Disease Detection of Solanaceous Crops Using Deep Learning for Robot Vision

A. H. Nurul Hidayah [1], A. R. Syafeeza [2,*], Norazlina Abdul Razak [3], Wira Hidayat Mohd Saad [4], Y. C. Wong [5], A. Azureen Naja [6]
[1, 2, 3, 4,5,6] Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer (FKEKK), Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia
Email: [1] hidayahhalim.nh@gmail.com, [2] syafeeza@utem.edu.my, [3] norazlina@utem.edu.my, [4] wira_yugi@utem.edu.my, [5] ycwong@utem.edu.my, [6] azureen.naja98@gmail.com
*Corresponding Author

Abstract— **Traditionally, the farmers manage the crops from the early growth stage until the mature harvest stage by manually identifying and monitoring plant diseases, nutrient deficiencies, controlled irrigation, and controlled fertilizers and pesticides. Even the farmers have difficulty detecting crop diseases using their naked eyes due to several similar crop diseases. Identifying the correct diseases is crucial since it can improve the quality and quantity of crop production. With the advent of Artificial Intelligence (AI) technology, all crop-managing tasks can be automated using a robot that mimics a farmer's ability. However, designing a robot with human capability, especially in detecting the crop's diseases in real-time, is another challenge to consider. Other research works are focusing on improving the mean average precision and the best result reported so far is 93% of mean Average Precision (mAP) by YOLOv5. This paper focuses on object detection of the Convolutional Neural Network (CNN) architecture-based to detect the disease of solanaceous crops for robot vision. This study's contribution involved reporting the developmental specifics and a suggested solution for issues that appear along with the conducted study. In addition, the output of this study is expected to become the algorithm of the robot's vision. This study uses images of four crops (tomato, potato, eggplant, and pepper), including 23 classes of healthy and diseased crops infected on the leaf and fruits. The dataset utilized combines the public dataset (PlantVillage) and self-collected samples. The total dataset of all 23 classes is 16580 images divided into three parts: training set, validation set, and testing set. The dataset used for training is 88% of the total dataset (15000 images), 8% of the dataset performed a validation process (1400 images), and the rest of the 4% dataset is for the test process (699 images). The performances of YOLOv5 were more robust in terms of 94.2% mAP, and the speed was slightly faster than Scaled-YOLOv4. This object detection-based approach has proven to be a promising solution in efficiently detecting crop disease in real-time.**

Keywords— Deep Learning; Convolutional Neural Network; Object Detection; YOLOv5; Solanaceous Crops; Crops disease.

I. INTRODUCTION

Agriculture has long been a vital economic and social sector. It is difficult for manpower to accurately detect crop diseases at an early stage to improve the quality and quantity of crop production. The causes of crop diseases are more likely due to many factors, such as shifting weather, lack of nutrition, and pest attacks. In general, crop disease detection is carried out manually using visual inspection or microscope techniques, which are time-intensive and prone to inaccuracy leading to different human vision and error information [1]. Mistakes or missteps are usually unavoidable when using manpower, especially when classifying the plant's type of disease because human eyes are prone to errors and require a time-consuming diagnosis. However, disease and pest control challenges still haunt some local farmers [1]. As a result, disease detection requires regular crop monitoring throughout the growing period. One practical approach to resolving these issues is the development of an automated agricultural robot capable of detecting disease and monitoring the field condition by moving around the field. The development of the robot's vision is a hurdle. We want the robot's vision to mimic the sight of human eyes [2]. The robot is expected to improve operational accuracy in the farming industry [3]. On top of that, robot motion planning in real-time application is also one of the key areas of research in computer science and computer geometry [4].

Artificial Intelligence (AI) is a suitable algorithm for robot vision if we aim for an intelligent system. Since AI focuses on developing computer software to make computing tasks smarter, AI research applications' ultimate focus is to develop computational approaches for intelligent behaviour [5]. The demand for an intelligent system with real-time control in manufacturing processes and productions is increasing rapidly [6]. The increasing powers of computers and embedded computing have further contributed to AI advancement [7]. The most common application of AI in computer vision is face recognition [8] which is heavily deployed on the smartphone. AI technology is widely used worldwide and positively impacts manufacturing, healthcare, and agriculture [9,10]. Agriculture is an extreme industry, with 30.7% contributing to economic progress [11]. Agriculture is a dynamic sector in which it is impossible to generalize situations to propose a standard solution [12]. In terms of accuracy and robustness, AI is at its best in supporting agricultural systems. An efficient technique of utilizing an AI system can help farmers to monitor their crops, including detecting any crop disease [13].

Deep learning is an AI technique that simulates how humans acquire knowledge. Deep learning is a critical component of information data science, covering statistics and prediction [14, 15]. As the most empowered machine learning technique, deep learning has been applied in various fields, including robotics and agriculture [16]. Today, deep learning-enabled developments in computer vision have led to a situation where disease diagnosis is dependent on automatic recognition using a deep learning-based monitoring process [10]. Deep Learning models perform exceptionally well in prediction and classification because of their large learning capacity and highly hierarchical structure [11,12]. They are also flexible and adaptable to various highly complex (from a data analysis perspective) challenges [17].

CNN is one of the most prominent deep learning approaches [18, 19]. CNN is an algorithm of deep learning which take images as input and can extract significant features automatically to learn and ultimately classify the input images to their suitable output class [20, 21]. Object detection acts better for computer vision techniques regarding network architecture, training techniques, and optimization functions [22, 23]. Object detection has several models, such as YOLO [24]. Faster R-CNN used to be the main model for object detection. However, the inference speed resulting from Faster R-CNN still does not meet the one derived by YOLO [22]. YOLO is an object detection method that acts as a real-time object detector [25]. Joseph Redmon created the original model of YOLO (You Only Look Once) in a custom-built framework called Darknet. Darknet is a very adaptable research framework written in low-level languages that have created computer vision that can achieve the most significant real-time object detectors, including YOLO, YOLOv2, YOLOv3, YOLOv4, and recently, the new one is YOLOv5 [26, 27].

In a previous study, Roy et al. (2021) reported that YOLOv3 reached 78% of mAP compared to YOLOv4 and 86% in detecting various plant disease classes [28]. Wu et al. (2021) used YOLOv3 and YOLOv4 toward 2670 images of an augmented dataset and achieved an accuracy above 90% for each model [29]. Thuan et al. (2021) YOLOv5 shows the model fast and reached high accuracy of 93% on train 3422 images with 100 epochs. One epoch cycle only takes around 20 seconds to complete [30]. Other related research works on robot vision, such as [31] implemented U-Net architecture to detect the leaf of the bean images captured in uncontrolled environmental conditions. The accuracy achieved was 91.02%.

The research contribution is to present a detailed process and a suggested solution for problems that arose throughout the development of object detection algorithms (YOLOv5 and Scaled-YOLOv4) to detect the diseases on the leaf and fruit of solanaceous crops. The output of this study is expected to become the algorithm of the robot's vision in real-time. The performance comparison of these models is also analyzed in terms of precision, recall, mean average precision (mAP) and training time. The detailed development for the mentioned purpose was discussed in detail.

This paper starts with the related work and theory, methods for completing the whole simulation, results, and performance of the YOLOv5 model compared with the previous YOLO (Scaled-YOLOv4), and the conclusion of the overall works.

II. RELATED WORKS

A. Deep Learning

Deep Learning is a type of machine learning that extends traditional machine learning by adding more "depth" (complexity) to the model and modifying the data using several features that allow data to be represented in a hierarchical form through multiple levels of abstraction [32]. If large datasets describing the problem exist, these complex models used in deep learning can reduce the errors, especially in regression problems, and improve classification accuracy [33].

Deep learning has several layers, such as convolution, fully connected, pooling, etc. The main feature of deep learning is that the features in these layers are learned from data instead of just designed by engineers through some learning procedures [34]. The organization of the layers will create different network architectures, such as Convolutional Neural Networks, Recursive Neural Networks, Unsupervised Pre-trained Networks, and Recurrent Neural Networks [35].

B. Convolutional Neural Network

Unlike other Deep Learning architectures, such as Recurrent Neural Networks or Long-Short Term Memory, in image and video applications, CNN architecture is preferable as the architecture design of CNN focuses on the spatial correlation of pixel intensities more efficient for images [36]. CNN model provides an essential visual feature extractor for crop diseases. It consists of three operation layers: convolutional layers, max-pooling layers, and fully connected layers that act as automatic feature extractors in one single module during training [37]. Then it employs 2D convolutional layers, making this architecture more ideal for interpreting 2D data, such as images, than other machine learning (ML) techniques [38]. CNN overcomes the limitation of the manual feature extraction process carried out by traditional ML techniques and can handle vast amounts of data [39]. CNN of its model extracts the data directly from images. CNN architecture consists of numerous layers that perform image processing operations. These layers include input, multiple hidden, and output layers. The hidden layers typically comprise several convolutional layers, pooling layers, and a set of fully connected layers to perform the classification task [40].

C. Object Detection

Traditional object detection algorithms use handcrafted designs and simple trainable architectures [41]. Their performance is easily stagnated by developing complicated ensembles that mix several low-level picture features with high-level information through object detectors and image classifiers. A traditional object detection architecture consists of region candidate generation, feature extractions, and classification tasks. The detection result from the classifier is fed onto the Non-Maximum Suppression (NMS) algorithm to optimize the results by combining multiple overlapping bounding boxes [42]. With the rapid advancement of deep learning, more powerful tools that can learn semantic, high-level, and deeper features are being offered to address the issues that older systems have [43].

Object detection is a computer vision technique for identifying and locating objects in images and videos [44]. Object detection has been an active research area in computer vision for decades [45]. It deals with instances of visual detection of any specific class, such as detecting humans, vehicles or animals. Object detection may count multiple objects in a scene, identify and trace their precise locations, and accurately label them with this type of identification and localization [46]. In short, an object detection algorithm allows us to locate and predict the specific location of the desired object using bounding boxes [47]. Object detection models can be divided into two categories: a one-stage target detection framework based on region proposal or a two-stage target detection framework based on regression [48]. One of the examples of a one-stage object detection method is the variants of the YOLO family. YOLO has been widely applied across various industries due to its suitability to be implemented in embedded controller systems through transfer learning, plus its ability to be a self-adaptive algorithm [49]. An adaptive neural network is usually applied when there is minimal prior knowledge of the environments [50]. The recent version of YOLO is known as YOLOv5. Since it was first introduced until now, many researchers and industry players have deployed the YOLOv5 model for tasks such as crop recognition, yield estimation and many more [47], [51]-[55].

D. *YOLOv5*

Glenn Jocher, the founder of Ultralytics, released an open-source implementation of the YOLOv5 model in June 2020 [56]. It is the first in the YOLO family to be released without a paper and is still in "continuing development" on its repository. The YOLOv5 switched from Darknet to Pytorch, achieving 140 frames per second in the Tesla P100, compared to 50 frames per second in the YOLOv4. YOLOv5 is suitable for real-time object detection and has many advantages over traditional object detectors [58]. YOLOv5 offers the same benefits as YOLOv4 and has a nearly identical architecture. Compared to YOLOv4 and YOLOv5, it is easier to train and detect the object [57].

The backbone, head, and detection are the three fundamental components of YOLOv5. A CNN serves as the backbone, gathering and shaping visual features at various levels of granularity. The YOLOv5 uses the Center and Scale Prediction (CSP) bottleneck to create image features. The detection is a method that localizes the bounding box, labels class prediction at the image, and uses features from the head of the structure [59]. The head comprises layers that aggregate image characteristics before being sent into a prediction algorithm. The PA-NET is also implemented in YOLOv5 for feature aggregation. Fig. 1 shows the architecture of YOLOv5.

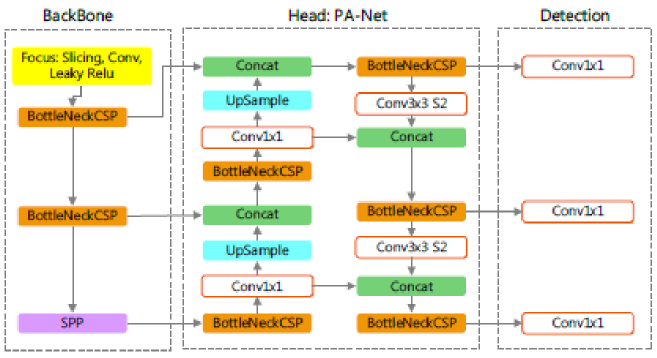


Fig. 1. YOLOv5 Architecture

III. METHOD

A. *Dataset*

Adequate dataset samples are requisite for all deep learning methods to obtain a good generalization result. The images for the dataset were collected from a mobile phone camera, downloaded from the internet on Kaggle, GitHub of Plant Village dataset, and searched by disease of solanaceous crops name on Google Image. After collecting the images, data separation was carried out according to their classes. The classes were named based on either healthy or diseases that infected the leaf and fruit. The healthy image was also collected for this dataset to make the system differentiate whether the solanaceous crops were healthy or infected by the disease. The initial number of samples for the dataset is around 300 images for each class, and the total number of classes consists of 23. These classes are shown in TABLE 1.

TABLE 1 DETAILS OF DATASET CLASSES

Solanaceous Crops	**Disease and Healthy Fruit and Leaf Name**
Pepper	Chili___Healthy_fruit Chili___Healthy_leaf Chili___Anthracnose_fruit Chili___Bacterial_leaf_spot Chili___Mosaic_virus_leaf
Eggplant	Eggplant___Healthy_fruit Eggplant___Healthy_leaf Eggplant___Fruit_rot Eggplant___Cercospora_leaf_spot Eggplant___Colorado_potato_beetle
Potato	Potato___Healthy_fruit Potato___Healthy_leaf Potato___Common_scab_fruit Potato___Alternaria_solani_leaf Potato___Phytopthora_infestans_leaf
Tomato	Tomato___Healthy_fruit Tomato___Healthy_leaf Tomato___Anthracnose_fruit Tomato___Early_blight_leaf Tomato___Late_blight_leaf Tomato___Leaf_mold Tomato___Tomato_yellow_leaf_curl_virus Tomato___Bacterial_spot_leaf

B. *Annotate and Labelled*

After collecting the dataset, the images were uploaded to the Roboflow.ai website, which provides various functions to improve our dataset. Roboflow also allows access to labeling datasets, annotating images, preprocessing, augmentation process, and other beneficial functions to handle the dataset. Some function mentioned has been applied in this project. Roboflow is a free online platform for labeling and annotation instead of downloading other software to your computer. The purpose is to secure your dataset and enable access on several devices, such as tablets or smartphones.

At the early stage, the images were split into a training set (70%), validation set (20%), and testing set (10%) after uploading the images to Roboflow. Then, the images are labeled by their class name and annotated by drawing a bounding box to identify the data features in the area of the diseases and healthy leaf and fruit of crops. Fig. 2 shows the annotation and labelling process on Roboflow.

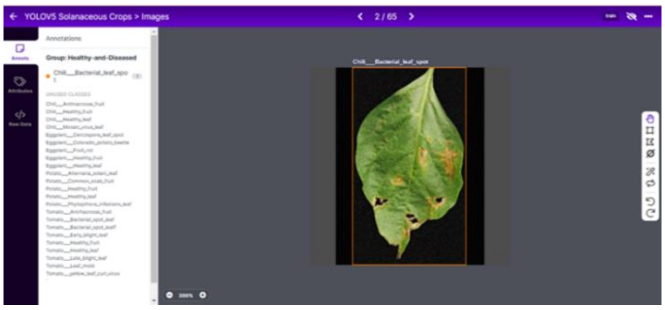
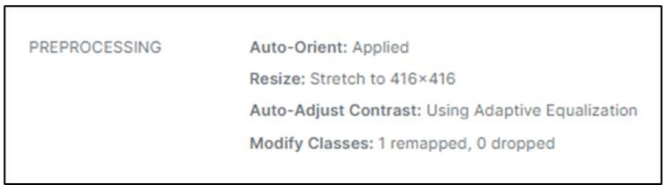

YOLOV5 Solanaceous Crops > Images
2 / 65
Annotations
Group: Healthy-and-Diseased
Chili__Bacterial_leaf_spot

Fig. 2. Annotation and Labelling Process on Roboflow

C. Preprocessing

Transforming the data from raw data to the desired format suitable for the YOLOv5 preprocessing process must occur. This procedure eliminates data discrepancies or duplication, which could otherwise degrade the accuracy of a model. Data preprocessing also guarantees that no inaccurate or lost values exist because of human mistakes or bugs. By adding image alterations to all the images in this dataset, training time can be saved, and performance can be improved. EXIF rotations should be removed, and pixel order should be standardized with the help of auto-oriented applied for this preprocessing method. The image resizes to 416×416 to standardize the size, and the smaller file size can help for faster training. Auto adjust contrast can help the model to detect edges around the object. Roboflow has a function that allows you to modify the labeled samples that have been labeled mistakenly. Fig. 3 shows the example of preprocessing in Roboflow.

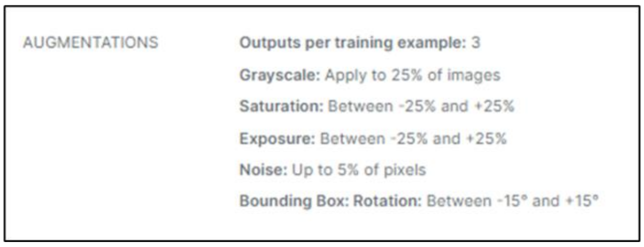


PREPROCESSING	
	Auto-Orient: Applied
	Resize: Stretch to 416×416
	Auto-Adjust Contrast: Using Adaptive Equalization
	Modify Classes: 1 remapped, 0 dropped

Fig. 3. Preprocessing in Roboflow

D. Data augmentation

In deep learning, the key feature to improve model performance accuracy is increasing the number of samples to train the system effectively. At the early stages of this process, the dataset only contains 300 images for each class. It is considered a small dataset and may lead to low-performance accuracy at the end of the training process. An augmentation process was applied to increase the number of samples to overcome this problem.



AUGMENTATIONS	
	Outputs per training example: 3
	Grayscale: Apply to 25% of images
	Saturation: Between -25% and +25%
	Exposure: Between -25% and +25%
	Noise: Up to 5% of pixels
	Bounding Box: Rotation: Between -15° and +15°

Fig. 4. Augmentation Process Selected in Roboflow

The augmentation process was also carried out at the Roboflow.ai website, providing an auto-generating function for the augmentation image. Based on Fig. 4, these are the augmentation techniques selected to increase the number of datasets. Random rotation augmentation can aid the model in detecting the object, even if the images are not precisely aligned. The same goes for bounding box rotation. Combining the methods can help the model stay sturdy to the camera roll in real-time usage. Grayscale, saturation, and exposure can help increase the various colors of the images so that when it is tested in real-time, it has learned to detect the object even in different lighting. Adding noise to the images can prevent overfitting and against adversarial attacks. Fig. 5 shows the augmented images auto-generated after selecting some techniques to increase the dataset.

After applying preprocessing and augmentation method, the dataset will be auto-generated, and random images will be selected for the training process. The dataset will expand to three times from the early total of images. At the end of this process, the dataset increases from 6900 images to 16580 images. The training set, validation set, and test set are separated into 88% (15000 images) for training, 8% (1400 images) for validation, and 2% (699 images) for the testing process. This train, valid, and test set is the final subset of the dataset applied to evaluate this project's performance.

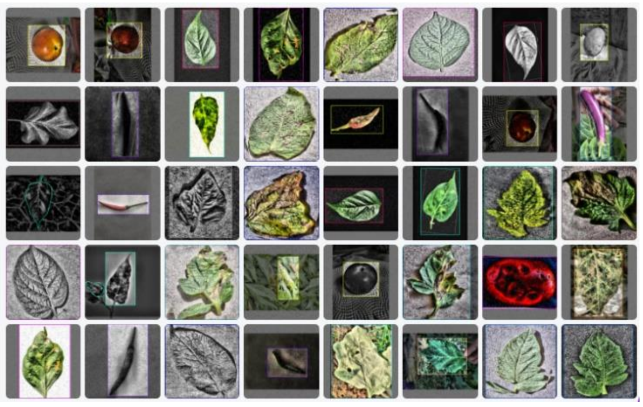


Fig. 5. Augmented images auto-generated from Roboflow

E. System Flowchart in Google Colab notebook

Fig. 6 depicts the flowchart of the whole simulation process in the Google Colab notebook. It consists of three main parts: set up the Google Colab environment, train the YOLOv5 model, and evaluate the performance.

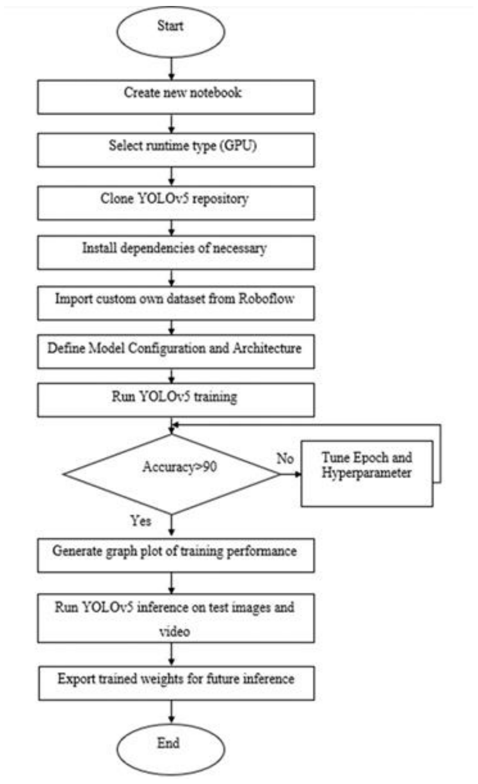

Start
Create new notebook
Select runtime type (GPU)
Clone YOLOv5 repository
Install dependencies of necessary
Import custom own dataset from Roboflow
Define Model Configuration and Architecture
Run YOLOv5 training
Accuracy>90
No
Tune Epoch and Hyperparameter
Yes
Generate graph plot of training performance
Run YOLOv5 inference on test images and video
Export trained weights for future inference
End

Fig. 6. System Flowchart in Google Colab notebook

1) Google Colab notebook

Google Colab provides free access to their Graphical Processing Unit (GPU). Users must select the Runtime operation, either Tensor Processing Unit (TPU), GPU or None. However, the upgrade version, Google Colab Pro, provides a random GPU, either Tesla T4 or Tesla P100.

2) Training YOLOv5 model

These are the steps involved in training the YOLOv5 model:

a) Installing the YOLOv5 environment

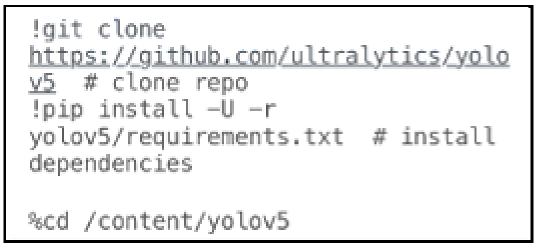
YOLOv5 pre-trained model repository provided by Ultralytics GitHub was used to train the dataset and provide the library dependencies. PyTorch requires the libraries before training the model. This step involves cloning the YOLOv5 repository before the Installation of the library dependency can be carried out. Fig. 7 shows the example of the cloning and installing process.

b) Download custom object detection in YOLOv5 format from Roboflow

After generating the augmented samples on Roboflow, a link was copied from the 'YOLOv5 PyTorch' to import the dataset. Before starting the training process, the augmented dataset from Roboflow is imported into the Google Colab notebook. It should be noted that the Ultralytics implementation supports a YAML file that specifies the location of the training and test data.

c) Define YOLOv5 Model Configuration and Architecture

Then, the YOLOv5 model configuration was defined by creating a YAML script that specifies the parameters for the YOLOv5 model, such as the number of classes, anchors, and backbone layers.


```
!git clone
https://github.com/ultralytics/yolo
v5  # clone repo
!pip install -U -r
yolov5/requirements.txt  # install
dependencies

%cd /content/yolov5
```

Fig. 7. The cloning and installing process

d) Training Custom YOLOv5 Detector

The training process is started when all the previous steps have been followed. This work uses the YOLOv5's model that runs a parameter of 100 epochs with 16 batch sizes and an input image size of 416. The training process will take around two to three hours to complete.

3) Evaluate the performance

Once the training process has been completed, the trained model's performance will be evaluated through the test images, whether it reaches 90% or above. If not, the training process is revoked by tuning the number of epochs and hyperparameters. The test images and videos used in this process are images that have never been seen during training. The training performance is evaluated through a plotted graph, including time taken to finish the training process, precision, recall, and mean average precision (mAP). The test images and video are verified to check the model's performance in detecting the disease of solanaceous crops. This training model can be used in real-time detection by exporting the trained weights of the network. The file can be kept in Google Drive for future use and deployed into real-world devices such as webcams, Raspberry Pi, Jetson Nano, mobile phones, and other supported devices.

IV. RESULT AND ANALYSIS

1) Model Comparison

The training was performed on 16580 images using 100 epochs and 16 batch sizes on the YOLOv5 and Scaled-YOLOv4. The performance of YOLOv5 was compared with Scaled-YOLOv4. TABLE II shows the comparison of the pretrained model used in this project.

TABLE II COMPARISON PRETRAINED MODEL YOLOV5S AND SCALED-YOLOV4

Model	YOLOv5s	Scaled -YOLOv4
Backbone	CSPDarknet	CSPDarknet53
Neck	PANet	PAN+SPP
Layers	283	334
Parameter (million)	7.2M	53M
Library Framework	PyTorch	PyTorch

Ultralytics supports numerous YOLOv5 architectures, known as P5 models, which differ primarily in size: YOLOv5n (nano), YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), YOLOv5x (extra-large). This project uses the YOLOv5s pretrained model to perform the object detection training because it is among the fastest model of P5 models [60]. Meanwhile, this proposed model compared with Scaled-YOLOv4 because among of YOLOv4 family, Scaled-YOLOv4 obtained record-breaking performance on the COCO benchmark [61]. YOLOv5s was compared to Scaled-YOLOv4 to prove this proposed model's effectiveness and robustness in detecting the solanaceous crops' disease. The backbone of YOLOv5s is CSPDarknet, and the neck uses PANet. For Scaled-YOLOv4, CSPDarknet53 is used as its backbone and PAN+SPP for the neck. YOLOv5s has 283 layers, and the parameter of this model is 7.2 million. Scaled-YOLOv4 has 334 layers and 53 million parameters. Both of these models used the PyTorch Library framework for their implementation.

2) Performance Evaluation

The plotted graph illustrates the time spent between the YOLO model to complete the training process and the metrics for each method for comparison. TABLE III displays each algorithm's training time, precision, recall, and mAP_0.5. Fig. 8 and Fig. 9 show the cumulative graph of performance characteristics of precision, recall, mAP_0.5, and mAP_0.5:0.95 on the YOLOv5's model and Scaled-YOLOv4 model.

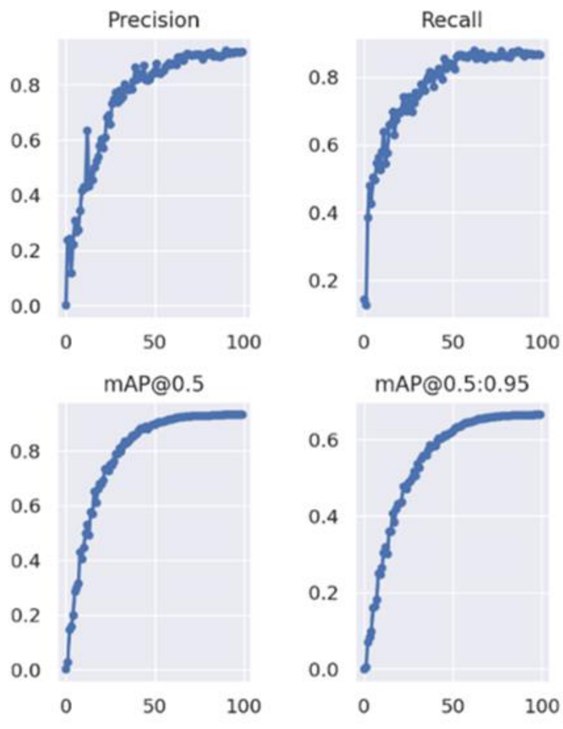


Fig. 8. Precision, Recall, mAP@0.5, mAP@0.5:0.95, of 100 epochs on YOLOv5's model

The performance of the training model evaluates from these attributes: precision, recall, and mAP. Precision is a means of determining how accurate the predictions are. It is the percentage of accurate predictions during the training process. Recall means how well it detects all the positives.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Where TP is True Positive, FP is False Positive and FN is False Negative

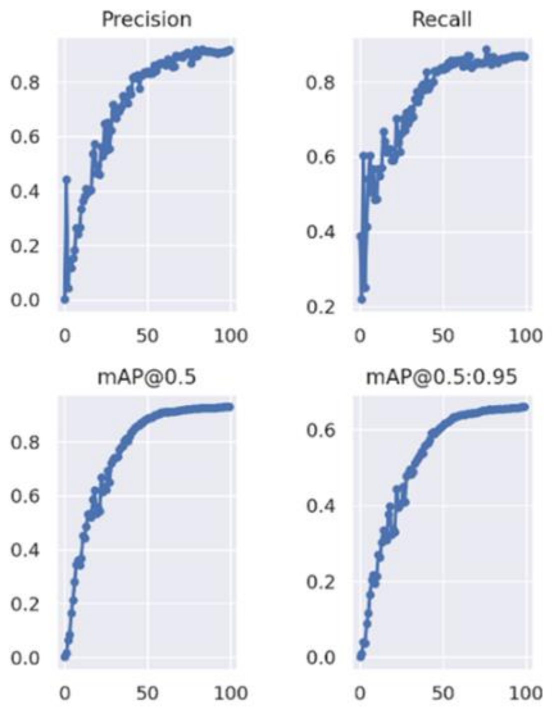


Fig. 9. Precision, Recall, mAP@0.5, mAP@0.5:0.95, of 100 epoch on Scaled-YOLOv4's model

The mean average precision, mAP, acts as an accuracy function. mAP computes a score by comparing the detected bounding box to the ground-truth bounding box. The greater the value, the more accurate the model's detections. The mAP@0.5 indicates that IoU is set to 0.5. The average percentage of all pictures of each category is calculated, and then all categories are averaged. IoU is an acronym that stands for interaction over the union. IoU will calculate the overlap of the two boundaries.

The Intersection over Union (IoU) calculates how much the estimated boundary overlaps with the actual boundary. mAP@0.5:0.95 is the average mAP for different IoU thresholds between 0.5 to 0.95 in the step of 0.05. Based on TABLE III, the YOLOv5 model performs better than Scaled-YOLOv4 in terms of accuracy, execution time and lightweight.

$$Intersection\ over\ Union, IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union}$$

Fig. 10, Fig. 11, Fig. 12, and Fig. 13 shows detailed comparison performance graph of precision, recall, mAP@0.5 and mAP @0.5:0.9 that being collected from trained process of YOLOv5 and Scaled-YOLOv4. From those graphs, it can be seen that YOLOv5 has a slightly better result but is much faster than Scaled-YOLOv4, as shown in Table III.

Table IV shows the performance achieved by other research works utilizing crop dataset for robot vision. As shown in the table, the proposed approach has achieved a slightly better mAP than the other approaches. Although the table is not a fair benchmarking due to different dataset and hardware used, the result shows that the proposed approach is promising.

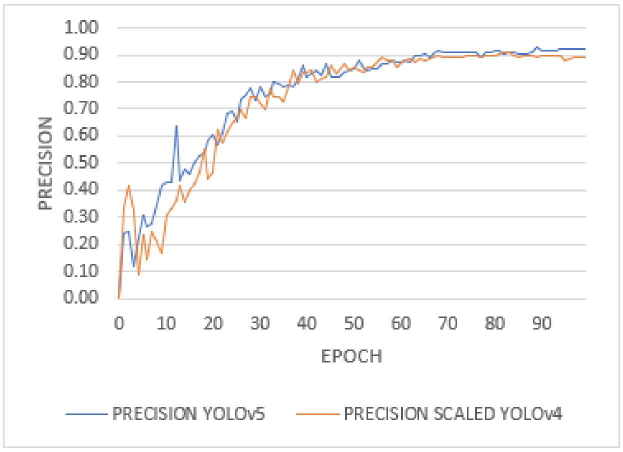


Fig. 10. Comparison graph (Precision) for YOLOv5 and Scaled YOLOv4

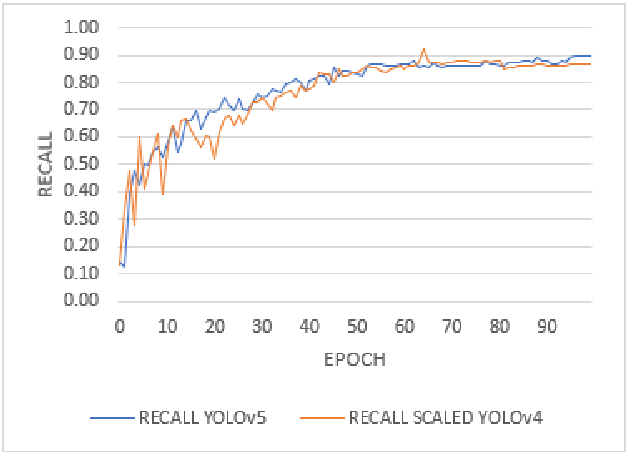


Fig. 11. Comparison graph (Recall) for YOLOv5 and Scaled YOLOv4

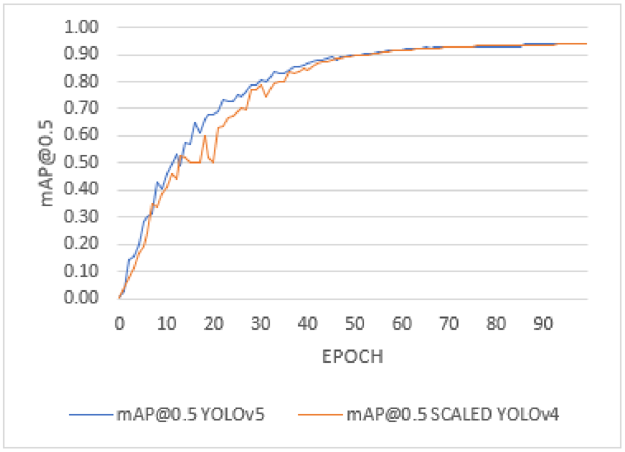


Fig. 12. Comparison graph (mAP@0.5) for YOLOv5 and Scaled YOLOv4

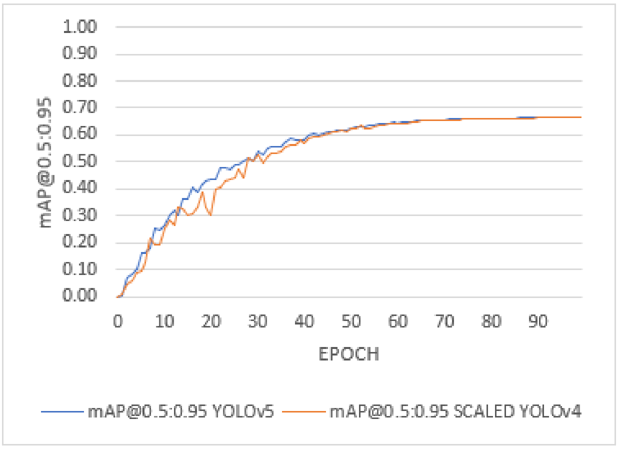


Fig. 13. Comparison graph (mAP@0.5:0.95) for YOLOv5 and Scaled YOLOv4

TABLE III PERFORMANCE OF TRAINING TIME, PRECISION, RECALL, AND MAP_0.5 ON YOLOV5 AND SCALED-YOLOV4 MODEL

YOLO Version	Training Time	Precision	Recall	mAP_0.5
YOLOv5s	2 hours 30 minutes	92.0%	89.6%	94.2%
Scaled-YOLOv4	3 hours 50 minutes	88.9%	86.6%	94.0%

TABLE IV PERFORMANCE OF OTHER RESEARCH WORKS UTILIZING CROP DATASET FOR ROBOT VISION

Reference	Approach	mean Average Precision (mAP)
Roy et al. (2021) [28]	YOLOv4	86%
Wu et al. (2021) [29]	YOLOv4	90%
Thuan et al. (2021) [30]	YOLOv5	93%
Abed et al. (2021) [31]	U-Net	91.02%
Proposed apprach	**YOLOv5**	**94%**

3) *Video Testing*

The video testing was performed to evaluate the detection accuracy of these two YOLO models. In Fig. 14 and Fig. 15, the video's time has been paused to show the comparison between these two models. The YOLOv5 model managed to localize each common potato scab that it saw in the YOLOv5's model by a bounding box, while the Scaled-YOLOv4 model could not detect all of the common potato scabs. Therefore, YOLOv5 detects more accurately and faster than Scaled-YOLOv4, meaning that YOLOv5 is more suitable for real-time object detection.

4) *Test Images*

Fig. 16 shows some of the test images detected on healthy and diseases class of solanaceous crops performed by the YOLOv5 model used in this research.

5) *Results Discussion*

From the results shown previously, the trained model has achieved a detection accuracy of around 94.2%. However, some bounding boxes are too big for the disease area. The full name labeled and prediction does not appear as a whole in the image. This is due to the name being set too long, making it not fully appear in the images. Therefore, the annotating and labeling must be done correctly using a shorter but meaningful name. The bounding box should be drawn close to the object's area that requires detection. This action can help the training algorithm learn only at the bounding box area.

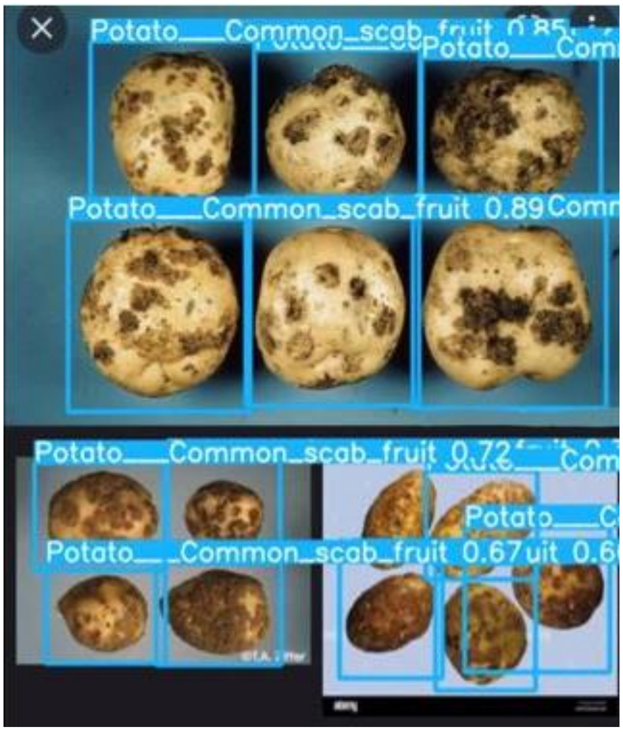

Potato___Common_scab_fruit 0.85
Potato___Common_scab_fruit 0.89
Potato___Common_scab_fruit 0.72
Potato___Common_scab_fruit 0.67

Fig. 14. Part of the testing video of the YOLOv5s model

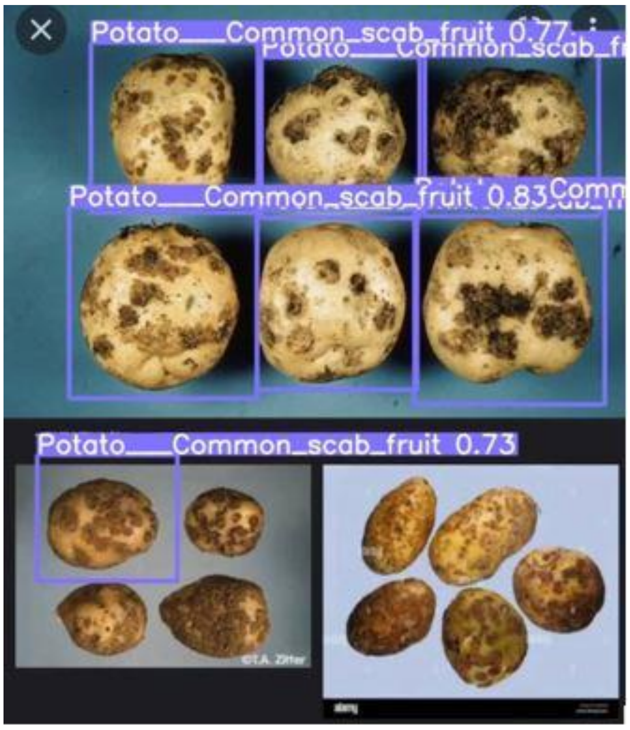

Potato___Common_scab_fruit 0.77
Potato___Common_scab_fruit 0.83
Potato___Common_scab_fruit 0.73

Fig. 15. Part of the testing video of the Scaled-YOLOv4 model

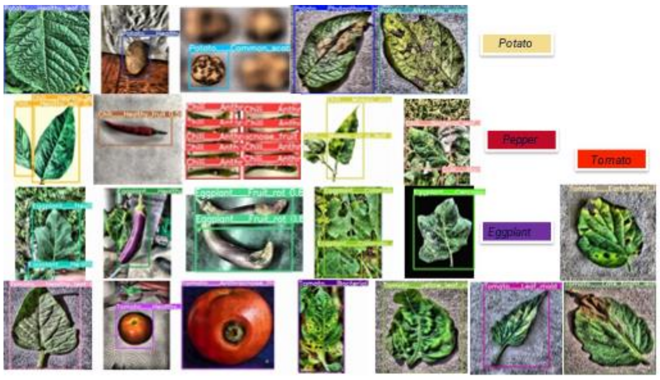


Fig. 16. Some of the detection images of diseases and healthy solanaceous crops

V. CONCLUSION

This study presents the process details and suggestions of the problems that arose during the development of crop disease detection for robot vision. An efficient object detector is required to ensure that the robot's vision mimics the ability of human sight. For that purpose, the Scaled-YOLOv4 and YOLOv5 were tested in this study. The simulation work was carried out using a Google Colab notebook through the Pytorch framework. The Roboflow.ai website aids in creating the custom dataset by providing annotating, labeling, preprocessing, and data augmentation functions. It can also help in exporting a particular file format into a format required for the training process. Performance has been evaluated from training the dataset of 16580 images with 100 epochs and 16 batch sizes and shows that the mean average precision using the YOLOv5 model is 94.2% which is better than Scaled-YOLOv4. YOLOv5 also has shown better performance in training time and video testing. The outcomes demonstrated the potential of YOLOv5 as an important robot vision.

In the future, the designed model can be deployed on a real-world device by converting the trained weights of the model's network into an embedded device, such as a mobile phone. After deployment, this model can assist modern farmers with automatic crop disease detection at any time and place. Future work should concentrate on detecting diseases in various crop parts, tracking disease progression, and suggesting information to prevent the diseases.

ACKNOWLEDGEMENT

The authors would like to thank Universiti Teknikal Malaysia Melaka (UTeM), Machine Learning and Signal Processing (MLSP) research group under the Centre for Telecommunication Research & Innovation (CeTRI) and PJP/2022/FKEKK/S01866 grant for sponsoring this work and the use of the existing facilities to complete this project.

REFERENCES

[1] N. Moitra, A. Singh, and S. Das, “Use of Convolutional Neural Network (CNN) to Detect Plant Disease,” In *Computational Advancement in Communication, Circuits and Systems*, pp. 43-51, 2022.

[2] M. B. Isman, "Challenges of Pest Management in the Twenty First Century: New Tools and Strategies to Combat Old and New Foes Alike," Frontiers in Agronomy, vol. 1, p. 2, Dec. 2019, doi: 10.3389/FAGRO.2019.00002.

[3] M. Tahmasebi, M. Gohari, and A. Emami, "An Autonomous Pesticide Sprayer Robot with a Color-based Vision System," *International Journal of Robotics and Control Systems*, vol. 2, no. 1, pp. 115–123, Feb. 2022, doi: 10.31763/ijrcs.v2i1.480.

[4] M. G. Mohanan and A. Salgaonkar, "Robotic Motion Planning in Dynamic Environments and its Applications," *International Journal of Robotics and Control Systems*, vol. 2, no. 4, pp. 666–691, Oct. 2022, doi: 10.31763/IJRCS.V2I4.816.

[5] A. K. Ali and M. M. Mahmoud, "Methodologies and Applications of Artificial Intelligence in Systems Engineering," *International Journal of Robotics and Control Systems*, vol. 2, no. 1, pp. 201–229, Mar. 2022, doi: 10.31763/IJRCS.V2I1.532.

[6] A. W. L. Yao and H. C. Chen, "An Intelligent Color Image Recognition and Mobile Control System for Robotic Arm," *International Journal of Robotics and Control Systems*, vol. 2, no. 1, pp. 97–104, Feb. 2022, doi: 10.31763/ijrcs.v2i1.557.

[7] A. Boubakri and S. Mettali Gamar, "A New Architecture of Autonomous Vehicles: Redundant Architecture to Improve Operational Safety," *International Journal of Robotics and Control Systems*, vol. 1, no. 3, pp. 355–368, Sep. 2021, doi: 10.31763/ijrcs.v1i3.437.

[8] T. Kamyab, A. Delrish, H. Daealhaq, A. M. Ghahfarokhi, and F. Beheshtinejad, "Comparison and Review of Face Recognition Methods Based on Gabor and Boosting Algorithms," *International Journal of Robotics and Control Systems*, vol. 2, no. 4, pp. 610–617, Sep. 2022.

[9] L. Alzubaidi et al., "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," Journal of Big Data, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/S40537-021-00444-8.

[10] A. Esteva et al., "Deep learning-enabled medical computer vision," npj Digital Medicine, vol. 4, no. 1, Dec. 2021, doi: 10.1038/S41746-020-00376-2.

[11] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," SN Computer Science, vol. 2, no. 6, pp. 1–20, Nov. 2021, doi: 10.1007/S42979-021-00815-1.

[12] B. Yang and Y. Xu, "Applications of deep-learning approaches in horticultural research: a review," Horticulture Research, vol. 8, no. 1, pp. 1–31, Jun. 2021, doi: 10.1038/s41438-021-00560-9.

[13] J. Deng, X. Xuan, W. Wang, Z. Li, H. Yao, and Z. Wang, "A review of research on object detection based on deep learning," in *Journal of Physics: Conference Series*, vol. 1684, no. 1, Nov. 2020, doi: 10.1088/1742-6596/1684/1/012028.

[14] S. P. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," Frontiers in Plant Science, vol. 7, no. September, p. 1419, Sep. 2016, doi: 10.3389/fpls.2016.01419.

[15] L. Alzubaidi et al., "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," Journal of Big Data, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/S40537-021-00444-8.

[16] P. Bharman, S. Ahmad Saad, S. Khan, I. Jahan, M. Ray, and M. Biswas, "Deep Learning in Agriculture: A Review," *Asian Journal of Research in Computer Science*, pp. 28–47, Feb. 2022, doi: 10.9734/ajrcos/2022/v13i230311.

[17] S. J. Pan and Q. Yang, "A survey on transfer learning," IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345–1359, 2010, doi: 10.1109/TKDE.2009.191.

[18] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," SN Computer Science 2021 2:6, vol. 2, no. 6, pp. 1–20, Aug. 2021, doi: 10.1007/S42979-021-00815-1.

[19] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach," Procedia Computer Science, vol. 132, pp. 679–688, Jan. 2018, doi: 10.1016/J.PROCS.2018.05.069.

[20] "A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way | by Sumit Saha | Towards Data Science." https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 (accessed May 27, 2022).

[21] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," Insights into Imaging, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/S13244-018-0639-9.

[22] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object Detection with Deep Learning: A Review," IEEE Transactions on Neural Networks and Learning Systems, vol. 30, no. 11, pp. 3212–3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.

[23] J. Kaur and W. Singh, "Tools, techniques, datasets and application areas for object detection in an image: a review," Multimedia Tools and Applications, pp. 1–55, Apr. 2022, doi: 10.1007/S11042-022-13153-Y.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2016-December, pp. 779–788, Jun. 2015, doi: 10.48550/arxiv.1506.02640.

[25] Karimi Grace, "Introduction to YOLO Algorithm for Object Detection | Engineering Education (EngEd) Program | Section," Apr. 15, 2022. https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/ (accessed Aug. 10, 2022).

[26] Nelson Joseph, "Your Comprehensive Guide to the YOLO Family of Models," Jun. 07, 2021. https://blog.roboflow.com/guide-to-yolo-models/ (accessed May 28, 2022).

[27] S. N, A. M. P, and H. P. V, "Object Detection using YOLO And Mobilenet SSD: A Comparative Study," International Journal of Engineering Research & Technology, vol. 11, no. 6, Jun. 2022, doi: 10.17577/IJERTV11IS060065.

[28] A. M. Roy and J. Bhaduri, "A Deep Learning Enabled Multi-Class Plant Disease Detection Model Based on Computer Vision," AI, vol. 2, no. 3, pp. 413–428, Aug. 2021, doi: 10.3390/ai2030026.

[29] L. Wu, J. Ma, Y. Zhao, and H. Liu, "Apple detection in complex scene using the improved yolov4 model," Agronomy, vol. 11, no. 3, Mar. 2021, doi: 10.3390/agronomy11030476.

[30] T. Do, "Evolution of yolo algorithm and yolov5: the state-of-the-art object detection algorithm," 2021.

[31] S. H. Abed, A. S. Al-Waisy, H. J. Mohammed, and S. Al-Fahdawi, "A modern deep learning framework in robot vision for automated bean leaves diseases detection," Int. J. Intell. Robot. Appl., vol. 5, no. 2, pp. 235–251, 2021, doi: 10.1007/s41315-021-00174-3.

[32] J. Schmidhuber, "Deep learning in neural networks: An overview," Neural Networks, vol. 61, pp. 85–117, Jan. 2015, doi: 10.1016/J.NEUNET.2014.09.003.

[33] K. Santosh, N. Das, and S. Ghosh, "Deep learning models," Deep Learning Models for Medical Imaging, pp. 65–97, 2022, doi: 10.1016/B978-0-12-823504-1.00013-1.

[34] M. Sewak, S. K. Sahay, and H. Rathore, "An overview of deep learning architecture of deep neural networks and autoencoders," *J Comput Theor Nanosci*, vol. 17, no. 1, pp. 182–188, 2020, doi: 10.1166/jctn.2020.8648.

[35] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," Insights into Imaging, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/S13244-018-0639-9.

[36] Md. R. Karim, Mohit. Sewak, and Pradeep. Pujari, "Practical Convolutional Neural Networks : Implement advanced deep learning models using Python," p. 211, Accessed: Oct. 25, 2022. [Online]. Available: https://www.perlego.com/book/593216/practical-convolutional-neural-networks-pdf

[37] V. Kate and P. Shukla, "A 3 Tier CNN model with deep discriminative feature extraction for discovering malignant growth in multi-scale histopathology images," Informatics in Medicine Unlocked, vol. 24, p. 100616, Jan. 2021, doi: 10.1016/J.IMU.2021.100616.

[38] J. Wen et al., "Convolutional neural networks for classification of Alzheimer's disease: Overview and reproducible evaluation," Medical Image Analysis, vol. 63, Jul. 2020, doi: 10.1016/J.MEDIA.2020.101694.

[39] H. Liang, X. Sun, Y. Sun, and Y. Gao, "Text feature extraction based on deep learning: a review," Eurasip Journal on Wireless Communications and Networking, vol. 2017, no. 1, Dec. 2017, doi: 10.1186/S13638-017-0993-1.

[40] M. S. H, "Plant Disease Detection and Classification using CNN", doi: 10.35940/ijrte.C6458.0910321.

[41] Y. Xiao et al., "A review of object detection based on deep learning," Multimedia Tools and Applications, vol. 79, no. 33–34, pp. 23729–23791, Sep. 2020, doi: 10.1007/S11042-020-08976-6.

[42] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object Detection in 20 Years: A Survey," May 2019, [Online]. Available: http://arxiv.org/abs/1905.05055.

[43] "Object Detection Guide | Fritz AI." https://www.fritz.ai/object-detection/ (accessed Jun. 13, 2022).

[44] D. Paper, "Object Detection," State-of-the-Art Deep Learning Models in TensorFlow, pp. 321–339, 2021, doi: 10.1007/978-1-4842-7341-8_13.

[45] J. Kaur and W. Singh, "Tools, techniques, datasets and application areas for object detection in an image: a review," *Multimedia Tools and Applications*, vol. 81, no. 27, pp. 38297–38351, Apr. 2022, doi: 10.1007/S11042-022-13153-Y.

[46] A. C. G, K. Krishnan, and K. S. Angel Viji Associate Professor, "Multiple Object Tracking using Deep Learning with YOLO V5," [Online]. Available: www.ijert.org.

[47] X. Wu, D. Sahoo, and S. C. H. Hoi, "Recent advances in deep learning for object detection," *Neurocomputing*, vol. 396, pp. 39–64, Jul. 2020, doi: 10.1016/J.NEUCOM.2020.01.085.

[48] J. Deng, X. Xuan, W. Wang, Z. Li, H. Yao, and Z. Wang, "A review of research on object detection based on deep learning," in *Journal of Physics: Conference Series*, Nov. 2020, vol. 1684, no. 1. doi: 10.1088/1742-6596/1684/1/012028.

[49] K. Zaatouri and T. Ezzedine, "A Self-Adaptive Traffic Light Control System Based on YOLO," 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), 2018, pp. 16-19, doi: 10.1109/IINTEC.2018.8695293.

[50] A. J. Abougarair, M. K. I. Aburakhis, and M. M. Edardar, "Adaptive Neural Networks Based Robust Output Feedback Controllers for Nonlinear Systems," *International Journal of Robotics and Control Systems*, vol. 2, no. 1, pp. 37–56, Jan. 2022, doi: 10.31763/ijrcs.v2i1.523.

[51] N. Wang *et al.*, "An Enhanced YOLOv5 Model for Greenhouse Cucumber Fruit Recognition Based on Color Space Features," *Agriculture*, vol. 12, no. 10, p. 1556, Sep. 2022, doi: 10.3390/AGRICULTURE12101556.

[52] M. Sozzi, S. Cantalamessa, A. Cogato, A. Kayad, and F. Marinello, "Automatic Bunch Detection in White Grape Varieties Using YOLOv3, YOLOv4, and YOLOv5 Deep Learning Algorithms," *Agronomy*, vol. 12, no. 2, p. 319, Jan. 2022, doi: 10.3390/AGRONOMY12020319.

[53] Z. Chen *et al.*, "Plant Disease Recognition Model Based on Improved YOLOv5," *Agronomy*, vol. 12, no. 2, p. 365, Jan. 2022, doi: 10.3390/AGRONOMY12020365.

[54] L. Zhijun, Y. Shenghui, S. Deshuai, L. Xingxing, and Z. Yongjun, "Yield Estimation Method of Apple Tree Based on Improved Lightweight YOLOv5," *Smart Agriculture*, vol. 3, no. 2, p. 100, Jun. 2021, doi: 10.12133/J.SMARTAG.2021.3.2.202105-SA005.

[55] L. Wang, Y. Zhao, Z. Xiong, S. Wang, Y. Li, and Y. Lan, "Fast and precise detection of litchi fruits for yield estimation based on the improved YOLOv5 model," *Front Plant Sci*, vol. 13, Aug. 2022, doi: 10.3389/FPLS.2022.965425.

[56] "Roboflow and Ultralytics, Creator of YOLOv5, Partner to Streamline Custom Computer Vision Model Development - EnterpriseTalk." https://enterprisetalk.com/news/roboflow-and-ultralytics-creator-of-yolov5-partner-to-streamline-custom-computer-vision-model-development/ (accessed Aug. 10, 2022).

[57] J. Ieamsaard, S. N. Charoensook, and S. Yammen, "Deep Learning-based Face Mask Detection Using YoloV5," in Proceeding of the 2021 9th International Electrical Engineering Congress, iEECON 2021, Mar. 2021, pp. 428–431, doi: 10.1109/iEECON51072.2021.9440346.

[58] Abdullah, S. Ali, Z. Khan, A. Hussain, A. Athar, and H. C. Kim, "Computer Vision Based Deep Learning Approach for the Detection and Classification of Algae Species Using Microscopic Images," *Water*, vol. 14, no. 14, p. 2219, Jul. 2022, doi: 10.3390/W14142219.

[59] V. Maslej-Krešňáková et al., "Automatic Detection of Atmospherics and Tweek Atmospherics in Radio Spectrograms Based on a Deep Learning Approach," Earth and Space Science, vol. 8, no. 11, Nov. 2021, doi: 10.1029/2021EA002007.

[60] H. Y. Chen et al., "Glove Defect Detection Via YOLO V5," Mekatronika, vol. 3, no. 2, pp. 25–30, 2020.

[61] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 13024–13033, 2021, doi: 10.1109/CVPR46437.2021.01283.