Deep learning pest detection on Indonesian red chili pepper plant based on fine-tuned YOLOv5

Indra Agustian [a,1,*], Ruvita Faurina [b,2], Sahrial Ihsani Ishak [b,3], Ferzha Putra Utama [b,4], Kusmea Dinata [c,5], Novalio Daratha [a,6]

[a] Study Program of Electrical Engineering, University of Bengkulu, Indonesia
[b] Study Program of Informatics, University of Bengkulu, Indonesia
[c] Agricultural Technology Study Center, Agriculture Department, Province of Bengkulu, Indonesia
[1] indraagustian@unib.ac.id *; [2] ruvita.faurina@unib.ac.id; [3]sahrialishak@gmail.com; [4]fputama@unib.ac.id; [5]dinata.kusmea@gmail.com; [6]ndhrata@unib.ac.id
* corresponding author

ARTICLE INFO

Article history
Received July 5, 2022
Revised May 21, 2023
Accepted June 20, 2023
Available online October 15, 2023

Keywords
Deep learning
Yolo
Indonesia red chili paper
Pest detection

ABSTRACT

This research developed a pest detection model for Indonesian red chili pepper based on fine-tuned YOLOv5. Indonesian red chili pepper is the third largest vegetable commodity produced in Indonesia. Pest attacks disrupt the quantity and quality of crop yields. To control pests effectively, it is necessary to detect the type of pest correctly. A viable solution is to leverage computer vision and deep learning technologies. However, no previous studies have developed a pest detection model for Indonesian red chili pepper based on this technology. YOLOv5 is a variant of the YOLO object detection algorithm, which has major advantages in terms of computation cost and execution speed. The dataset comprises 4,994 image files collected from a chili plantation in Bengkulu province, Indonesia, covering 4 different classes and a total of 10,683 pests. The image is 1216 x1216 px with the smallest, largest, and average object dimensions of 2%, 35%, and 4% of the image dimensions. The training model used is fine-tuning YOLOv5s with variations of patience as an early stop parameter of 100, 200, and 300. The evaluation of the trained model is based on train loss, validation loss, and mAP@0.5:0.95, the best-trained model is the 445th epoch on patience 100 with the best confidence value of 0.321 and the highest TF1 of 0.74. From the best-trained model testing on the test dataset, the mAP@0.5 performance for all classes is 81.3%. The model not only detected large pests but was also able to detect objects that were small in size compared to the image size. The best-trained model's best mAP@0.5 performance and speed are 82.6% and 20 ms/image, or 50 fps on NVIDIA P100 GPU.

1. Introduction

Indonesian red chili pepper (*Capsicum annuum* L.) or better known as "Red chili pepper" is one of the most popular vegetable commodities that has a high economic value in Indonesia [1]. Indonesian Red chili pepper production in Indonesia is the third-largest production after shallots and cabbages [2]. The market price of this commodity often varies due to supply disruptions caused by pests and diseases, leading to reduced production quality and quantity. Indonesian Red chili pepper plants are known to be highly susceptible to a variety of pests, Subagyono *et al.* [3] stated that there are five main pests that attack the Indonesian red chili pepper plant, (1) Green peach aphid (*Myzus persicae* Sulz.), (2) Thrips (*Thrips parvispinus* Karny), (3) Broad mite (*Polyphagotarsonemus latus* Banks), (4) Oriental fruit fly

(*Bactrocera dorsalis* Hendel), and (5) Cotton bollworm (*Helicoverpa armigera* Hubner). These five pests can cause crop failure between 20-100% so they require proper control.

In line with the problem of plant pest, the Indonesian government has created an Integrated Pest Control Program, which is regulated in Law of the Republic of Indonesia No. 12 of 1992 regarding Plant Cultivation System that is expected to reduce farmers' losses due to pest attacks [4]. This program requires that control be carried out by applying the "six correct" pesticides, namely: correct target, correct quality, correct type, correct time, correct dose, and correct method. This requires a different handling process depending on the type of pest that attacks. Farmers often misuse pesticides, which leads to their ineffectiveness, wastage of pest control costs, and incomplete eradication of pests. Therefore, it is necessary to have a pest control program on large chilies accurately and quickly.

The initial stage of integrated pest control is the ability to identify pests correctly. Conventional pest identification is obtained using direct observation of the pest. Pests can be identified based on their shape and or effects. In the conventional way, observers need special knowledge or expertise regarding pests and plant types. Computer vision technology has modernized the process of recognizing and detecting pests, as digital images captured by cameras can now be processed to automatically identify pests computer-based. The use of computers and digital cameras has led to large and rapid changes in pest identification and detection systems, from basic techniques in image processing to advanced machine learning and deep learning.

Motivated by the development of deep learning in computer vision technology, especially object detection based on the YOLO algorithm [5], this study developed a pest detection system based on YOLOv5 [6], which is the latest version of YOLO. The pests detected were pests that attacked Indonesian red chili pepper plants, such as Green peach aphid (*Myzus persicae*), Silverleaf whitefly (*Bemisia tabaci*), Thrips, Cotton bollworm (*Helicoverpa armigera*), and Tobacco cutworm (*Spodoptera litura*). The dataset used is a primary dataset collected by the research team from the Indonesian Red chili pepper plantation in Bengkulu, Indonesia.

YOLO is one of the most popular models and is claimed to have an excellent performance, especially in terms of speed, compared to other deep learning models for object detection that have been developed to date. The YOLO model has developed very quickly, starting from YOLOv1 [5], YOLOv2 [7], YOLOv3 [8], YOLOv4 [9] to YOLOv5 [6]. YOLOv4 and YOLOv5 are considered to have the best performance compared to the previous YOLO version. In this study, the YOLOv5s release 6 model was chosen for pest detection, considering its performance in terms of mAP and speed. The objective was to develop a model that could be utilized in smartphone applications.

Previous studies have not developed a pest detection system that attacks Indonesian Red chili pepper plants, and there is no study that has studied the value of the patience parameter on YOLOv5s to get the best model. Patience serves to determine an early stop. If in the number of epochs (n+patience), the trained model does not experience improvement, then the best trained model is the n^{th} epoch [6]. Therefore, the major contributions of this work are as follows: 1). Detection model of pest that attacks Indonesian Red Chili pepper plant based on fine-tuning learning YOLOv5s. The types of pests detected were Green peach aphid (*Myzus persicae*), Silverleaf whitefly (*Bemisia tabaci*), Thrips, Cotton bollworm (*Helicoverpa armigera*), and Tobacco cutworm (*Spodoptera litura*). In addition to detecting larger pests, the model is capable of detecting small and tiny pest objects compared to the size of the image. 2). Comparison of models based on the value of patience as a determinant of early stop training. 3). Primary pest dataset collected from Indonesian red chili pepper plantation in Bengkulu, Indonesia.

The rest of this paper is organized as follows. Section 2 describes the previous studies related to this research. Section 3 describes the steps and details of the method. Section 4 explains the results of the research and its discussion. And finally, Section 5 contains the conclusions of the research that has been done.

2. Related work

The following are some previous studies that used basic image processing techniques for the detection or classification of pests on plants. The method of comparing the detected image with reference images [10], [11] or with heuristic indicators of healthy plants [12], and using some basic filters to improve quality and extract important features. The Relative Difference Intensity (RDI) algorithm employs a simple computation involving two subtractions and one comparison per pixel [13]. Image processing analysis by combining the Grab Cut algorithm and the One Cut algorithm to improve segmentation efficiency and accuracy [14]. Edge detection is utilized for the identification of pests, enabling the extraction of geometric shapes of insects [15]. Early detection of pests on greenhouse plants by combining image processing techniques with knowledge-based [16]. Pest detection and positioning based image segmentation, binocular stereo vision [17] and there is also pest segmentation using thermal imaging [18].

With the invention of machine learning, many previous researchers have made improvements in terms of feature extraction, classification, detection, or a combination of both, intending to increase the speed and accuracy of pest detection. Machine learning is employed in two ways for pest classification: the first is for segmentation or feature extraction, and the second is for classification purposes. Pest classification systems using machine learning are generally claimed to have better performance than using the previous basic image processing techniques.

Pratheba *et al.* [19] used machine learning for segmentation using the K-Means and Fuzzy C-means (FCM). Vinushree *et al.* [20] used a segmentation method based on a kernel-based fuzzy C-means clustering algorithm (KFCM) and a Neural Network (NN) classifier. Gondal & Khan [21] carried out morphology-based segmentation and classification using the Support Vector Machine (SVM). Bayat *et al.* [22] compared several feature extraction methods such as histograms, Laplacian filters, Canny, Sobel, and Principal Component Analysis (PCA) combined with NN and SVM classification methods. Ebrahimi *et al.* [23] developed a Thrips pest detection system on strawberry plants using the SVM method with a difference kernel functions. Paranjothi [24] used color histogram and contour-based feature extraction and SVM classifier. Dey *et al.* [25] used statistical-based feature extraction such as Gray Level Run Length Matrix (GLRLM) and Gray Level Co-occurrence Matrix (GLCM) and compared the performance among SVM, Artificial Neural Network (ANN), Bayesian, Binary decision tree, and K -Nearest Neighbor (KNN) classifier. Fina *et al.* [26] conducted a pest classification based on K-Means Clustering Algorithm & Correspondence Filters.

The discovery of deep learning technology has made a lot of improvisations to get a more accurate detection system. Deep learning has revolutionized feature extraction methods, replacing traditional image processing and machine learning techniques. With its convolutional layers, deep learning can independently extract features, rendering older methods obsolete. The following are previous studies that have been carried out to classify pests using deep learning methods. Türkoğlu & Hanbay [27] carried out pest classification with transfer learning of AlexNet, VGG16, VGG19, SqueezeNet, GoogleNet, Inceptionv3, InceptionResNetv2, ResNet50, and ResNet101 architectures, and compared them with SVM, extreme learning machine (ELM), and K-nearest neighbor (KNN). Malathi & Gopinath [28] carried out the classification of rice pests using transfer learning CNN with the architecture of AlexNet, GoogleNet, Resnet34, Resnet50, and Fine-Tuned Resnet-50.

Studies on pest detection systems based on deep learning conducted by previous researchers can be divided into three main parts, R-CNN based, YOLO based, and other methods. The following are previous studies of R-CNN-based pest detection systems. Fuentes *et al.* [29] carried out pest detection on tomato plants using the Faster Region-based Convolutional Neural Network (Faster R-CNN), Region-based Fully Convolutional Network (R-FCN), and Single Shot Multibox Detector (SSD) algorithm with a combination of VGG and ResNet architecture. Jiao *et al.* [30] modified the Region Convolutional Neural Network (RCNN) algorithm into an anchor-free Region Convolutional Neural Network (AF-RCNN) to detect multi-class pests. D. Li *et al.* [31] developed the detection of pests on rice plants using faster RCNN. W. Li *et al.* [32] detected tiny pests from sticky trap images with TPest-RCNN based on the Faster R-CNN. Patel & Bhatt [33] carried out pest detection with Faster R-CNN.

Zhang *et al.* [34] developed a multi-feature fusion Faster R-CNN (MF3 RCNN) to detect pests and diseases on soybean leaves. Jiao *et al.* [35] detected pests using CNN as feature extraction and used two-stage R-CNN for refining predicted bounding boxes. Lin *et al.* [36] detected pests on Sweet Peppers using faster R-CNN and Mask RCNN with Inception V2, ResNet-50, and ResNet-101 backbones.

Previous studies on pest detection systems using YOLO were as follows. J. Liu & Wang [37] carried out early detection of diseases on tomatoes based on MobileNetv2-YOLOv3. Bhatt *et al.* [38] used YOLOv3 to detect tea plant pests with DarkNet-19, MobileNet, Inception v2, ResNet-101, and DarkNet-53 architecture. Chen *et al.* [39] developed a smartphone application to detect the species of mealybugs, Coccidae, and Diaspididae using YOLOv4. Kuzuhara *et al.* [40] developed a regional proposal network for insect pest detection using YOLOv3 and the re-identification method using the Xception model. Liang *et al.* [41] detected butterfly using YOLOv3. Legaspi *et al.* [42] detected Whiteflies and Fruit Flies using YOLOv3. Mamdouh & Khattab [43] developed Olive Fruit Fly Detection and counting system using YOLOv4. Rustia *et al.* [44] developed greenhouse insect detection system using YOLOv3 and cascade CNN classifier. Tang *et al.* [45] developed a real-time pest detection based on YOLOv4. Verma *et al.* [46] developed pest detection on Soybean crop based on YOLOv3, YOLOv4, and YOLOv5. Zha *et al.* [47] developed pest detection based on YOLOv4 with MobileNetv2 as the feature extraction block. D. Li *et al.* [48] developed YOLO-based Jute Pest detection with a modified backbone that integrates the Sand Clock Feature Extraction Module (SCFEM), Deep Sand Clock Feature Extraction Module (DSCFEM), and Spatial Pyramid Pooling Module (SPPM) to extract image features effectively. J. Liu & Wang [49] detected pests on tomato plants using YOLOv3 with feature fusion to increase the number of feature maps. Önler [50] developed a real-time detection system for the thistle caterpillar (Vanessa cardui) which is a pest on sunflower plants using YOLOv5. Ahmad *et al.* [51] developed an insect pest detector based on YOLOv5 which can operate in real-time with image input from smartphone IP-camera.

The following are previous studies on pest detection systems other than R-CNN and YOLO based. Selvaraj *et al.* [52] detected diseases and pests on bananas using transfer learning method of the Convolutional Neural Network (CNN) SSD (Single Shot Detector) and compared the architectures of ResNet50, InceptionV2, and MobileNetV1. Burhan *et al.* [53] compared the architectures of CNN VGG16, VGG19, ResNet50, ResNet50V2, and ResNet101V2 for the classification of pests and diseases in rice. Nguyen *et al.* [54] conducted a comparison of transfer learning from eight variants of the EfficientNet architecture for the classification of Ladybird, Mosquito, Grasshopper, Butterfly, and Dragonfly pests. L. Liu *et al.* [55] designed a pest detection model called "PestNet" based on CNN with the addition of Channel-Spatial Attention (CSA) as a feature extraction backbone and adopted a Region Proposal Network (RPN) for the selection of area proposals and using a Position-Sensitive Score Map (PSSM), replaces the fully connected layer for the classification and bounding box regression. Roosjen *et al.* [56] detected The fruit fly Drosophila suzukii, or spotted wing drosophila (SWD) using a CNN developed by Kellenberger *et al.* [57] to detect mammals in UAV images. Lyu *et al.* [58] detected small grain pests using the SSD feature fusion method. L. Liu *et al.* [59] detected wild pests using Local activated Region Proposal Network (LaRPN) with Global activated Feature Pyramid Network (GaFPN) extraction. Wang *et al.* [60] used a Sampling-balanced region proposal (S-RPN) and fast R-CNN to detect small pests. Türkoğlu *et al.* [61] developed PlantDiseasesNet using the CNN feature extraction and SVM classifier for the classification of plant pests and diseases.

The previous studies mentioned above generally research and develop detection systems based on their respective datasets and the types of pests on certain plants. However, there were only a few studies that discuss detection performance for tiny pests. Several studies conducted studies by comparing the performance of several deep learning models, such as comparing the performance of the YOLO variant with the R-CNN variant. The significant research contributions mentioned in Chapter 1 are based on several studies that have compared the performance of different architectures, such as R-CNN with ResNet-50 versus ResNet-101, within a single deep learning model.

3. Method

3.1. Dataset

3.1.1. Images Acquisition

The pest images used for the dataset in this study were taken directly at the Indonesian red chili pepper garden center in Bengkulu Province, Indonesia, using a smartphone camera with a minimum of 13 MP (Mega Pixel) camera so as to produce a fairly good resolution. Pest images are taken at a distance of 10–20 cm from the camera. Data collection time was carried out in the morning and evening. For the morning session, it started at 7.00–10.00 Western Indonesian Time. Meanwhile, for the afternoon session started at 14.30–16.00 Western Indonesian Time. The schedule is adjusted to the time the pests begin to appear. The picture is not taken at a certain specific angle.

The data collection process was conducted five times. The first data collection was carried out on December 1, 2021. The second data collection was carried out on December 4, 2021. The third data collection was carried out on December 5, 2021. The fourth data collection was carried out on December 28, 2021, and lastly, the fifth data collection was carried out on December 30, 2021.

Four types of pests were successfully validated from the collected images, Green peach aphid (*Myzus persicae* Sulz.), Silverleaf whitefly (*Bemisia tabaci*), Thrips, Cotton bollworm (*Helicoverpa armigera*), and Tobacco cutworm (*Spodoptera litura*). The validation process is carried out by the relevant experts. Due to their similar visual appearance, the Cotton bollworm and Tobacco cutworm are combined into a single pest object class called "Caterpillar". The sample image dataset is presented in Fig. 1, and a large pest reference image is presented in Fig. 2.

(*Myzus persicae*) (*Bemisia tabaci*) (Thrips) (Caterpillar)

Fig. 1. Image Samples

https://www.inaturalist.org/photos/75370975 (a)

https://www.ars.usda.gov/oc/images/photos/nov11/d288-4/ (b)

https://commons.wikimedia.org/wiki/File:Thrips_(15817836204).jpg (c)

https://commons.wikimedia.org/wiki/File:Spodoptera_litura1.jpg (d)

https://commons.wikimedia.org/wiki/File:Helicoverpa_armigera_(24161899965).jpg (e)

Fig. 2. Pest image references (a) *Myzus persicae*; (b) *Bemisia tabaci*; (c) Thrips; (d) *Spodoptera litura*; (e) *Helicoverpa armigera*.

3.1.2. Dataset preparation and annotation

At the dataset preparation stage, the process of resizing, annotating, augmenting, and dividing data is carried out. To reduce computational costs, the image size was changed from 3240 x 3240 to 1216 x

1216 px while taking into account that the pest object in the annotated image is at least 24 x 24 px. The location distribution and object dimensions in the image are shown in Fig. 3. The smallest, largest, and average object dimensions are about 2%, 35%, and 4% of the image size, respectively, with an image size of 1216. Therefore, each dimension is 24, 462, and 49 px.

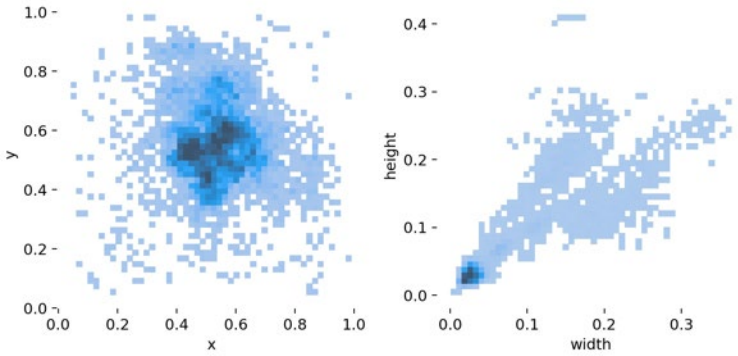


Fig. 3. The distribution of pest size data

In the annotation process, the area of each pest contained in the image is bound in a box and labeled according to the type of pest. Annotation examples are shown in Fig. 4.

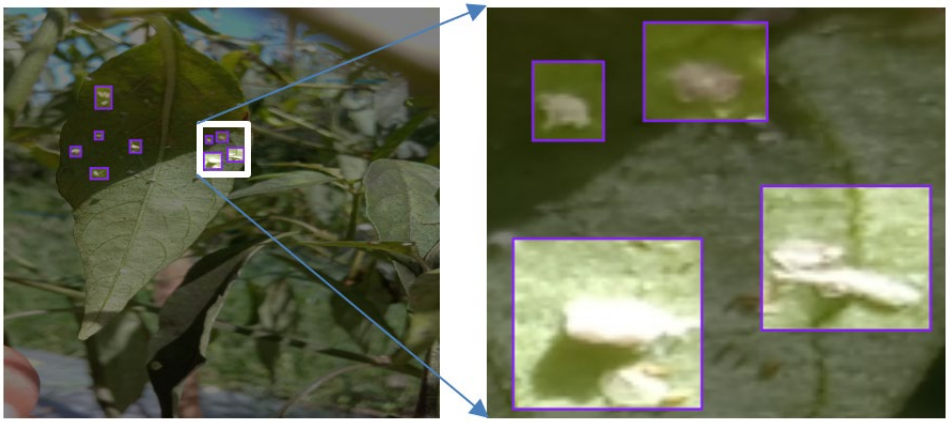


Fig. 4. *Bemisia tabaci* annotation sample

To enhance the variety of input image data, each input image is augmented into three output images. The augmentation technique used is presented in Table 1.

Table 1. The augmentation technique

No	Augmentation	Value
1	Rotation	-10° - 10°
2	Shear	10° horizontal, 10°vertical
3	Brightness	-15% - 15%
4	Noise	3%
5	Blur	3 px

The last step of data preparation is to divide the data for training, validation, and testing purposes, which are stored in the train, val, and test folders. Each of these folders consists of two sub-folders: an "image" sub-folder and a "labels" sub-folder. The "image" sub-folder contains image data in jpg or png format, while the "labels" sub-folder contains annotation data in txt format. The information in the txt file in the labels sub-folder is the data class, x-axis, y-axis, width, and height, each separated by a space. Meanwhile, the class name data for the entire dataset is stored in the "data.yaml" file. The dataset root folder structure is presented in Table 2. The total and number of each class are presented in Table 3 and the composition of the training, validation, and test datasets is shown in Table 4.

Table 2. The dataset folder structure

Dataset Folder						
train folder		*val folder*		*test folder*		*data.yaml*
images	labels	images	labels	images	labels	

Table 3. The composition of dataset based on pest class

No	Class	Class Label	Total images	Total object
1	Green Peach Aphid (*Myzus persicae*)	MP	568	3.589
2	Silverleaf whitefly (*Bemisia tabaci*)	BT	1.473	1.739
3	Thrips	T	795	2.580
4	Caterpillar	C	2.158	2.505
	Total		4.994	10.683

Table 4. The composition of training, validation, and test datasets

No	Dataset	Total
1	Train	70% = 3.496
2	Val	20% = 999
3	Test	10% = 499
	Total	100% = 4.994

3.2. Model and Training

Yolo is a single-stage object detector consisting of a backbone, a neck, and a head model [5]. The backbone is used for feature extraction from the input image. The neck component is utilized to generate pyramidal features, enabling the system to detect objects at various scales, while the head component is responsible for the final detection process.

YOLOv5 consists of several release versions, and when this research was conducted, the latest release used was v6.0. YOLOv5 has high detection accuracy, a lightweight model, and a high detection speed. Therefore, YOLOv5 can be used for real-time detection systems on embedded devices [6]. The main block of architecture of YOLOv5 - v6.0 is shown in Fig. 5. The Backbone YOLOv5 – v6.0 uses New CSP (Cross Stage Partial Networks)-Darknet53. The Neck model uses SPPF (Spatial Pyramid Pooling – Fast) and the New CSP-PAN (Path Aggregation Network). While the head model uses YOLOv3 Head, the same as the previous version. The hidden and final detection layer use the Sigmoid-Weighted Linear Units (SiLU) activation function. The optimization function consists of two options, SGD (Stochastic Gradient Descent) or Adam [6].

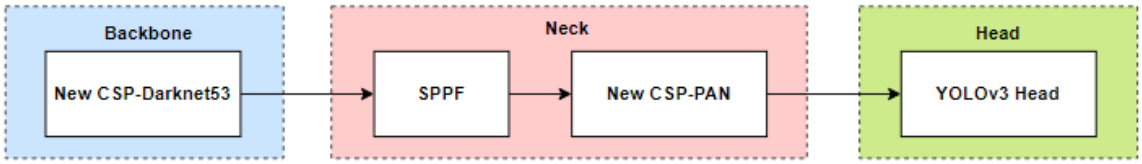


Fig. 5. Main block of Architecture YOLOv5

The training process is carried out at the Google Collaboratory with the Nvidia Tesla-P100 16GB GPU hardware accelerator. YoloV5 used is version 6 with the pyTorch framework. Fine-tuning with pre-trained model yolov5s.pt, batch 16, hyp.scratch.yaml, image size 1216, 1200 epoch, and SGD optimization function. The architecture of YOLOv5s with an input image size of 1216 is shown in Fig. 6.

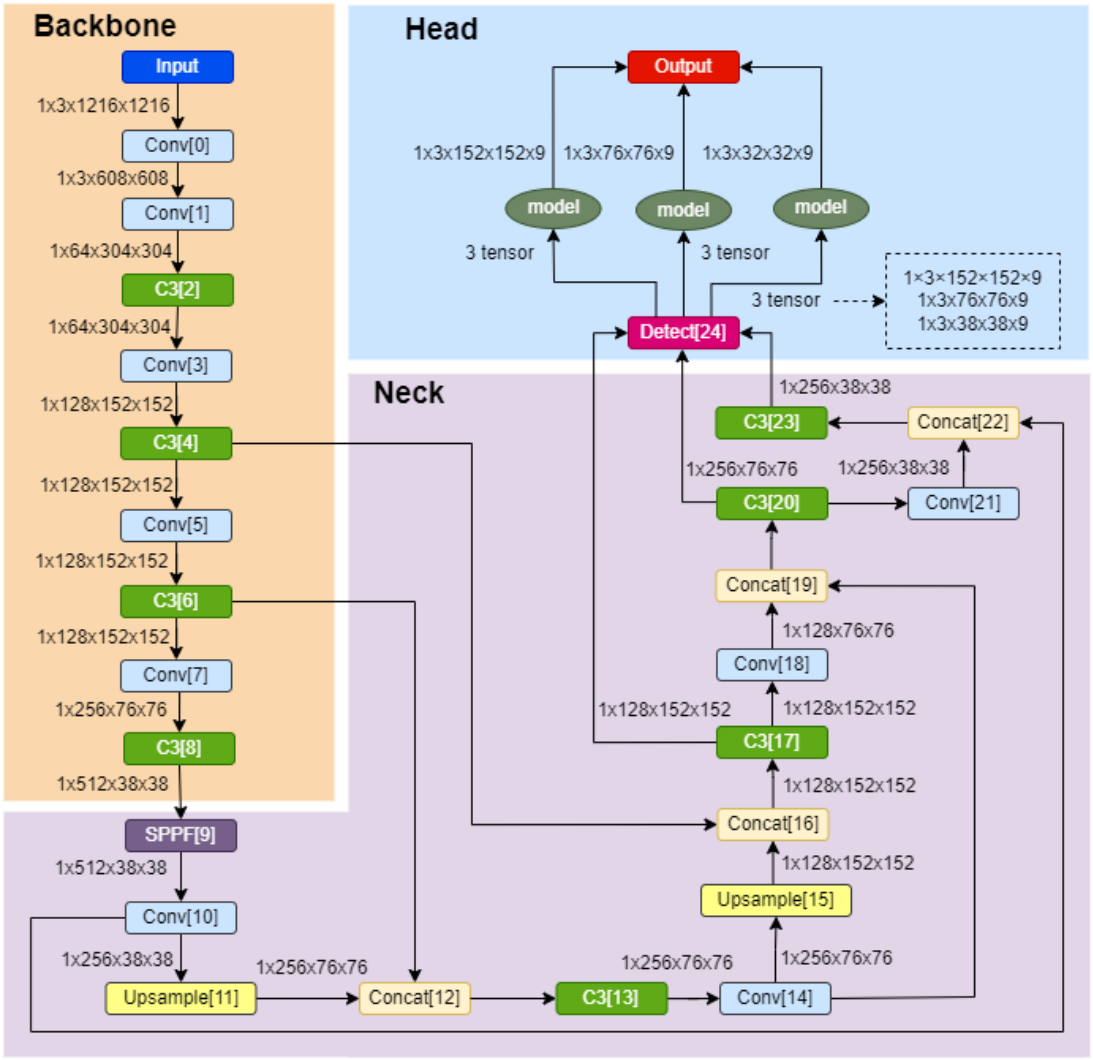


Fig. 6. Model YOLOv5s Architecture with input size 1216 x 1216 px

The SPPF, C3, Bottleneck, and Conv structure layers are shown in Fig. 7. The SPPF layer consists of two Conv layers and three MaxPool2D layers. The bottleneck layer in layer C3 consists of two types, bottleneck 1 and bottleneck 2. The type and number of bottlenecks used in each layer of C3 are presented in Table 5. Conv consists of three sequential processes: Conv2d, Bathnorm2D, and SiLu.

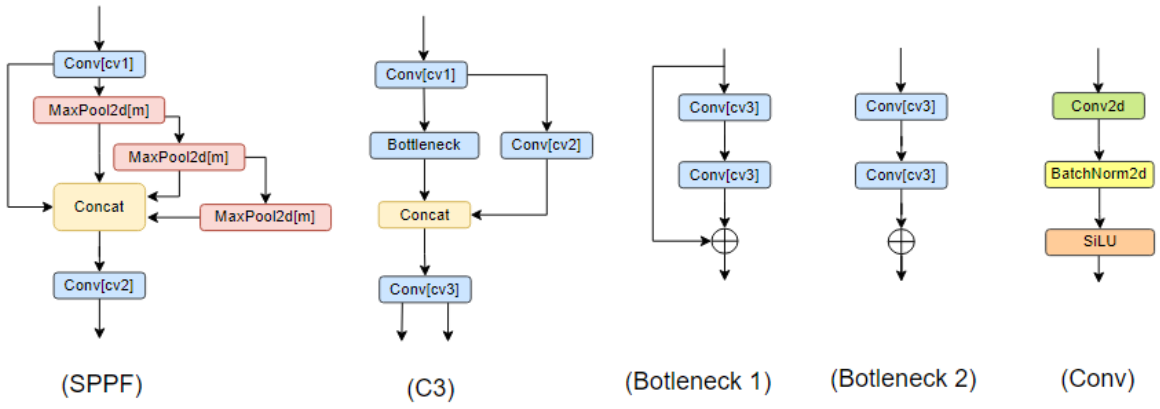


Fig. 7. SPPF, C3, Bottleneck, Conv structure

Table 5. Layer C3-Bottleneck

No	C3 Block	Botleneck 1	Botleneck 2
1	[2]	-	1
2	[4]	-	2
3	[6]	-	3
4	[8]	-	1
5	[13]	1	-
6	[17]	1	-
7	[20]	1	-
8	[23]	1	-

3.3. Evaluation

3.3.1. Evaluation Metrics

To evaluate the performance of the trained model, the observed indicators are train and validation loss data, precision, recall, F1, mAP@0.5, mAP@0.5:0.95, and mAP versus GPU Speed. Train and validation loss data consists of box loss, object loss, and class loss. A box loss occurs when there is an error in box prediction, while an object loss arises from an inaccurate prediction of the Intersection over Union (IoU) between the prediction. IoU is the ratio of the overlap area to the combined area of the predicted bounding box and the ground-truth box. A class loss is a loss due to deviation from predicting '1' for the correct class and '0' for all other classes for the object in the box. Precision, recall, and F1 are calculated based on confusion matrix data using equations (1), (2), and (3).

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{1}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{2}$$

$$\text{F1} = 2 \times \frac{\text{recall} \times \text{precision}}{\text{recall+precision}} \tag{3}$$

Precision shows the ratio of true positive prediction to all positive predictions data. Recall is shows the ratio of true positive predictions to all real positive data. The Confusion matrix shows performance of object prediction to the correct label of object, encompassing True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). The Confusion matrix form is presented in Fig. 8. TP indicates the number of correct positive predictions, TN is the number of correct negative predictions, FP is the number of false positive predictions, and FN is the number of false negative predictions.

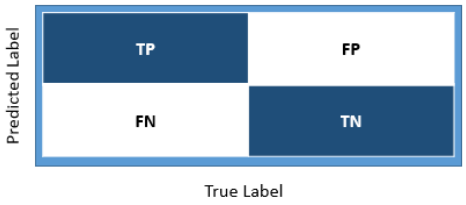


Fig. 8. Confusion Matrix

Average precision serves as an indicator of the model's capability to correctly recognize true positive labels while minimizing errors of predicting true negative labels as positive. As a result, a high average precision is achieved when the model accurately predicts positive labels. The average precision value is the area under the precision-recall curve based on different IoU threshold values. If precision and recall are each worth 1, the average precision value is 1. The average value of the average precision of all detection results is called the mean Average Precision (mAP).

3.3.2. Best-Trained Model

The best training model among patience 100, 200, and 300 is determined by comparing the performance of the training model based on evaluation indicators. The best-trained model of each

YOLOv5 training epoch is determined based on equation (4). Best-trained models were chosen and tested on the test dataset.

$$\text{best} = \max\{(0.1 \times \text{mAP@0.5}) \times (0.9 \times \text{mAP@0.5:0.95})\} \quad (4)$$

In the validation section, utilizing the default confidence threshold value of 0.001 in YOLOv5 can lead to high recall but lower precision. To overcome this, the best confidence value to use is chosen based on TF1 curve, which is the confidence value at the highest TF1 score.

In the evaluation section of the model, the mAP with IoU 0.5 (mAP@0.5) and the average mAP with IoU 0.5:0.95 (mAP@0.5:0.95) were used. As for the implementation of detection, the primary challenge metric uses mAP@0.5:0.95, the PASCAL VOC metric uses mAP@0.5, and the strict metric uses mAP@0.75. In this study, the IoU threshold used is 0.6, which is the default Non-Maximum Suppression IoU (NMS-IoU) of YOLO5v.

4. Results and Discussion

4.1. The Best Trained Model Evaluation

Based on the comparison of the patience value as an early stop determining parameter, the trained model with patience 100 is the best with the best-trained model obtained in the 445th epoch of the total target of 2000 epochs. Fig. 9, Fig. 10, and Fig. 11 show the training performance, validation, and mAP metrics of the trained model. The best training model obtained for each patience value is shown in Table 6. Patience values of 200 and 300 demonstrate improved training loss; however, they exhibit lower precision and recall metrics compared to a patience value of 100, particularly a significant decrease in recall at a patience of 300. This indicates that the ability of the trained model to predict TP is decreasing.

Table 6. Best Epoch based on patience

No	Patience	Early Stop Epoch	Best epoch
1	100	545 of 2000	445
2	200	1198 of 2000	998
3	300	1578 of 2000	1278

Based on loss validation, the performance of the trained model is getting lower with a higher patience value, even though the performance of the train loss is getting smaller. This suggests that increasing the training epoch with higher patience values can lead to overfitting of the trained model, as evidenced by the decreasing mAP metric with higher patience values.

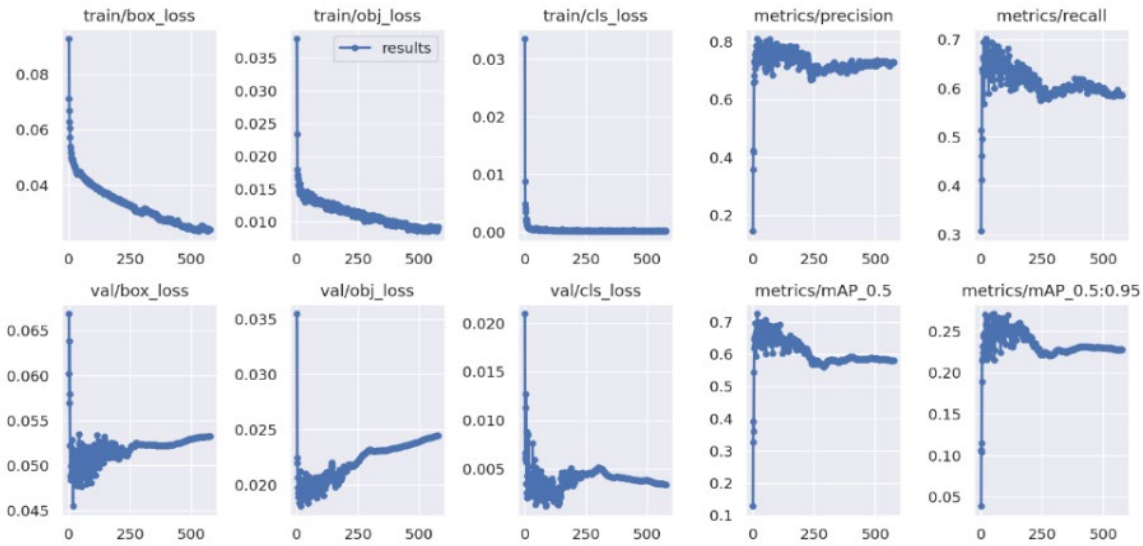


Fig. 9. Patience 100

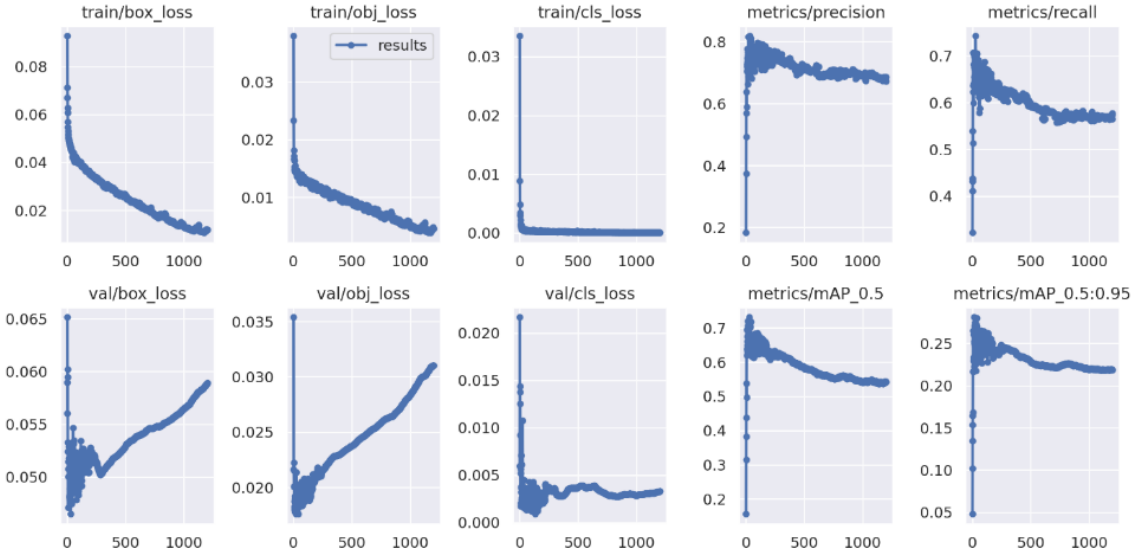


Fig. 10. Patience 200

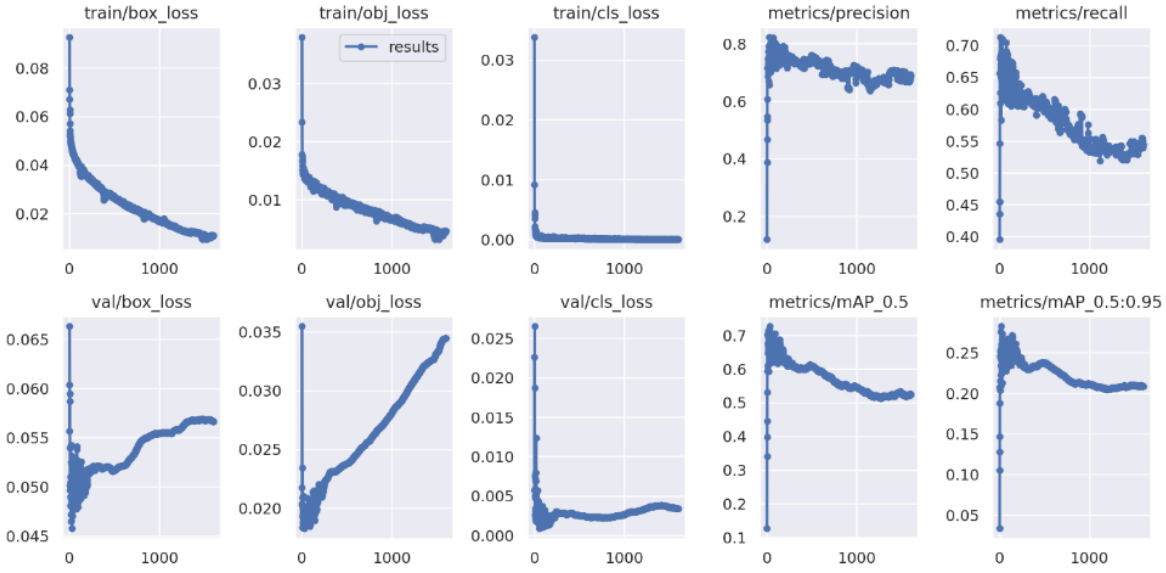


Fig. 11. Patience 300

The performance of GPU Speed versus mAP is shown in Fig. 12. As a performance comparison, we used mAP@0.5:0.95 from EfficientDet detection on the MS-COCO dataset with NVIDIA V100 GPU. The best-trained model performance for every patience with mAP@0.5:0.95 is lower than EfficientDet, but this is quite reasonable because this study used the NVIDIA P100 GPU, which has a DL training speed of 10 TFLOPS, while the NVIDIA V100 GPU has 120 TFLOPS. In addition, it is also caused by the difference in the ratio of the object size to the image size. The MS-COCO dataset contains fairly large common objects, such as people, cats, dogs, and others, while the pest dataset in this study consists of small objects. However, for mAP@0.5, the best trained model for each patience was much better than EfficientDet, and the best trained model for patience 100 showed the best performance. The best-trained model's best mAP@0.5 performance and speed are 82.6% is 20 ms/image or 50 fps on NVIDIA P100 GPU. The term "best trained model" pertains to the model trained with a patience of 100, which was chosen for subsequent evaluation on the test dataset. Details of the indicator data for the best epoch training results are presented in Table 7.

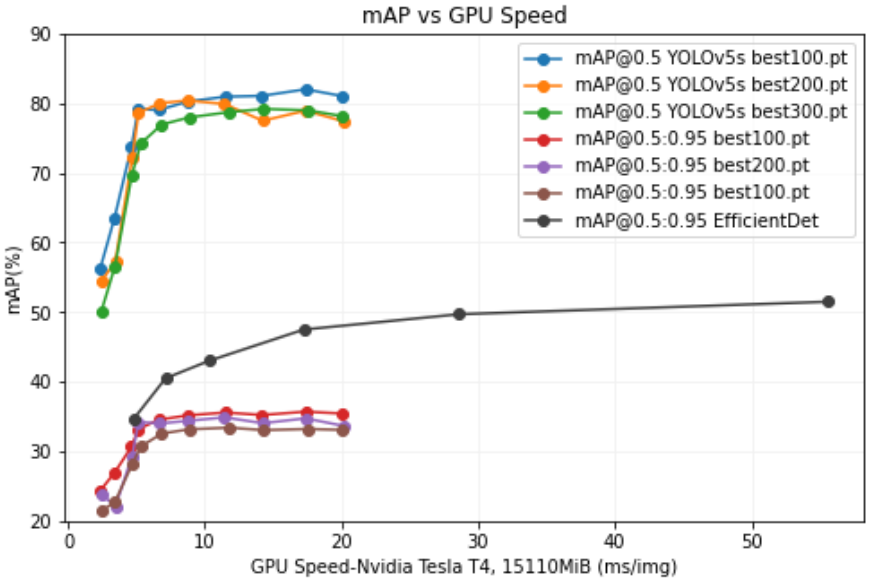


Fig. 12. GPU Speed vs mAP, best trained model with patience 100:best100.pt, patience200:best200.pt, patience 300:best300.pt (this research used NVIDIA P100, EfficientDet used NVIDIA V100)

Table 7. 445th epoch training data and validation with patience 100

Data	Box loss	Object loss	Class loss
Train	0.02619	0.0094296	0.00022905
Validasi	0.052495	0.023551	0.003869
Metrics	***precision***	***recall***	***F1***
	0.71407	0.60979	0.65782295
	mAP		
	@0.5	@0.5:0.95	
	0.64382	0.23058	

The performance of training with patience 100 is presented in Fig. 13. Precision has the highest value of 1 with a confidence value of 0.841. The highest recall is 0.88 with a confidence of 0.01, which indicates that precision and recall cannot be used as sufficient performance evaluation metrics for the model. The confidence value that will be used for the detection system cannot be taken from these two indicators. A high confidence value results in a low success rate of true positive (TP) predictions, but also reduces the number of false positive (FP) predictions. Conversely, a low confidence value leads to a higher TP prediction rate, but also increases the number of FP predictions. To get the best detection ability, the confidence value used is the confidence value with the highest F1-Score. Based on the F1-Score curve in Fig. 13, for all classes, the highest F1-Score is 0.74 with a confidence value of 0.321.

From the final results of the training process, validation with a default confidence value of YOLOv5s 0.001, the mAP metrics are shown in the Precision-Recall graph in Fig. 13. With an IoU of 0.5, the mAP for all classes is 0.726. However, as presented in Fig. 14 (confidence 0.001), this confidence value causes a very high false predictions. With a confidence value of 0.321, the results of object detection in the validation dataset are presented in Fig. 14 (confidence 0.321). Out of a total of 999 files, 837 files were successfully detected with objects in them, indicating a significantly enhanced object detection capability compared to the default confidence value.Based on the data on the detection test results indicators in Table 8, it can be seen that with 0.321 confidence, the trained model can detect 90.04% of objects on the validation datasets with more balanced precision and recall values and better mAP metrics. The lowest detection rate from the total validation dataset label occurred in BT class with a percentage of 63.55%. The performance of the best trained model detection results on the overall validation dataset image is presented in the confusion matrix in Fig. 15 (Validation dataset).

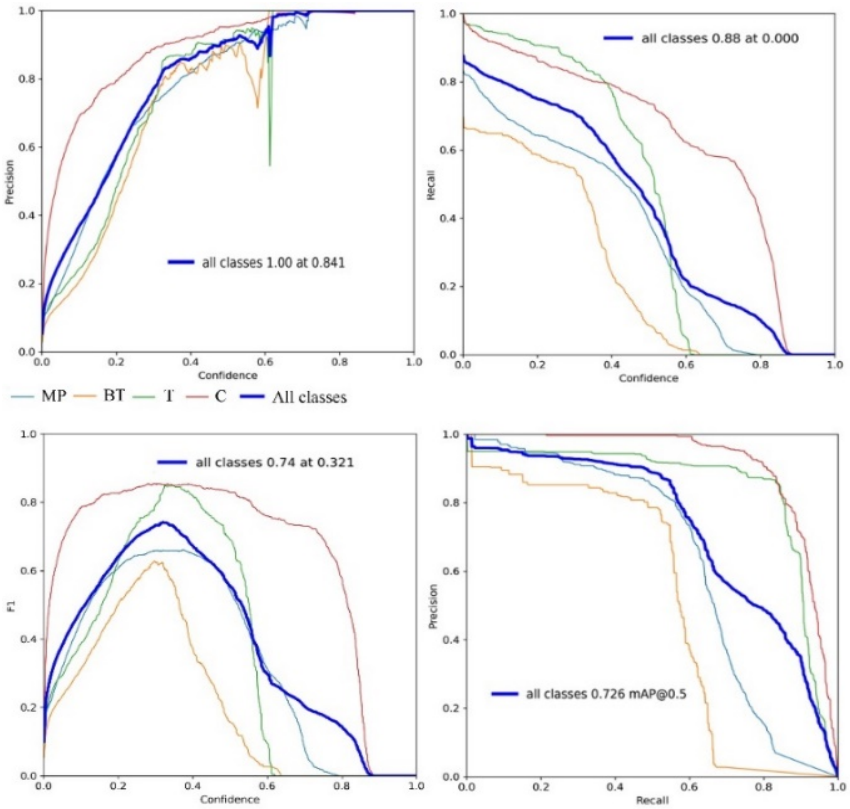
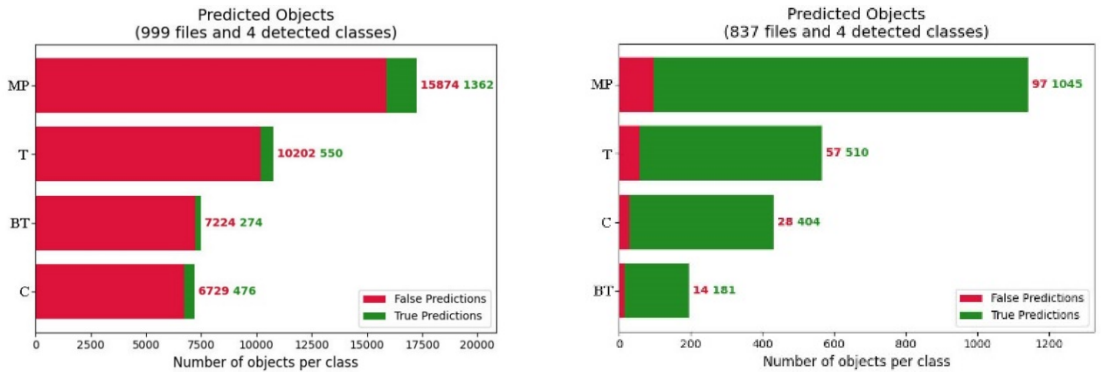


Fig. 13. Metrics evaluation validation of the best training model



(Confidence 0.001) (Confidence 0.321)

Fig. 14. Prediction performance on validation datasets

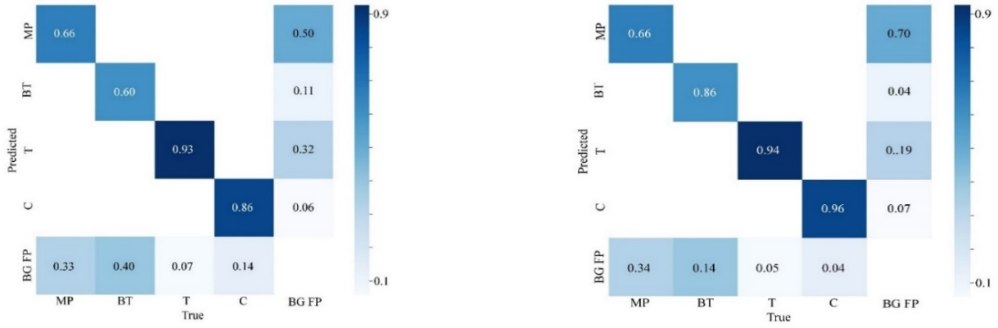
Table 8. The perfomance of model on validation dataset based on confidence value and IoU 0.6

	Class	Total Images	Labels	Detected Percentage	P	R	mAP @0.5	mAP @0.5:0.95
Confidence 0.001	***all***	999	2772	100%	0.798	0.697	0.726	0.286
	MP	999	1447	100%	0.737	0.591	0.634	0.234
	BT	999	299	100%	0.766	0.522	0.514	0.18
	T	999	550	100%	0.795	0.858	0.844	0.262
	C	999	476	100%	0.894	0.816	0.911	0.468
Confidence 0.321 (based on best TF1-Score)	***all***	999	2496	90.04%	0.813	0.817	0.826	0.356
	MP	999	1377	95.16%	0.747	0.619	0.706	0.293
	BT	999	190	63.55%	0.785	0.805	0.766	0.294
	T	999	521	94.73%	0.824	0.896	0.873	0.291
	C	999	408	85.71%	0.896	0.949	0.958	0.545

[a] MP: *Myzus persicae*, BT: *Bemisia tabaci*, T: Thrips, C: Caterpillar

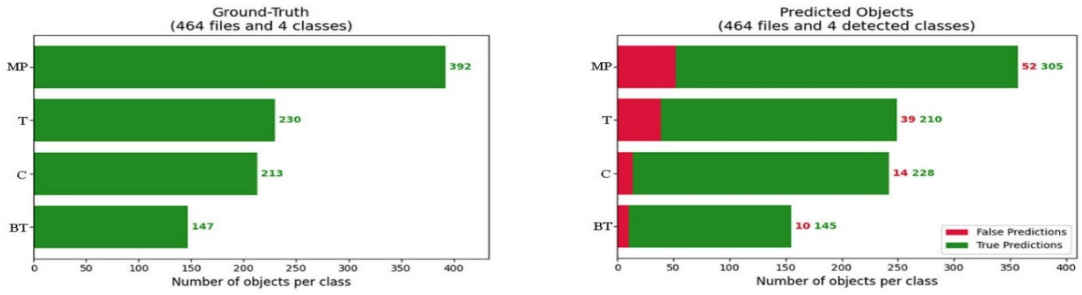
4.2. Evaluation of The Best Trained Model On Datasets Test

Out of a total of 499 test dataset image files, using a confidence threshold of 0.321 and IoU of 0.6, 464 images were identified as containing objects. Among the 1082 labels present, only 982 labels were successfully detected. This indicates that the trained models achieved a 90.93% accuracy in distinguishing the ground truth from the background. The confusion matrix of the best-trained model test on the test dataset for the entire ground-truth label is shown in Fig. 15 (test dataset). Class C exhibits the highest performance, with a true prediction rate of 96%, whereas Class MP has the lowest performance, with a true prediction rate of 66%. On average, the true prediction rate across all classes is 85.5%. Fig. 16 shows the detection performance of 464 image files that have detected the presence of objects. The comparison of the number of true predictions with false predictions shows that the best trained model can distinguish between each class very well.



(Validation dataset) (Test dataset)

Fig. 15. Confusion matrix best trained model



(Ground-Truth) (Predicted)

Fig. 16. Prediction Perfomance on Test dataset

Based on the mAP@0.5 metrics in Table 9, the T class has the lowest mAP, which is 67%, not much different from the percentage from the confusion matrix, while the highest is class C, with 98.1% mAP, also not too different from the confusion metrics data. The mAP of the detection test of the overall test datasets is 81.3%. This performance closely aligns with the average true prediction rate indicated by the confusion matrix. Comparison of the mAP@0.5 metrics and the confusion matrix data shows that the trained model exhibits outstanding performance. Meanwhile, when compared with the evaluation metrics on the previous validation dataset test, it shows that the trained model does not experience overfitting. Fig. 17, Fig. 18, Fig. 19, and Fig. 20 show the visualization of the detection of image data samples for each pest.

Table 9. Evaluation metrics best trained model on test dataset

Class	Images	Labels	Detected Labels	Detected Percentage	P	R	mAP@0.5	mAP@0.5:0.95
all	499	1080	982	90.93%	0.792	0.826	0.813	0.37
MP	499	448	392	87.50%	0.737	0.671	0.727	0.326
BT	499	173	147	84.97%	0.845	0.891	0.874	0.362
T	499	217	213	98.16%	0.659	0.77	0.67	0.217
C	499	242	230	95.04%	0.926	0.974	0.981	0.576

(Original) (Ground-Truth) (Prediction)

Fig. 17. MP detection

(Original) (Ground-Truth) (Prediction)

Fig. 18. BT detection

(Original) (Ground-Truth) (Prediction)

Fig. 19. T detection

(Original) (Ground-Truth) (Prediction)

Fig. 20. C detection

5. Conclusion

In this study, a detection model of pest that attacks Indonesian red chili pepper was developed based on the YOLOv5 algorithm. Using a primary dataset from a chili plantation in Bengkulu province, the model was trained using a fine-tuning mode of the YOLOv5s pre-trained model. Based on the difference in patience values, the best model obtained is the model on the 445th epoch with patience 100, and the

best confidence value is 0.321 on TF1 0.74. In retesting the pest detection on the train and validation dataset with IoU 0.6 and confidence 0.321, the mAP@0.5 were 72.5% and 82.6%. This mAP comparison showed that the mode is not overfitting. In the test dataset test, it was obtained that mAP@0.5 was 81.3% with the lowest 67% for thrips class and the highest 98.1% for caterpillar class (*Helicoverpa armigera*, *Spodoptera litura*). With an average pest object dimension of 4% of the image dimensions, this indicated that the model can detect tiny pests, such as Thrips and Silverleaf whitefly well. The best-trained model's best mAP@0.5 performance and speed are 82.6% is 20 ms/image or 50 fps on NVIDIA P100 GPU, it shows that the model can be used for real-time detection. In this study, mAP@0.5 was 81.3%, but it still has to be improved. Therefore suggestions for future research include how to improve mAP@0.5 and mAP@0.95. Furthermore, studies can be conducted to increase real-time performance so that it can be deployed on embedded devices with lower computational capabilities, such as the NVIDIA Jetson Nano.

Acknowledgment

This research is the first part of the Smart-robot weed control research supported by the University of Bengkulu through university assignment research program No. 1359/UN30.15/PO/2021.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.
Funding statement. This research was funded by University of Bengkulu with grant number 1359/UN30.15/PO/2021.
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

References

[1] M. Khalil, E. Kesumawati, and S. Zakaria, "Begomoviral disease rates and the implications to the growth and yield of chili plants (Capsicum annuum L.) at different elevations in Indonesia," *J. Appl. Hortic.*, vol. 22, no. 1, pp. 71–75, Apr. 2020, doi: 10.37855/JAH.2020.V22I01.14.

[2] Badan Pusat Statistik Indonesia, "Statistik tanaman sayuran dan buah-buahan semusim Indonesia 2018," *BPS-Statistics Indones.*, pp. 1–101, 2018, Accessed: April. 23, 2023. Available at : https://www.bps.go.id/publication/2019/10/07/9c5dede09c805bc38302ea1c/statistik-tanaman-sayuran-dan-buah---buahan-semusim-indonesia-2018.html.

[3] K. Subagyono *et al.*, *Budidaya dan Pascapanen Cabai Merah (Capsicum annuum L.)*. BPTP Jateng / KAN, 2010. Available at: https://repository.pertanian.go.id/handle/123456789/9109.

[4] P. R. Indonesia *et al.*, "Law of the Republic of Indonesia No. 12 of 1992 concerning: Plant Cultivation System," vol. 2010, no. 1, pp. 1–5, 1991. Available at : https://www.dpr.go.id/dokjdih/document/uu/628.pdf.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 779-788, 2016. doi: 10.1109/CVPR.2016.91.

[6] G. Jocher *et al.*, "ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation." Nov. 22, 2022. doi: 10.5281/ZENODO.7347926.

[7] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 6517–6525, Nov. 2017, doi: 10.1109/CVPR.2017.690.

[8] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *Comput. Vis. Pattern Recognit.*, pp. 1–6, Apr. 2018,. Available: https://arxiv.org/abs/1804.02767v1.

[9] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *Comput. Vis. Pattern Recognit.*, pp. 1–17, Apr. 2020, Accessed: May 10, 2023. [Online]. Available: https://arxiv.org/abs/2004.10934v1.

[10] J. L. Miranda, B. D. Gerardo, and B. T. Tanguilig III, "Pest Detection and Extraction Using Image Processing Techniques," *Int. J. Comput. Commun. Eng.*, vol. 3, no. 3, pp. 189–192, 2014, doi: 10.7763/IJCCE.2014.V3.317.

[11] Z. Bin Husin, A. Y. Bin Md Shakaff, A. H. Bin Abdul Aziz, and R. B. S. Mohamed Farook, "Feasibility study on plant chili disease detection using image processing techniques," *Proc. - 3rd Int. Conf. Intell. Syst. Model. Simulation, ISMS 2012*, pp. 291–296, 2012, doi: 10.1109/ISMS.2012.33.

[12] V. Khanaa and K. P. Thooyamani, "An Efficient Weed and Pest Detection System," *Indian J. Sci. Technol.*, vol. 8, no. 32, pp. 1–7, Nov. 2015, doi: 10.17485/IJST/2015/V8I32/87476.

[13] S. R. Huddar, S. Gowri, K. Keerthana, S. Vasanthi, and S. R. Rupanagudi, "Novel algorithm for segmentation and automatic identification of pests on plants using image processing," *2012 3rd Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2012*, pp. 1-5, 2012, doi: 10.1109/ICCCNT.2012.6396012.

[14] L. Deng *et al.*, "Application of agricultural insect pest detection and control map based on image processing analysis," *J. Intell. Fuzzy Syst.*, vol. 38, no. 1, pp. 379–389, Jan. 2020, doi: 10.3233/JIFS-179413.

[15] K. Thenmozhi and U. S. Reddy, "Image processing techniques for insect shape detection in field crops," *Proc. Int. Conf. Inven. Comput. Informatics, ICICI 2017*, pp. 699–704, May 2018, doi: 10.1109/ICICI.2017.8365226.

[16] P. Boissard, V. Martin, and S. Moisan, "A cognitive vision approach to early pest detection in greenhouse crops," *Comput. Electron. Agric.*, vol. 62, no. 2, pp. 81–93, Jul. 2008, doi: 10.1016/J.COMPAG.2007.11.009.

[17] Y. Li, C. Xia, and J. Lee, "Vision-based pest detection and automatic spray of greenhouse plant," *IEEE Int. Symp. Ind. Electron.*, pp. 920–925, 2009, doi: 10.1109/ISIE.2009.5218251.

[18] A. Ahmed, A. Ibrahim, and S. Hussein, "Detection of palm tree pests using thermal imaging: A review," *Stud. Comput. Intell.*, vol. 801, pp. 253–270, 2019, doi: 10.1007/978-3-030-02357-7_12.

[19] R. Pratheba, A. Sivasangari, and D. Saraswady, "Performance analysis of pest detection for agricultural field using clustering techniques," *2014 Int. Conf. Circuits, Power Comput. Technol. ICCPCT 2014*, pp. 1426–1431, Mar. 2014, doi: 10.1109/ICCPCT.2014.7054833.

[20] N. Vinushree, B. Hemalatha, and V. K. Kaliappan, "Efficient kernel-based fuzzy C-means clustering for pest detection and classification," *Proc. - 2014 World Congr. Comput. Commun. Technol. WCCCT 2014*, pp. 179–181, 2014, doi: 10.1109/WCCCT.2014.61.

[21] P. Nagababu, M. R. Reddy, K. Anila, and M. Srujana, "Early Pest Detection From Crop Using Image Processing and Computational Intelligence," *Interantional J. Sci. Res. Eng. Manag.*, vol. 06, no. 06, pp. 1–5, 2022, doi: 10.55041/ijsrem14606.

[22] M. Bayat, Mahdi Abbasi, and Ali Yosefi, "Improvement of Pest Detection Using Histogram Adjustment Method and Gabor Wavelet," *J. Asian Sci. Res.*, vol. 6, no. 2, pp. 24–33, Feb. 2016, doi: 10.18488/JOURNAL.2/2016.6.2/2.2.24.33.

[23] M. A. Ebrahimi, M. H. Khoshtaghaza, S. Minaei, and B. Jamshidi, "Vision-based pest detection based on SVM classification method," *Comput. Electron. Agric.*, vol. 137, pp. 52–58, May 2017, doi: 10.1016/J.COMPAG.2017.03.016.

[24] P. Ashok, J. Jayachandran, S. Sankara Gomathi, and M. Jayaprakasan, "Pest detection and identification by applying color histogram and contour detectionby Svm model," *Int. J. Eng. Adv. Technol.*, vol. 8, no. 3 Special Issue, pp. 463–467, 2019. Available at: https://www.ijeat.org/portfolio-item/c10970283s19/.

[25] A. Dey, D. Bhoumik, and K. Dey, "Automatic Detection of Whitefly Pest using Statistical Feature Extraction and Image Classification Methods," Int. Res. J. Eng. Technol., vol. 03, no. 06, pp. 1–10, 2016, Available at: https://www.irjet.net/archives/V3/i9/IRJET-V3I9171.pdf .

[26] F. Faithpraise, B. Philip, Y. Rupert, O. J, F. Bassey, and C. Chris, "Automatic plant pest detection & recognition using k-means clustering algorithm & correspondence filters," Int. J. Adv. Biotechnol. Res., vol. 4, no. 2, pp. 1052–1062, 2013. Available at: http://sro.sussex.ac.uk/49042/1/IJABR-V4I2-2013-02.pdf.

[27] M. Türkoğlu and D. Hanbay, "Plant disease and pest detection using deep learning-based features," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 27, no. 3, pp. 1636–1651, Jan. 2019, doi: 10.3906/elk-1809-181.

[28] V. Malathi and M. P. Gopinath, "Classification of pest detection in paddy crop based on transfer learning approach," *Acta Agriculturae Scandinavica, Section B — Soil & Plant Science*, vol. 71, no. 7, pp. 552–559, 2021, doi: 10.1080/09064710.2021.1874045.

[29] A. Fuentes, S. Yoon, S. C. Kim, and D. S. Park, "A Robust Deep-Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition," *Sensors 2017,* vol. 17, no. 9, p. 2022, Sep. 2017, doi: 10.3390/S17092022.

[30] L. Jiao, S. Dong, S. Zhang, C. Xie, and H. Wang, "AF-RCNN: An anchor-free convolutional neural network for multi-categories agricultural pest detection," *Comput. Electron. Agric.*, vol. 174, p. 105522, Jul. 2020, doi: 10.1016/J.COMPAG.2020.105522.

[31] D. Li *et al.*, "A Recognition Method for Rice Plant Diseases and Pests Video Detection Based on Deep Convolutional Neural Network," *Sensors 2020,* vol. 20, no. 3, p. 578, Jan. 2020, doi: 10.3390/S20030578.

[32] W. Li, D. Wang, M. Li, Y. Gao, J. Wu, and X. Yang, "Field detection of tiny pests from sticky trap images using deep learning in agricultural greenhouse," *Comput. Electron. Agric.*, vol. 183, p. 106048, Apr. 2021, doi: 10.1016/J.COMPAG.2021.106048.

[33] S. Li, X. Zhao, D. Patel, N. Bhatt, and A. Professor, "Improved accuracy of pest detection using augmentation approach with Faster R-CNN," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1042, no. 1, p. 012020, Jan. 2021, doi: 10.1088/1757-899X/1042/1/012020.

[34] K. Zhang, Q. Wu, and Y. Chen, "Detecting soybean leaf disease from synthetic image using multi-feature fusion faster R-CNN," *Comput. Electron. Agric.*, vol. 183, p. 106064, Apr. 2021, doi: 10.1016/J.COMPAG.2021.106064.

[35] L. Jiao, C. Xie, P. Chen, J. Du, R. Li, and J. Zhang, "Adaptive feature fusion pyramid network for multi-classes agricultural pest detection," *Comput. Electron. Agric.*, vol. 195, p. 106827, Apr. 2022, doi: 10.1016/J.COMPAG.2022.106827.

[36] T. L. Lin, H. Y. Chang, and K. H. Chen, "Pest and Disease Identification in the Growth of Sweet Peppers using Faster R-CNN," *2019 IEEE Int. Conf. Consum. Electron. - Taiwan, ICCE-TW 2019*, pp. 1-2 May 2019, doi: 10.1109/ICCE-TW46550.2019.8991893.

[37] J. Liu and X. Wang, "Early recognition of tomato gray leaf spot disease based on MobileNetv2-YOLOv3 model," *Plant Methods*, vol. 16, no. 1, pp. 1–16, Jun. 2020, doi: 10.1186/S13007-020-00624-2.

[38] P. V. Bhatt, S. Sarangi, and S. Pappula, "Detection of diseases and pests on images captured in uncontrolled conditions from tea plantations," *Proceedings Volume 11008, Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping IV*, vol. 11008, pp. 73–82, May 2019, doi: 10.1117/12.2518868.

[39] J. W. Chen, W. J. Lin, H. J. Cheng, C. L. Hung, C. Y. Lin, and S. P. Chen, "A Smartphone-Based Application for Scale Pest Detection Using Multiple-Object Detection Methods," *Electron. 2021,* vol. 10, no. 4, p. 372, Feb. 2021, doi: 10.3390/ELECTRONICS10040372.

[40] H. Kuzuhara, H. Takimoto, Y. Sato, and A. Kanagawa, "Insect Pest Detection and Identification Method Based on Deep Learning for Realizing a Pest Control System," *2020 59th Annu. Conf. Soc. Instrum. Control Eng. Japan, SICE 2020*, pp. 709–714, Sep. 2020, doi: 10.23919/SICE48898.2020.9240458.

[41] B. Liang, S. Wu, K. Xu, and J. Hao, "Butterfly detection and classification based on integrated YOLO algorithm," *Adv. Intell. Syst. Comput.*, vol. 1107 AISC, pp. 500–512, 2020, doi: 10.1007/978-981-15-3308-2_55.

[42] K. R. B. Legaspi, N. W. S. Sison, and J. F. Villaverde, "Detection and Classification of Whiteflies and Fruit Flies Using YOLO," *2021 13th Int. Conf. Comput. Autom. Eng. ICCAE 2021*, pp. 1–4, Mar. 2021, doi: 10.1109/ICCAE51876.2021.9426129.

[43] N. Mamdouh and A. Khattab, "YOLO-Based Deep Learning Framework for Olive Fruit Fly Detection and Counting," *IEEE Access*, vol. 9, pp. 84252–84262, 2021, doi: 10.1109/ACCESS.2021.3088075.

[44] D. J. A. Rustia *et al.*, "Automatic greenhouse insect pest detection and recognition based on a cascaded deep learning classification method," *J. Appl. Entomol.*, vol. 145, no. 3, pp. 206–222, Apr. 2021, doi: 10.1111/JEN.12834.

[45] Z. Tang, Z. Chen, F. Qi, L. Zhang, and S. Chen, "Pest-YOLO: Deep Image Mining and Multi-Feature Fusion for Real-Time Agriculture Pest Detection," *Proc. - IEEE Int. Conf. Data Mining, ICDM*, vol. 2021-December, pp. 1348–1353, 2021, doi: 10.1109/ICDM51629.2021.00169.

[46] S. Verma, S. Tripathi, A. Singh, M. Ojha, and R. R. Saxena, "Insect Detection and Identification using YOLO Algorithms on Soybean Crop," *IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON*, vol. 2021-December, pp. 272–277, 2021, doi: 10.1109/TENCON54134.2021.9707354.

[47] M. Zha, W. Qian, W. Yi, and J. Hua, "A Lightweight YOLOv4-Based Forestry Pest Detection Method Using Coordinate Attention and Feature Fusion," *Entropy 2021*, vol. 23, no. 12, p. 1587, Nov. 2021, doi: 10.3390/E23121587.

[48] D. Li, F. Ahmed, N. Wu, and A. I. Sethi, "YOLO-JD: A Deep Learning Network for Jute Diseases and Pests Detection from Images," *Plants 2022*, vol. 11, no. 7, p. 937, Mar. 2022, doi: 10.3390/PLANTS11070937.

[49] J. Liu and X. Wang, "Tomato Diseases and Pests Detection Based on Improved Yolo V3 Convolutional Neural Network," *Front. Plant Sci.*, vol. 11, p. 521544, Jun. 2020, doi: 10.3389/FPLS.2020.00898.

[50] E. Önler, "Real Time Pest Detection Using YOLOv5," *Int. J. Agric. Nat. Sci.*, vol. 14, no. 3, pp. 232–246, Dec. 2021, Available at: https://www.ijans.org/index.php/ijans/article/view/550.

[51] I. Ahmad *et al.*, "Deep Learning Based Detector YOLOv5 for Identifying Insect Pests," *Appl. Sci. 2022*, vol. 12, no. 19, p. 10167, Oct. 2022, doi: 10.3390/APP121910167.

[52] M. G. Selvaraj *et al.*, "AI-powered banana diseases and pest detection," *Plant Methods*, vol. 15, no. 1, pp. 1–11, Aug. 2019, doi: 10.1186/S13007-019-0475-Z.

[53] S. A. Burhan, D. S. Minhas, D. A. Tariq, and M. Nabeel Hassan, "Comparative Study of Deep Learning Algorithms for Disease and Pest Detection in Rice Crops," *Proc. 12th Int. Conf. Electron. Comput. Artif. Intell. ECAI 2020*, pp. 1-5, Jun. 2020, doi: 10.1109/ECAI50035.2020.9223239.

[54] T. T. Nguyen, Q. T. Vien, and H. Sellahewa, "An Efficient Pest Classification In Smart Agriculture Using Transfer Learning," *EAI Endorsed Trans. Ind. Networks Intell. Syst.*, vol. 8, no. 26, pp. 1–8, Jan. 2021, doi: 10.4108/EAI.26-1-2021.168227.

[55] L. Liu *et al.*, "PestNet: An End-to-End Deep Learning Approach for Large-Scale Multi-Class Pest Detection and Classification," *IEEE Access*, vol. 7, pp. 45301–45312, 2019, doi: 10.1109/ACCESS.2019.2909522.

[56] P. P. J. Roosjen, B. Kellenberger, L. Kooistra, D. R. Green, and J. Fahrentrapp, "Deep Learning For Automated Detection Of Drosophila Suzukii: Potential For UAV-Based Monitoring," *Pest Manag. Sci.*, vol. 76, no. 9, pp. 2994–3002, Sep. 2020, doi: 10.1002/PS.5845.

[57] B. Kellenberger, D. Marcos, and D. Tuia, "Detecting mammals in UAV images: Best practices to address a substantially imbalanced dataset with deep learning," *Remote Sens. Environ.*, vol. 216, pp. 139–153, Oct. 2018, doi: 10.1016/J.RSE.2018.06.028.

[58] Z. Lyu, H. Jin, T. Zhen, F. Sun, and H. Xu, "Small Object Recognition Algorithm of Grain Pests Based on SSD Feature Fusion," *IEEE Access*, vol. 9, pp. 43202–43213, 2021, doi: 10.1109/ACCESS.2021.3066510.

[59] L. Liu *et al.*, "Deep Learning Based Automatic Multiclass Wild Pest Monitoring Approach Using Hybrid Global and Local Activated Features," *IEEE Trans. Ind. Informatics*, vol. 17, no. 11, pp. 7589–7598, Nov. 2021, doi: 10.1109/TII.2020.2995208.

[60] R. Wang, L. Jiao, C. Xie, P. Chen, J. Du, and R. Li, "S-RPN: Sampling-balanced region proposal network for small crop pest detection," *Comput. Electron. Agric.*, vol. 187, p. 106290, Aug. 2021, doi: 10.1016/J.COMPAG.2021.106290.

[61] M. Turkoglu, B. Yanikoğlu, and D. Hanbay, "PlantDiseaseNet: convolutional neural network ensemble for plant disease and pest detection," *Signal, Image Video Process.*, vol. 16, no. 2, pp. 301–309, Mar. 2022, doi: 10.1007/S11760-021-01909-2.