to kill or repel pests; biological control methods, which involve introducing predators to reduce the pest population naturally; cultural control methods, which involve applying crop rotation and intercropping practices; and, lastly, mechanical control, which involves applying physical barriers to prevent pest access to the farm (Kumar et al., 2017;Johnson, 2020;Srivastava and Mishra, 2018). Although there has been some success with these methods, Banga et al. (2018) found that these conventional methods of pest control are labor-intensive, slow, subjective, and damaging. Hence, there is a need for a more effective pest control solution.

Artificial intelligence (AI) has garnered increased interest recently for its potential to support automated monitoring and precision agriculture. While automated monitoring suggests continuous farm monitoring to offer real-time information on pest activity, precision agriculture optimizes resources, such as insect presence on the farm, for precise pesticide administration (Mendoza et al., 2023). In order to create pest detection models, studies like Prasath and Akila (2023),Alanazi et al. (2023),Anwar and Masood (2023), and Karar et al. (2022) used YOLOV-3, Residual Network (ResNet), and Visual Geometry Group (VGG). Although the accuracy of the results for all models was over 96%, there was still a gap in the problem's control solution. (Vemuri, 2023;Thomas et al., 2023) filled this gap by incorporating Wi-Fi (Bluetooth) and machine learning algorithms for pest detection and control. Although the outcomes showed a remarkable success rate, the ongoing difficulty is caused by the pests' dynamic and ever-changing nature.

According to Emeric et al. (2022), the existing machine learning approach may not be able to sufficiently handle the complexities introduced by the variances in pest species across different locations, such as pests in Europe differing from those in Asia, Africa, etc. The sorts of pests that thrive in different geographical places vary due to factors such as climate, ecosystem diversity, and agricultural techniques (Emeric et al., 2022). Temperature, humidity, and the presence of particular host plants are examples of factors that might affect the types and prevalence of pests in a given area (Cammell et al., 1992). As a result, the current A.I. model may have problems or false positives. Thus, a sophisticated model that is trained to identify various pests on a farm while taking into account different regions is required. This model should also be able to apply intelligent control measures that prompt responses to problems for prompt management.

1. RESEARCH METHODOLOGY

The methodology to be used for this work is the dynamic system development model. The reason for choosing this methodology is that it allows for rapid application development through an iterative process of prototyping and user feedback. This methodology is well-suited for projects with changing requirements, as it enables the development team to adapt quickly to new information or changes in scope. By breaking the project down into smaller, manageable pieces and incorporating user input at each stage, the dynamic system development model helps ensure that the final product meets the needs of the stakeholders and end-users. The Figure 1 presents the architecture of the proposed system.

International Journal of Artificial Intelligence Trends (IJAIT)
Vol. 5, Issue VI; No. 69, June, 2026, pp. 700-715

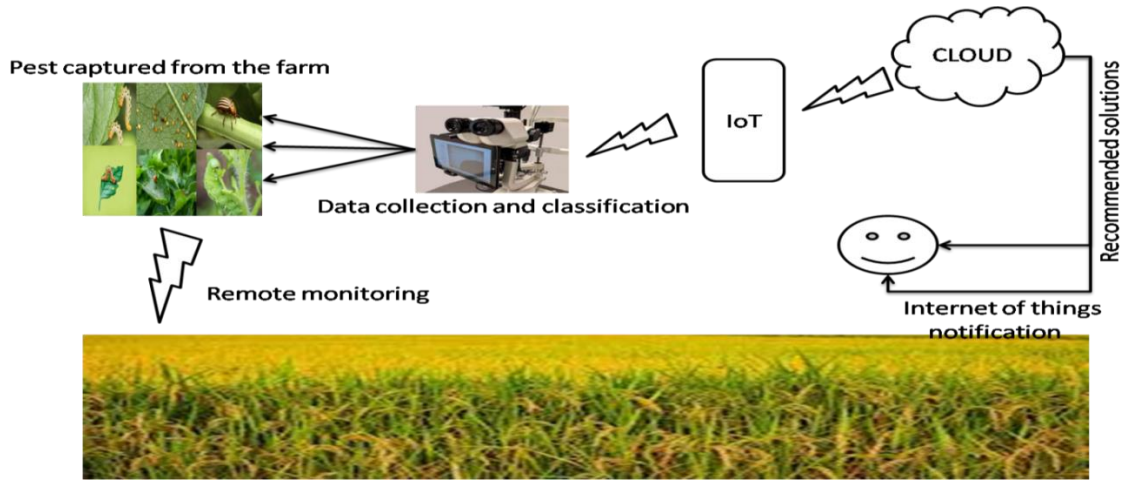

Figure 1: Architecture of the proposed system

The proposed system when situated in the farm through a camera, collected data of pest and then classify using the trained YOLOV-10 model with decision-based algorithm. The IoT algorithm was applied to notify the farmer of the classification output through email notification and also attached to the notification email is a recommended pesticide suitable to management the impact of every particular pest classified on the farm.

1.1 Data Collection

The data used for this work was collected from three farms at Ndeabo in Aninri local Government Area, Enugu State. The geographical coordinate of the site is at Latitude 6.01.30N and Longitude 7.34.30E. The instruments for the data collection are HD USB camera, raspberry pi, and Laptop to collect the files. The results obtained for the data collection are reported in Figure 2.

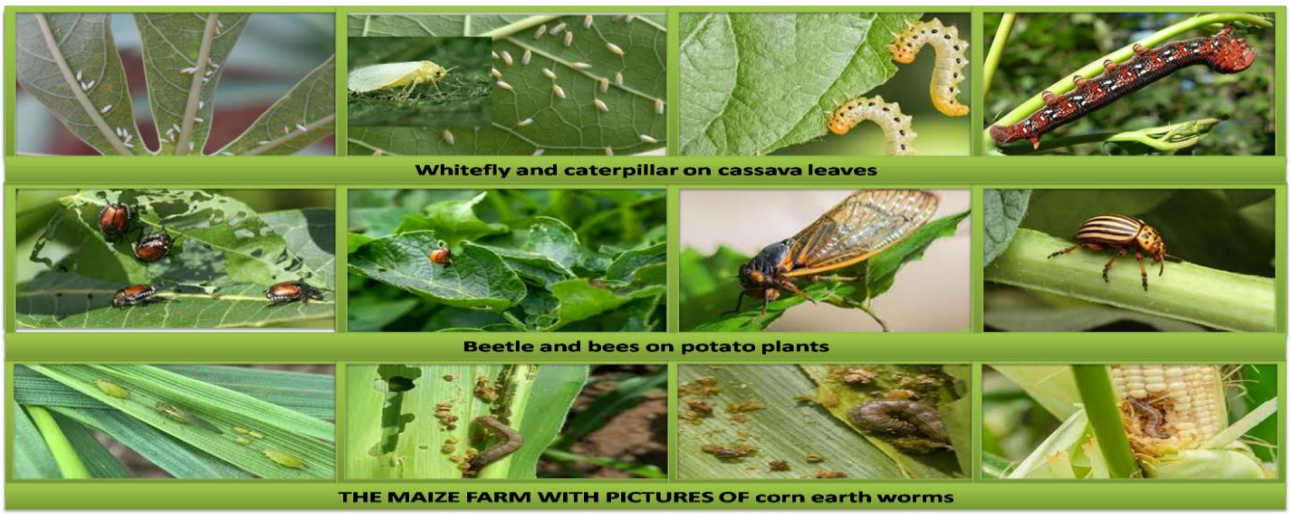

Figure 2: The primary testbed for data collection

Figure 2 illustrates the primary testbed used for the initial data collection phase in the smart farm environment. This setup included cameras strategically mounted within farm plots to capture real-time images of pest activity on crops. The testbed was designed to simulate actual farm conditions, ensuring that the images reflected the natural habitat and behaviours of pests as observed in local agricultural settings. Once the images were collected, they were organized and processed through a structured dataset. As depicted in Figure 3, the raw images were extracted and pre-processed to facilitate manual annotation and labelling. This step was crucial for training machine learning models, as it provided accurate bounding boxes and class labels corresponding to various pest species. The total number of images collected from the primary source was 497 pest images. While these images served as a reliable starting point, the relatively small

dataset posed limitations in terms of model generalizability and performance. To enhance the robustness of the dataset and ensure better model training, a secondary dataset was incorporated from Kaggle, a well-known open data platform. The secondary dataset, titled "Insect Pest Detection for YOLO", provided a comprehensive collection of 17,641 labelled images featuring a wide variety of pests and insects commonly found in agricultural environments. This dataset significantly enriched the diversity and quantity of training data, helping to improve the model's ability to detect and classify pests across different farm settings. The data source is https://www.kaggle.com/datasets/cubeai/insect-pest-detection-for-yolo. By combining locally captured data with a publicly available the total sample size is 18138 images of pest. This forms a strong foundation for training the deep learning model. The data samples after annotation and labelling are reported in Figure 3 and 4.

Figure 3: Pest data from secondary source (Source: Kaggle)

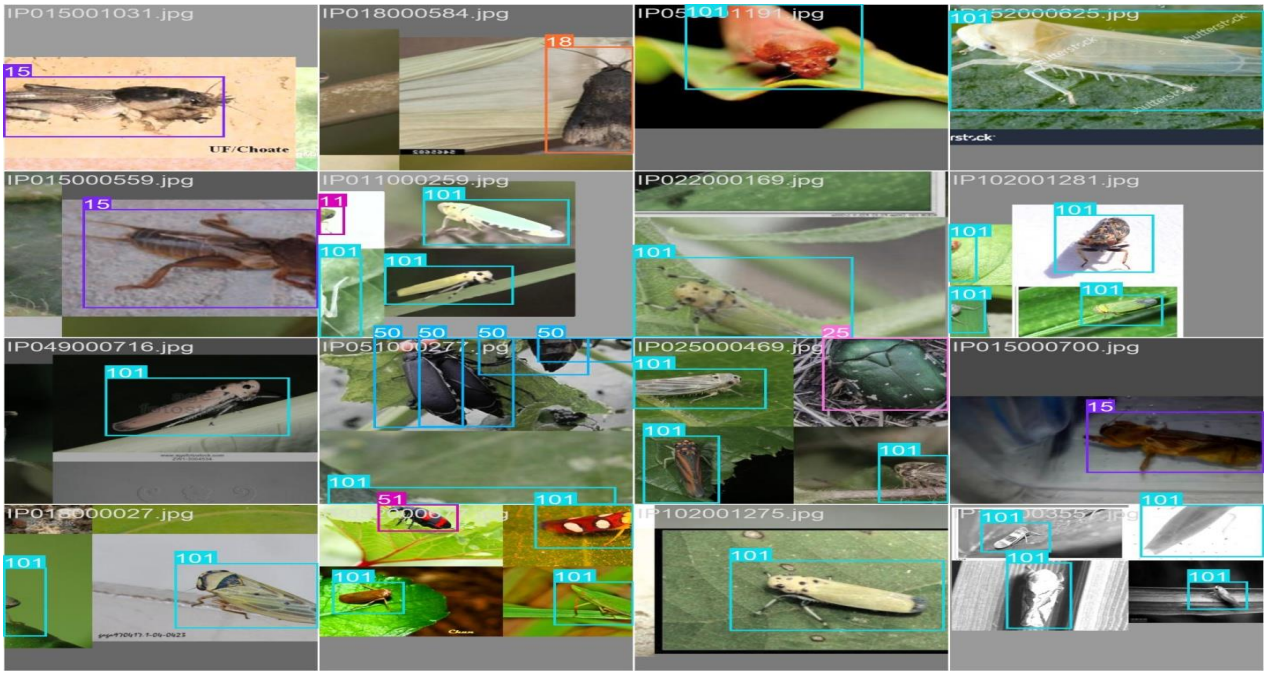

Figure 4: Data annotation images

2. MODEL OF THE PEST MANAGEMENT SYSTEM

The pest management system is made of the transfer learning model, the IoT model and the control model. The transfer learning model is the YOLOV-10. The IoT model used SMTP while the control model is actions necessary to mitigate the impact of the pest on the farms.

2.1 Model of the transfer learning technique

The transfer learning technique adopted for this work is the YOLOV-10. The model is made of three main sections which are the backbone, the neck and the head. The back bone accepts images from the images dimensioned and then through the application of Cross Stage Partial with 2 bottlenecks (C2F) layer, convolutional layer, C2F-Convolutional Inverted Bottleneck (CIB) C2FCIB layer, Spatial Channel Decoupled Downsampling (SCDown), Spatial Pyramid Pooling Function (SPPF) and Parallel Self Attention (PSA), extracts the information, concatenate in the neck and then classify in the output. The reason why YOLOV-10 was adopted was due to its ability for small object detection, making it very suitable for pest classification. The architectural model of the YOLOV-10 is in Figure 5.

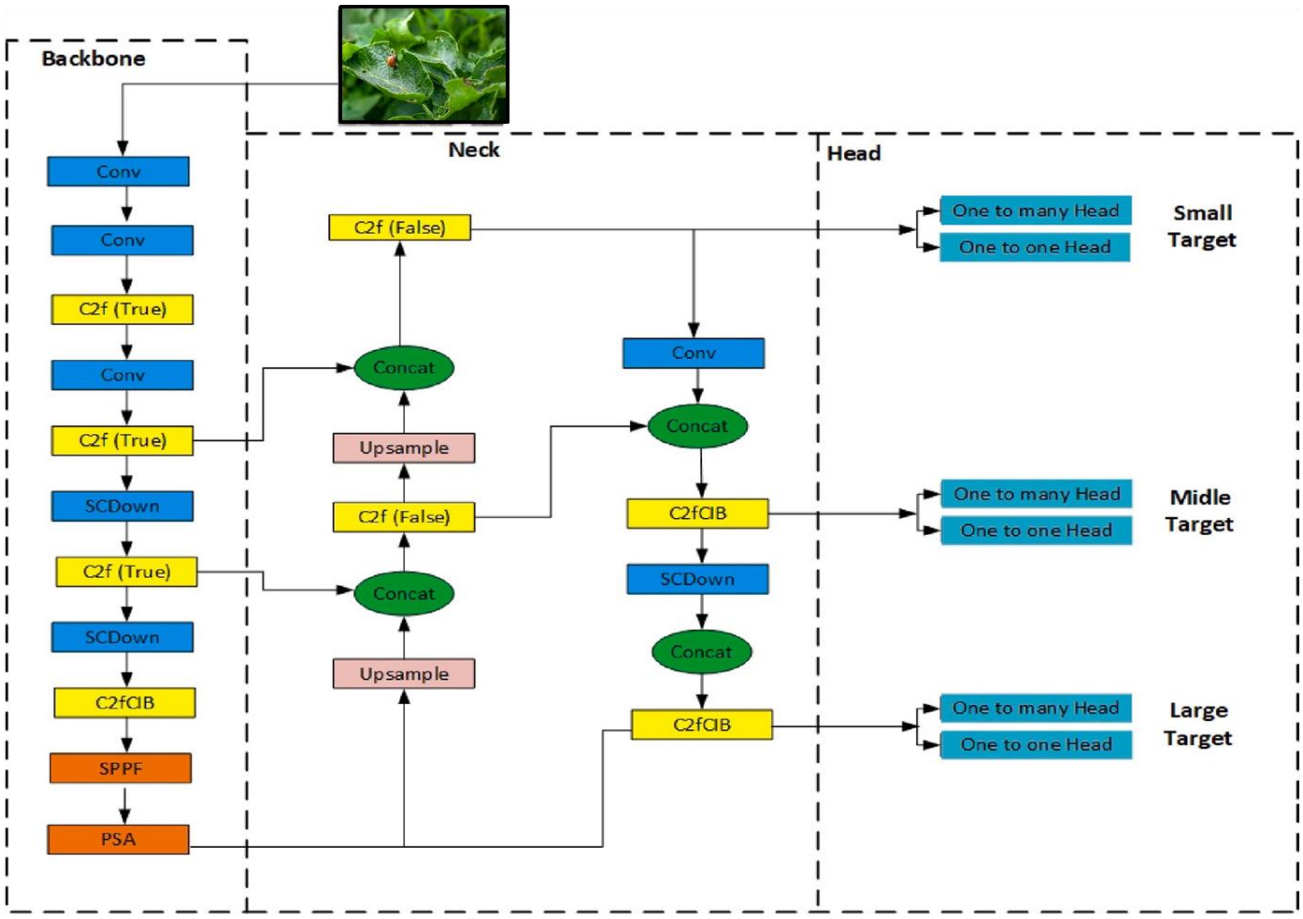

Figure 5: Architecture of the YOLOV-10

The Figure 5 presented the model of the YOLOV-10. The C2F layer contains two bottleneck, convolutional layer and concatenation layer. The bottleneck allows balance between learning nee features and reusing existing one. Each bottle neck is made of convolutional layers connected in series. For the C2F (True) layer, the bottleneck is connected in series and also has additional channel from the input to the output. For the C2F (False), the bottle neck has only two convolutional layers connected in series. The convolutional layers allow for feature convolution with filters to extract local image information. The C2FCIB is an improved C2F layer using CIB. This section expands feature dimension and facilitate more abstract representation of features, while maintaining computational. The SPPF layer contains convolutional layer, concatenation layer and maximum pooling layers interconnected as shown in Figure 6.

The role is to maximize feature extraction in multiscale and varying sizes. The PSA is an attention mechanism made of Multi Head Self Attention (MSHA) and Feed forward Neural Network (FFN), which collectively allows the model to have focus on more important features through localization of spatial regions in the image strides. This helps in the capturing of long-range dependencies and optimizes model ability in small object detection like pest in 11 dimensions. The SCDown first increase the channel dimension with point wise 11 convolution, then spatial downsampling with 33 depth wise convolution to reduce the spatial dimension of each feature. Overall the SCDown expands number of channels through downsampling.

2.2 Improving the YOLOV-10

This section is tailored towards improving the YOLOV-10 model to be compatible with the tiny sized of pest to facilitate improved detection. In achieving this, a critical look at the architecture was carried out and SPPF as considered as an area to improve to facilitate enhanced feature extraction process. Although SPPF improves efficiency, the correlation between features is weak due to its direct splicing of feature maps after separating and processing, which limits its feature expression ability (Zhang et al., 2025).The proposed SPPF employed average and maximum pooling layer as shown in Figure 6 to improve feature extraction process and address information loss challenges.

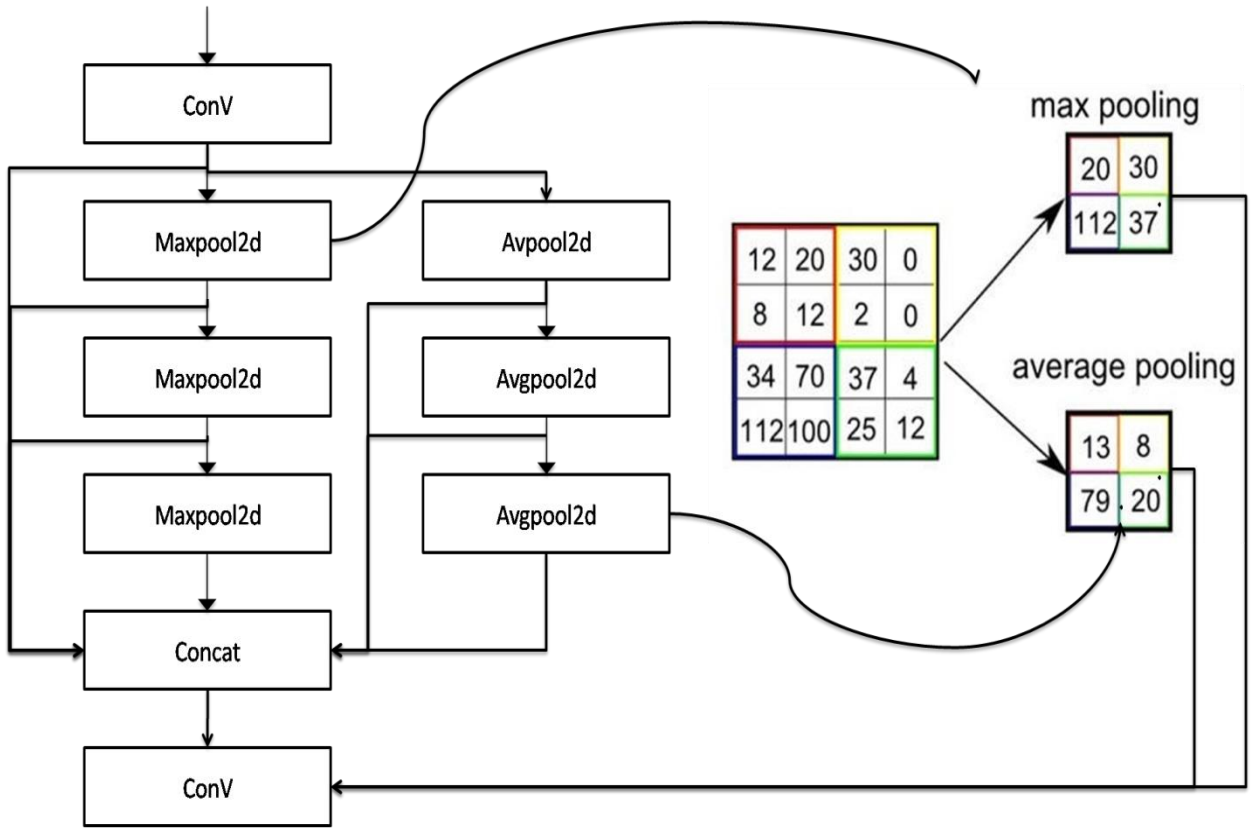

Figure 6: Proposed SPPF Module

Figure 6 illustrates the architecture of the proposed Spatial Pyramid Pooling Fast (SPPF) module integrated within the YOLOv10 detection framework. This enhanced SPPF design incorporates both average pooling and max pooling operations, applied in parallel across multiple receptive fields to independently capture complementary spatial and contextual features. The pooled outputs are then concatenated along the channel dimension, enabling the aggregation of both fine-grained and dominant spatial cues. This dual-path pooling strategy enriches the feature representation by increasing the diversity of extracted descriptors, thereby enhancing the model's ability to detect small-scale and densely packed objects such as agricultural pests. The final concatenated tensor is passed through a 1×1 convolutional bottleneck

layer to reduce dimensionality and improve computational efficiency, while preserving salient information necessary for robust object localization and classification. Figure 7 present the improved YOLOV-10 architecture.

2.3 Training of the Improved YOLOV-10

The training process of the YOLOv10 model for pest detection was conducted using a hybrid dataset comprising 497 custom-captured pest images and an additional 17,641 annotated pest samples sourced from the public Kaggle repository. The combined dataset was pre-processed and formatted to comply with the YOLO training pipeline using Roboflow for annotation standardization, bounding box formatting (in YOLO txt format), and class label encoding. The training was carried out using the YOLOv10 architecture, initialized with pre-trained weights from the COCO dataset to applied transfer learning and expedite convergence. The dataset was split into 70% training, 20% validation, and 10% testing to ensure proper generalization and to avoid overfitting. During training, a multi-scale training regime was employed, wherein the input resolution was randomly varied between 640×640 and 1280×1280. This approach enabled the model to generalize better across different pest sizes, particularly aiding in the detection of tiny or partially occluded pests. The loss function was composed of three components: CIoU loss for bounding box regression, object loss for presence confidence, and focal loss for classification, with carefully tuned weights to prioritize precision over recall in a densely populated pest scenario. The Adam optimizer with weight decay was used for training, alongside a cosine learning rate scheduler to allow smooth convergence. Training was conducted over 300 epochs on an NVIDIA GPU platform, with check-pointing and early stopping mechanisms based on validation mAP (mean Average Precision) to ensure optimal performance and prevent overfitting.

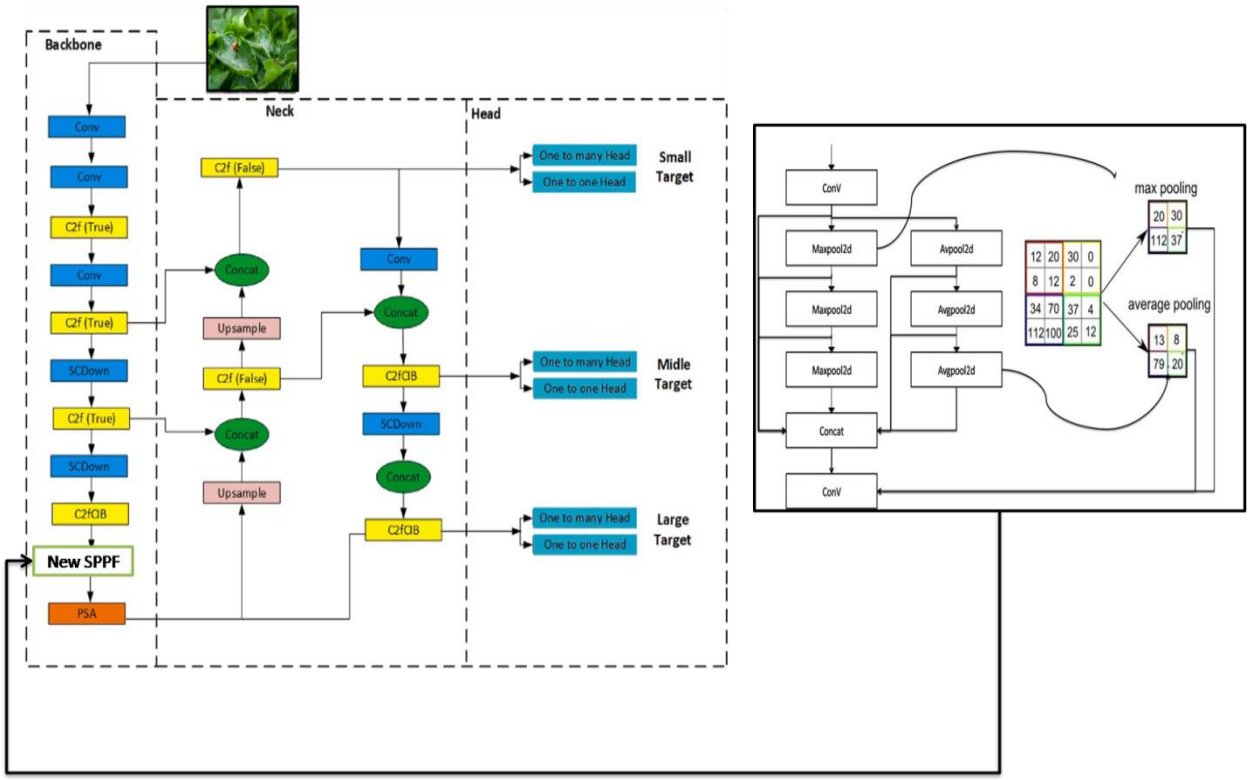

Figure 7: Improved YOLOV-10 Architecture

2.4 System Flowchart

This section presents the flowchart of the YOLOv10 model enhanced with an Improved Spatial Pyramid Fast Fusion (SPFF), tailored for advanced pest detection in a precision agriculture system. The diagram in Figure 8 illustrates a

detailed step-by-step process that combines deep learning architecture with spatial pooling strategies to enhance multi-scale feature extraction and pest classification accuracy.

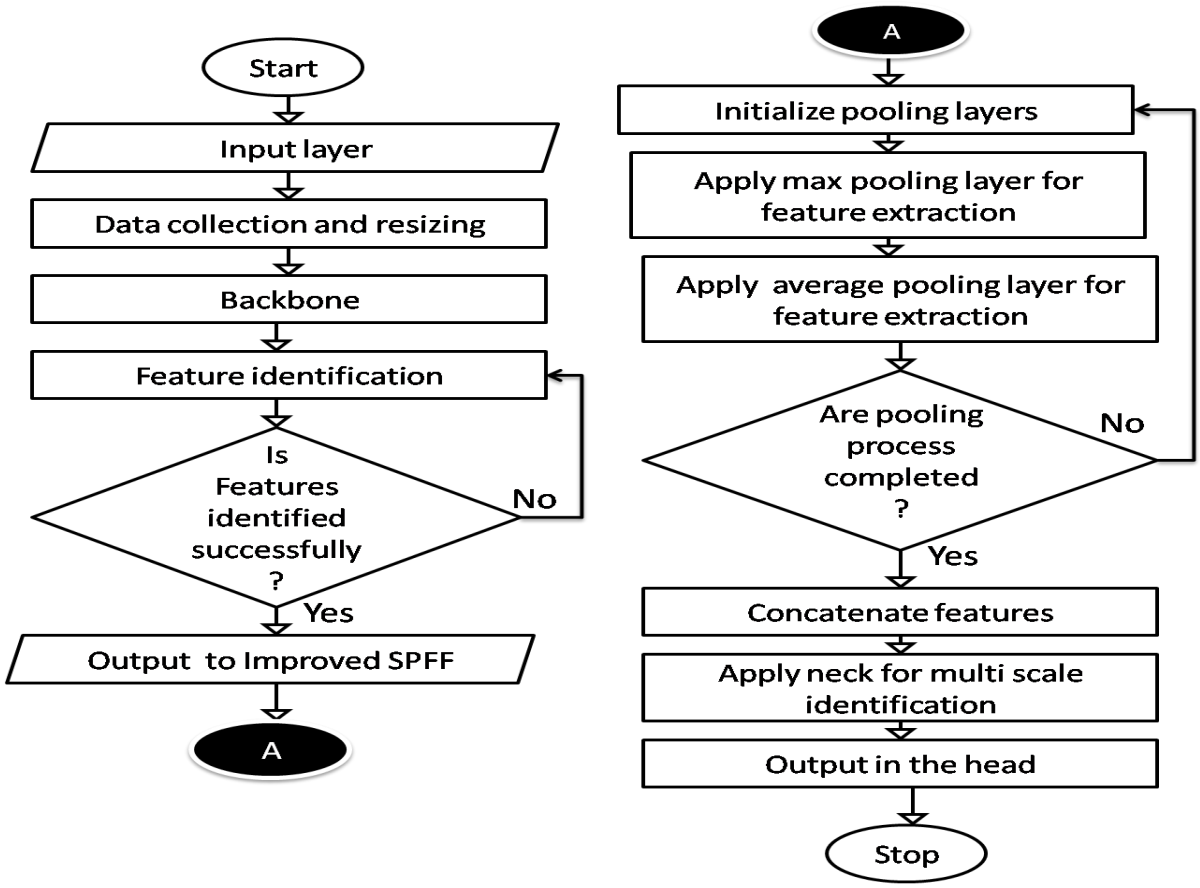

Figure 8: Flowchart of YOLOV-10 with improved SPFF

Figure 8 presents the flowchart of the YOLOv10 model integrated with an Improved SPFF module, specifically designed for advanced pest detection in a precision agriculture context. The flowchart outlines the sequential steps that combine deep learning-based object detection with enhanced spatial feature extraction strategies to improve classification accuracy across multiple scales. The process begins at the Input Layer, which receives image data captured through IoT-enabled camera sensors deployed across the agricultural field. This input is then subjected to the Data Collection and Resizing stage, where raw images are preprocessed and resized to match the input dimensions required by the YOLOv10 model. This resizing ensures consistency across the dataset and optimizes the model's detection capabilities.

Following preprocessing, the image data is fed into the Backbone of the YOLOv10 architecture. The backbone comprises a series of convolutional layers designed to extract both low-level and high-level features, including shapes, textures, and structural patterns indicative of pest presence. Subsequently, the Feature Identification block processes the extracted information to identify and highlight discriminative features relevant to pest classification.A decision node then evaluates whether features have been successfully identified. If feature identification fails, the process may either loop back for reevaluation or terminate, depending on system configuration. If successful, the output is forwarded to the Improved SPFF module, with a transition denoted by the label "A" in the flowchart.

Within Section A, the model initiates the Pooling Layer Initialization, preparing for a two-phase pooling mechanism. The first step applies Max Pooling, which emphasizes the most prominent features within the feature maps. This is followed by Average Pooling, which helps retain broader contextual information, ensuring the model remains sensitive to both dominant and subtle signals. After pooling, a conditional check determines whether the feature refinement is complete. If

not, additional pooling operations may be executed. Upon completion, the features are merged through the Concatenate Features step, producing a robust, multi-scale representation of the input data.

The concatenated features are then passed through the Neck module, which performs Multi-Scale Identification by integrating features across different layers of abstraction. This step enhances the model's capacity to detect pests of varying sizes, shapes, and orientations in diverse field conditions. Finally, the refined features are processed by the Head module, which generates the final outputs, including bounding boxes and class probabilities that indicate the presence and type of pests. Figure 9 presents the pest management system flow chart.

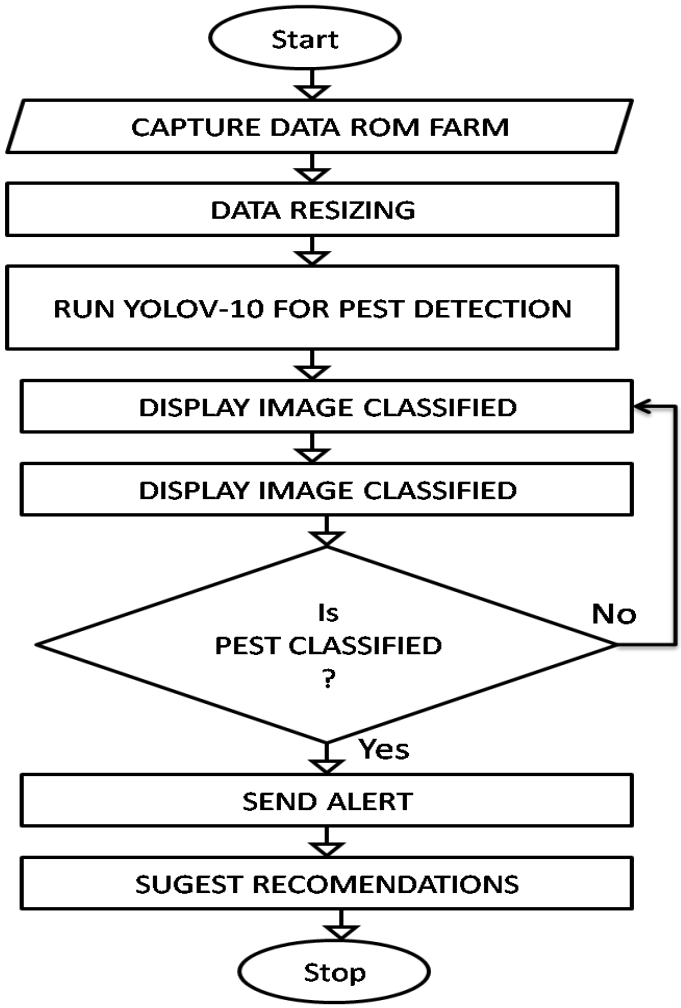

Figure 9: Flow chart of the pest management system

Figure 9 presents the flowchart of the pest detection and classification system utilizing the YOLOv10 model within an agricultural monitoring framework. The process begins with the acquisition of image data directly from the farm environment. These images are captured using IoT-enabled sensors or cameras strategically positioned for optimal field coverage. Following data acquisition, the images undergo a preprocessing stage where resizing is performed to ensure compatibility with the input dimension requirements of the YOLOv10 model. This resizing step is essential for maintaining uniformity in the dataset and improving the model's detection accuracy. The YOLOv10 algorithm is executed for pest detection. This model performs real-time object detection by identifying and classifying pests present in the resized images. YOLOv10 applies an improved architecture that includes enhanced modules such as the SPFF for better feature extraction and classification accuracy.

Once the detection phase is complete, the system proceeds to the visualization stage, where the classified images are displayed for further analysis or verification. Although the image display is shown twice in the diagram, it may be interpreted as a step for both system-side visualization and end-user interface output. The system then enters a decision phase to evaluate whether pest classification was successfully achieved. If classification fails, the process may halt or reinitiate depending on the implementation logic. If classification is successful, the system generates an alert notification. This alert is intended to inform farm managers or agricultural stakeholders about the pest occurrence in real time. The final stage involves generating recommendations based on the identified pest species. These recommendations may include control strategies, suggested pesticides, or biological treatments tailored to the specific pest type. Overall, the system integrates deep learning-based object detection with real-time monitoring technologies to enhance precision agriculture through timely and informed pest control measures.

3. SYSTEM IMPLEMENTATION

This section describes the practical implementation of the smart farm pest detection and notification system. It outlines the proposed system requirements, including the necessary hardware and software components needed to deploy and run the solution efficiently. Figure 10 presents the implementation interface results.The development environment for the proposed pest management system was carefully selected to ensure optimal compatibility with deep learning libraries, real-time data processing capabilities, and seamless integration with notification and web services. The primary programming language utilized was Python 3.8+, chosen for its robustness, simplicity, and vast ecosystem of libraries that support machine learning, image processing, and rapid development of AI solutions. Model development and debugging were carried out using Jupyter Notebook and Visual Studio Code, both of which offer rich development features, flexibility, and interactive visualization tools suitable for iterative experimentation.

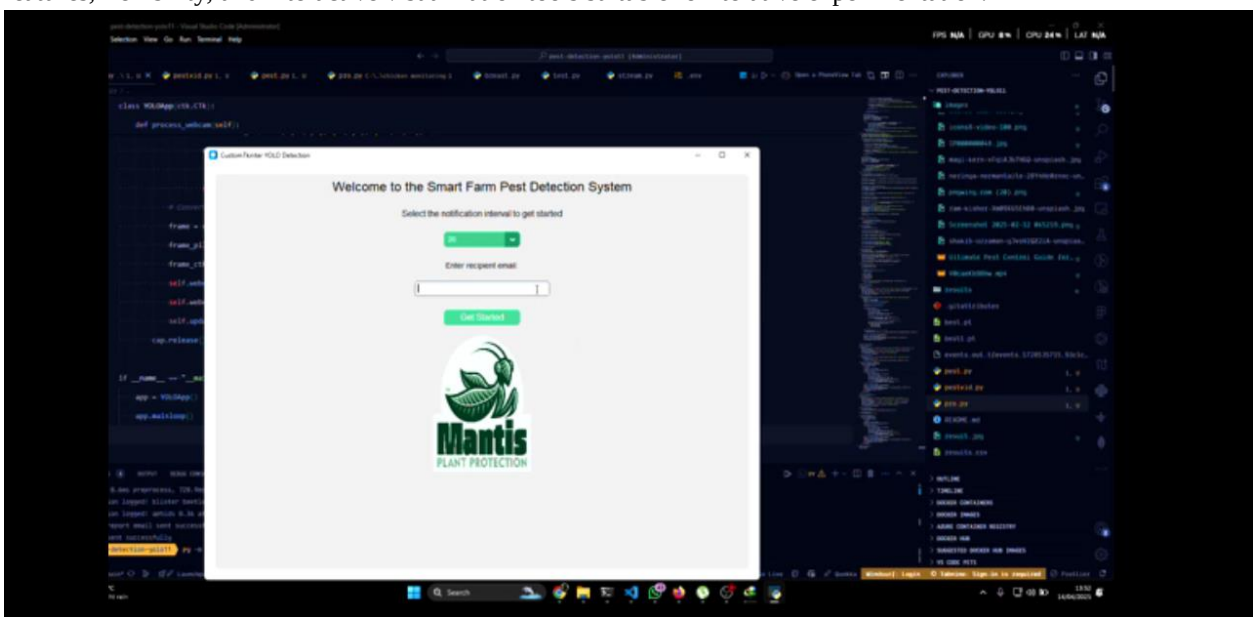

Figure 10: The system implementation results

For the core deep learning tasks, the PyTorch framework was employed to implement and train the YOLOv10 model. PyTorch was preferred due to its dynamic computation graph, native CUDA support, and active developer community, making it ideal for high-performance training and customization. The entire system was developed on Ubuntu 20.04 LTS, an open-source operating system renowned for its stability, security, and seamless integration with NVIDIA GPUs and the CUDA toolkit, which are critical for accelerating deep learning workloads.

3.1 System Results

The performance metrics employed for this evaluation include accuracy, training loss, precision, and mean Average Precision (mAP). Table 1 presents the training outcomes for the proposed YOLOv10 model integrated with the Improved

SPPF layer. The enhanced model demonstrates noticeable gains in precision and mAP, reflecting its superior ability to generalize and accurately detect pests of varying scales. The introduction of the improved SPPF module facilitated better multi-scale feature aggregation by combining both max and average pooling operations, thereby enhancing the model's capability to extract context-aware features from different receptive fields.

Table 1: Result of the YOLOV-10+New SPFF training

Table 1 presents the results of the YOLOV-10+NEW SPFF. The Table 2 reported the final training performance

Epoch	Train Box Loss	Val Box Loss	Train Obj Loss	Val Obj Loss	Train Cls Loss	Val Cls Loss	Precision	Recall	mAP@0.5	mAP@0.5:0.95
1	0.11324	0.10372	0.08592	0.07485	0.04908	0.05267	0.71008	0.64512	0.54169	0.33167
2	0.10931	0.10333	0.08429	0.07355	0.04978	0.05349	0.71504	0.65715	0.55083	0.34076
3	0.10829	0.10117	0.08336	0.07439	0.04974	0.04576	0.7267	0.66227	0.56374	0.3519
4	0.10753	0.09801	0.08138	0.07238	0.04861	0.0486	0.7355	0.66924	0.57397	0.3644
5	0.10158	0.09712	0.07465	0.06944	0.04643	0.04513	0.74122	0.67707	0.58503	0.37192
6	0.09921	0.09614	0.07373	0.06809	0.04743	0.04444	0.75153	0.68928	0.59597	0.38104
7	0.10054	0.09705	0.07488	0.06883	0.04294	0.04324	0.75338	0.69316	0.6074	0.38736
8	0.09669	0.09164	0.07318	0.06356	0.04371	0.04174	0.75939	0.70665	0.61606	0.40044
9	0.09206	0.08988	0.07153	0.06435	0.04319	0.04263	0.76506	0.70619	0.62929	0.40884
10	0.09198	0.08734	0.0766	0.06141	0.04297	0.03997	0.77815	0.72074	0.63447	0.42022
11	0.08793	0.08183	0.06849	0.05998	0.04237	0.0413	0.78282	0.72456	0.64508	0.42338
12	0.08595	0.08385	0.06811	0.06125	0.03775	0.03981	0.78756	0.73053	0.6535	0.4323
13	0.08545	0.08232	0.06628	0.05938	0.036	0.0386	0.79421	0.73602	0.66593	0.43999
14	0.07928	0.08546	0.06425	0.05808	0.04072	0.03647	0.79721	0.74411	0.67183	0.45005
15	0.07785	0.07854	0.06094	0.05782	0.03796	0.03635	0.80998	0.74926	0.68166	0.45389
16	0.07842	0.07796	0.06175	0.05404	0.03495	0.03826	0.81081	0.75738	0.68454	0.46161
17	0.07582	0.07577	0.05739	0.0542	0.03872	0.03702	0.81608	0.76107	0.69573	0.46903
18	0.07682	0.07203	0.0572	0.05108	0.03506	0.03335	0.82245	0.76665	0.70442	0.47476
19	0.07278	0.07522	0.05548	0.05124	0.03641	0.03482	0.82696	0.77096	0.71004	0.47986

The Table 2 reported the final training performance, while Table 3 reported the final training performance for the traditional YOLOV-10. The comparative results of the training process were recorded in Table 4, while Figure 11 and Figure 12 presented the graphical analysis of the comparative YOLV-10 performance.

Table 2: Result of the YOLOV-10 + New SPFF training

Val Box Loss	**Train Obj Loss**	**Val Obj Loss**	**Train Cls Loss**	**Val Cls Loss**	**Precision**	**Recall**	**mAP@0.5**	**mAP@0.5:0.95**
0.02785	0.02109	0.02006	0.00807	0.01475	0.97519	0.93163	0.92486	0.67272

Table 3: Result of the YOLOV-10 + SPFF training

Train Box Loss	**Val Box Loss**	**Train Obj Loss**	**Val Obj Loss**	**Train Cls Loss**	**Val Cls Loss**	**Precision**	**Recall**	**mAP@0.5**	**mAP@0.5:0.95**
0.03516	0.03685	0.02479	0.02866	0.01447	0.02075	0.88519	0.84463	0.82746	0.58272

Table 4: Comparative results of training

Metrics	YOLOV-10+ SPFF	YOLOV-10+ New SPFF
Train Box Loss	0.03516	0.02616
Val Box Loss	0.03685	0.02785
Train Obj Loss	0.02479	0.02109
Val Obj Loss	0.02866	0.02006
Train Cls Loss	0. 01447	0.00807
Val Cls Loss	0.02075	0.01475
Precision	0.88519	0.97519
Recall	0.84463	0.93163
mAP@0.5	0.82746	0.92486
mAP@0.5:0.95	0.58272	0.67272

Table 4 compared the results of the two models considering loss function, precision and recall. The Figure 11 compared the loss function during training and validation respectively, while the Figure 12 compared the accuracy, precision and recall of the two models respectively.

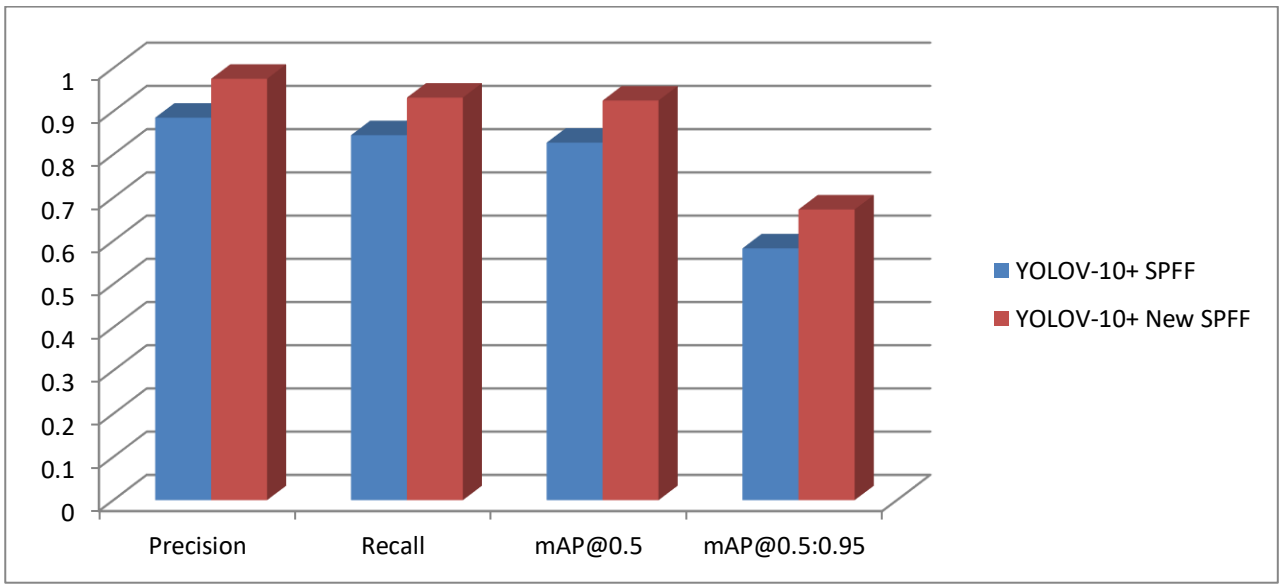

Figure 11: Comparative loss function

The Figure 11 presents the training bounding box loss which recorded 0.02616 for the new YOLOV-10 as against 0.03516 for the traditional YOLOV-10. This result measures the loss which occurred when the model tries to predict the bounding box on the image classified. The validation bounding box loss for the new model recorded 0.02785 as against 0.03685 for the traditional YOLOV-10. The training object loss measures the error which occurs when the model tries to predict the object from the video frame, and the results obtained for training is 0.02479 and validation of 0.02866 as against 0.02109 in the training object loss of the new model and validation loss of 0.02006 respectively. The classification loss measures the loss which occurred when the model tried to classify the image of pest in the farm.

The traditional YOLOV-10 recorded for the training 0.01447 and validation loss of 0.02075 while for the new YOLOV-10, the loss recorded is 0.00807 and validation loss of 0.01475. Figure 12 compared the recall, precision and mean absolute precision of the two models. The precision measures the chances that the model was able to successfully predict

pest on the farm correctly, the recall measures the probability that the model was able to successfully predict pest in the farm and it was actually a pest. The mean absolute precision measures the precision score of the model at different intersection over unit of 50 and 50 to 95% respectively.

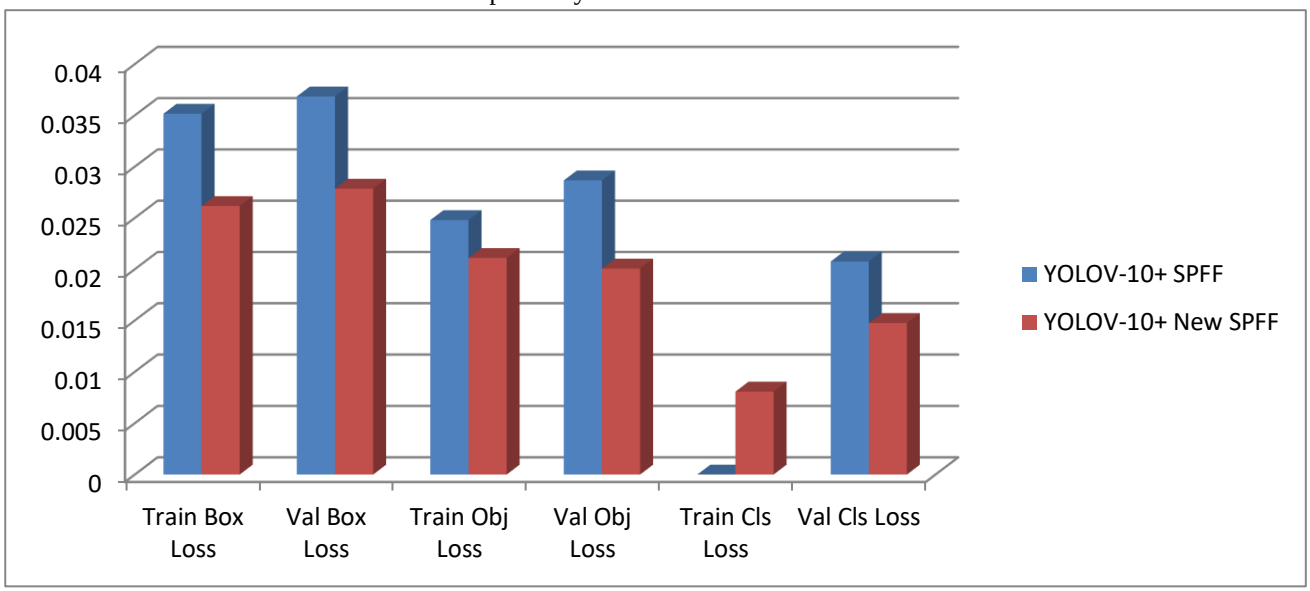

Figure 12: Comparative training results

From the results, it was observed that precision of the model with YOLOV-10 recorded 0.88519 while the recall recorded 0.84463. The mAP@0.5 recorded 0.82746 and 0.58272 for the traditional YOLOV-10, while for the new model the precision recorded 0.97519, recall reported 0.93163 and mAP@0.5 recorded 0.92486 and 0.67272 respectively. From these results, it was observed that the new YOLOV-10 in all metrics supersedes the traditional models and the reason was due to the optimization of the SPPF layer which increased the quality and quantity of features extracted, thereby optimizing the model performance in correctly pest detection in the farm.

4. CONCLUSION

In this study, an improved transfer learning model was developed using YOLOV-10 and optimized SPPF module achieved with global and maximum pooling layer. The model was trained as pest detection and classification system for real-time. This model was trained with data of pest and generate model for the classification of pest in real-time. In addition, a recommended model was developed which notify the farmer the appropriate measures to take upon detection of pest. This was conveyed to the farmer through email notification algorithm developed with simple mail transfer protocol. In evaluating the performance of both the traditional YOLOV-10 and the newly improved YOLOV-10 models were compared. These include the overall training and validation losses, the bounding box losses, and the pest detection losses. Each of these metrics provides insight into how well the models are learning to detect and localize pests within images or video frames.

The overall training and validation loss values indicate how well the model is learning during the training phase and how accurately it performs on unseen data. The traditional YOLOV-10 model recorded a training loss of 0.01447 and a validation loss of 0.02075. In comparison, the new YOLOV-10 model showed improved performance with a lower training loss of 0.00807 and a validation loss of 0.01475. These lower values signify that the new model is learning better during training and also generalizes more effectively to new data, reducing the chances of overfitting. The new YOLOV-10 model recorded a training bounding box loss of 0.02616, which is lower than the 0.03516 of the traditional models. Similarly, the validation bounding box loss for the new model was 0.02785, compared to 0.03685 for the traditional model. These improvements show that the new model is better at precisely locating the pests within the frame. Pest loss measures the model's ability to identify the presence or absence of pests in the frame essentially, how well it

distinguishes between regions with and without relevant pests. The traditional YOLOV-10 model had a training pest loss of 0.02479 and a validation pest loss of 0.02866. On the other hand, the new model performed better, with a training pest loss of 0.02109 and a validation pest loss of 0.02006. The decrease in pest loss confirms that the new model is more accurate in recognizing pests, even in varying and complex backgrounds.

The precision of a model tells us how often the pests that the model detected are actually correct. The traditional YOLOV-10 model recorded a precision of 0.88519, meaning about 88.5% of its detections were accurate. The new model significantly improved on this, with a precision of 0.97519, meaning it correctly identified nearly 98% of its detections a strong sign of improvement. The recall metric tells us how well the model finds all the relevant pests in the image. So if there are 100 pests in an image and the model only detects 84, the recall is 0.84. For the traditional model, recall was 0.84463, meaning it successfully detected around 84.5% of the actual pests. The new model, however, achieved a higher recall of 0.93163, meaning it was able to detect over 93% of the pests. This shows that the new model misses fewer pests and is more sensitive in its detection ability. The mAP@0.5 (mean Average Precision at Intersection over Union threshold of 0.5) is a summary measure that combines both precision and recall, while also considering how accurately the model predicts the position (bounding box) of each pest. It's a more comprehensive performance metric. The traditional YOLOV-10 recorded 0.82746 for the first mAP score and 0.58272 for the second, which are decent but leave room for improvement. The new model, on the other hand, achieved 0.92486 and 0.67272 for these two mAP values, showing that it not only detects pests better but also places bounding boxes more accurately around them.

In summary, every metric shows that the new YOLOV-10 model is more precise (fewer false detections), more sensitive (finds more actual pests), and more accurate in pest localization (better bounding boxes). These improvements mean the new model is more trustworthy and effective for real-world pest detection task environments where accuracy and reliability are critical. System integration was carried out using Python programming language. The performance of the model was evaluated considering different pests, and the results reported successful detection and classification of pest.

REFERENCES

Alanazi, A., Shakeabubakor, A., Abdel-Khalek, S., & Alkhalaf, S. (2023). IoT enhanced metaheuristics with deep transfer learning based robust crop pest recognition and classification. *Alexandria Engineering Journal, 84*, 100–111. https://doi.org/10.1016/j.aej.2023.11.008

Anwar, Z., & Masood, S. (2023). Exploring deep ensemble model for insect and pest detection from images. *International Conference on Machine Learning and Data Engineering. Procedia Computer Science, 218*, 2328–2337.

Banga, K. S., Kotwaliwale, N., Mohapatra, D., & Giri, S. K. (2018). Techniques for insect detection in stored food grains: An overview. *Food Control, 94*, 167–176.

Cammell, M. E., Tatchell, G. M., & Whittaker, J. B. (1992). Effects of climatic change on the population dynamics of crop pests. *Advances in Ecological Research*.

Chen, P., Xiao, Q., Zhang, J., Xie, C., & Wang, B. (2020). Occurrence prediction of cotton pests and diseases by bidirectional long short-term memory networks with climate and atmosphere circulation. *Computers and Electronics in Agriculture, 176*, 105612.

Cheng, X., Zhang, Y., Chen, Y., Wu, Y., & Yue, Y. (2017). Pest identification via deep residual learning in complex background. *Computers and Electronics in Agriculture, 141*, 351–356.

Emeric, C., Petit, S., & Poggi, S. (2022). Weather and landscape drivers of the regional level of pest occurrence in arable agriculture: A multi-pest analysis at the French national scale. *Agriculture, Ecosystems & Environment, 338*, 108105.

Gao, J., Nuyttens, D., Lootens, P., He, Y., & Pieters, J. G. (2018). Recognising weeds in a maize crop using a random forest machine-learning algorithm and near-infrared snapshot mosaic hyperspectral imagery. *Biosystems Engineering, 170*, 39–50.

Johnson, J. B. (2020). An overview of near-infrared spectroscopy (NIRS) for the detection of insect pests in stored grains. *Journal of Stored Products Research, 2020*, 101558.

Karar, M. E., Alsunaydi, F., Albusaymi, S., & Alotaibi, S. (2021). A new mobile application of agricultural pests recognition using deep learning in cloud computing system. *Alexandria Engineering Journal*, *60*(5), 4423–4432.

Karar, M., Abdel-Aty, A., Algarni, F., Hassan, M., Abdou, M., & Reyad, O. (2022). Smart IoT-based system for detecting RPW larvae in date palms using mixed depthwise convolutional networks. *Alexandria Engineering Journal*, *61*, 5309–5319. https://doi.org/10.1016/j.aej.2021.10.050

Kumar, D., & Kalita, P. (2017). Reducing postharvest losses during storage of grain crops to strengthen food security in developing countries. *Foods*, *6*(8).

Mendoza, Q., Lester, P., & Mitchell, N. (2023). Application of machine learning for insects monitoring in grain facility. *AI*, *4*(1). https://doi.org/10.3390/ai4010006

Prasath, B., & Akila, M. (2023). IoT-based pest detection and classification using deep features with enhanced deep learning strategies. *Engineering Applications of Artificial Intelligence*, *121*, 105985. https://doi.org/10.1016/j.engappai.2023.105985

Srivastava, S., & Mishra, H. N. (2021). Detection of insect damaged rice grains using visible and near-infrared hyperspectral imaging technique. *Chemometrics and Intelligent Laboratory Systems*, *221*, 104489.

Vemuri, H. (2023). Pest detection system. *International Journal of Science, Engineering and Technology*. ISSN (Online): 2348-4098, ISSN (Print): 2395-2404.

Zhang, Q., Wang, X., Shi, H., Wang, K., Tian, Y., Xu, Z., Zhang, Y., & Jia, G. (2025). BRA-YOLOv10: UAV small target detection based on YOLOv10. *Drones*, *9*(3), 159. https://doi.org/10.3390/drones9030159