A Comparative Study on Plant Diseases Using Object Detection Models

Abu Adnan Sadi[1], Ziaul Hossain[2], Ashfaq Uddin Ahmed[1], and Md. Tazin Morshed Shad[1]

[1] North South University, Dhaka, Bangladesh
[2] University of the Fraser Valley, British Columbia, Canada
abu.sadi05@northsouth.edu, ziaul.hossain@ufv.ca, ashfaq.ahmed@northsouth.edu, tazin.shad@northsouth.edu

Abstract. Plant diseases are the most widespread and significant hazard to 'Precision Agriculture'. With early detection and analysis of diseases, the successful yield of cultivation can be increased; therefore, this process is regarded as a critical event. Unfortunately, manual observation-based detection method is error-prone, hard, and costly. Automation in identifying plant diseases is extremely beneficial because it saves time and manpower. Applying a neural network-based solution can detect disease symptoms at an early stage and facilitate the process of taking preventive or reactive measures. There have been various deep learning-based solutions, which were developed using lengthy training/testing cycles with large datasets. This study aims to investigate the suitability of computer vision-based approaches for this purpose. A comparative study has been performed using recently proposed object detection models such as YOLOv5, YOLOX, Scaled Yolov4, and SSD. A tailored version of the "PlantVillage" and "PlantDoc" datasets was used in the Indian sub-continent context, which included plant disease classes related to Potato, Corn, and Tomato plants. This study provides a detailed comparison between these object detection models and summarizes the suitability of these models for different cases. This paper can be useful for prospective researchers to decide which object detection models could be used for a specific scenario of Plant Disease Detection.

Keywords: Plant diseases, Machine Learning, Object Detection, YOLO, SSD, Comparative Study.

1 Introduction

Modern innovations have enabled humanity to produce food that can fill out the needs of the world population, but still almost 800 million people are undernourished. [1]. However, the food production and supply chain can still be jeopardized by a variety of factors such as climate change, plant illnesses, displacement of pollinators, and others. Besides causing a threat to food production, the consequences of diseased plants on small-scale farmers are a financial disaster. Small-scale farmers produce more than 80% of the countryside generation in developing countries, and stories of abdicating misfortune of more than 50% due to troubles and diseases are typical [2]. The unsuccessful low crop yield due to diseases further worsens the nourishment level of small-scale farmer families, which are typically already suffering from this problem. Cultivation is one of the most valuable callings in countries such as Bangladesh, India, and Pakistan (Indian subcontinent). Most farmers in these countries, however, face financial crises throughout the whole term. Diseases in plants pose a significant risk to

crop production and food quality. They are one of the leading sources of crop failure and financial loss in agribusiness [3]. Traditional agribusiness is focused on water management, agricultural development, and marketing. If the preceding stages are maximized, it may ease the financial progress of ranchers. Many crops are grown in these countries, and agriculturists frequently don't know whether the picked trim is diseased or not. The observable proof of plant diseases is critical for maximizing the crop yield and calculating wages.

Numerous researchers have undertaken several efforts to control the adverse effects of plant diseases. To prevent and control these, correctly recognizing the disease is the most important task, which can prove to be a difficult task that demands experience and skill. Ailments or their symptoms, such as colored dots or streaks, can usually be observed as the plant takes off. Microbes, including parasites, tiny creatures, and diseases, are frequently responsible for the plant's ailments [4]. Because of the source or etiology of the plant ailment, there is a wide range of symptoms and side effects. In recent years, traditional machine-learning techniques have become popular to determine disease location. Deep learning, a more concise approach of machine learning, refers to the use of simulated neural network models with many training layers. As examples of conclusion-to-conclusion learning, neural systems have been gaining popularity in a variety of fields [5]. In a neural network, hubs are a numerical capacity that takes numerical inputs from the approaching edges and outputs a numerical yield as an active edge. The Convolutional Neural Network (CNN) may hold its uses in the agricultural field, counting identifiable proof of diseases as well as evaluating the affected zone. In farming, most of the infections are identified with just an observer's (farmer) eye perception. This technique entails investing a lot of time in enormous farms and is very exhausting. The use of convolutional neural nets in the early recognition and detection of plant diseases will be helpful in increasing the quality of products [6]. The use of a neural network to identify plant disease is tremendously valuable to agriculture, primarily because it saves time and workforce. It saves time and effort in big agricultural fields by detecting illness symptoms early on, allowing farmers to take the required safeguards to avert harm.

Even though numerous research has been conducted on the introduction, management, and classification of plant disease detection, the comparison of these methods was not done in a conclusive manner. Moreover, the studies were conducted in the global context, while this study aims to focus on problems that are more common in the Indian sub-continent context. This study has presented the comparison of a few of the cutting-edge computer vision-based systems for early plant disease diagnosis which can be treated as a valuable resource for researchers who want to decide on which method to use in a particular scenario.

To construct such a precise picture classifier aimed at determining plant infections, a large, managed, and verified dataset containing various infected and non-infected plant images is needed. The "PlantVillage" [7] and the "PlantDoc" [8] datasets have gathered thousands of plant images and made them available for free usage. The "PlantVillage" dataset contains thousands of images of individual plant leaves that were taken in a lab environment (see Fig. 1). On the other hand, the "PlantDoc" dataset contains images of plants that are taken in a more natural environment (see Fig. 2). In a more natural environment, it is expected that a single image will contain multiple target objects (diseased leaves), as opposed to only one target object per image that can be seen in the PlantVillage dataset. In order to improve the robustness of the models, images from both datasets were used to investigate the performance of the different object detection

approaches. Object detection models such as YOLOv5 [21], Scaled YOLOv4 [11], and SSD [12] have been used in a comparative manner. A customized version of the "PlantVillage" and "PlantDoc" datasets has been used, which included plant disease classes more relevant to the settings of Bangladesh, India, and Pakistan. In this experiment, an attempt was made to detect some common diseases for three types of crops – tomato, corn, and potato. These three types of crops a prevalent in the Indian sub-continent, hence making them suitable for this study. A subset of images was chosen from both datasets to create the new customized dataset.

The two primary contributions of this study are – 1) A customized dataset containing images from both lab and natural environments was used to train the models, improving models' capability to detect diseases in more diverse settings, and 2) A comparative study was performed on plants and diseases that are prevalently found in the Indian sub-continent. The aim is to provide future researchers of this region with important information on which object detection model they could use for their own research purposes.

Section 2 presents some object detection-based models, their background, their evolution from the deep learning studies, and relevant comparison studies. Next, in Section 3, the experimental methodology is explained with a few details of the tested detection models. After that, the criteria on which the comparison was formulated, in other words, the evaluation metrics, is presented in Section 4. Sections 5 and 6 contain the experimental setup details, observed outcomes, and a discussion on the observation. Section 7 briefly discusses some of the limitations of this study. Finally, the findings and contents of the paper are concluded in Section 8.

2 Related Works

2.1 Deep Learning and Plant Diseases

Plant disease detection has advanced fast, thanks to the advances in image processing and deep learning. Plant disease detection and diagnostic technologies currently focus on plant disease classification, detection, and recognition of plant species. Many researchers have developed different techniques based on deep learning for classifying and detecting plant diseases.

In paper [13], Anjanadevi B et al. proposed an improved model for plant disease detection. Both the PlantVillage and PlantDoc datasets were used in this study. The outcomes of the experiments were compared to architectures such as Mobile Net, Dark Net-19, and ResNet-101. This proposed approach is more precise in terms of both localization and categorization. In their paper [14], Sk Mahmudul Hassan et al. used different deep convolutional neural network (CNN) models for the classification of plant diseases. The performance of these models was assessed using different parameters like batch size, epochs, etc. Models such as InceptionV3, InceptionResNetV2, MobileNetV2, and EfficientNetB0, respectively yielded the accuracy rates of 98.42%, 99.11%, 97.02%, and 99.56%. In paper [15], Shao Xiang et al. presented a lightweight convolutional neural network (CNN) for the identification of plant disease severity. This lightweight CNN model managed to achieve an accuracy rate of 90.6% on the Plant Disease Severity dataset and 97.9% on the PlantVillage dataset.

In their work [16], Kakade et al. suggested a new technique that uses neural network-based classification and image processing to identify and categorize illnesses in leaves. Their goal was to detect and adequately assess the early signs of diseases. They were able to detect and classify diseases with 92.94% accuracy. For the identification of diseases and pests in tomato plants, in paper [17], Fuentes et al. suggested an effective deep-learning-based technique. As deep learning meta-architectures, they utilized Faster R-CNN, R-FCN, and SSD and integrated all of them with VGG net and ResNet used for feature extraction. Their technology was able to distinguish between nine distinct illnesses and pests. In their study [18], Sharada P. Mohanty et al. used the public PlantVillage dataset for classification. The proposed deep CNN managed to yield up to a correctness of 99.35% on the test set.

2.2 YOLO (You Only Look Once)

In 2015, Joseph Redmon et al. introduced YOLO [19], a novel object detection methodology. Objects in the image were previously localized using areas in previous detection techniques. On the other hand, YOLO examines the whole picture and finds and locates the object. YOLO can recognize numerous classes in a photo at the same time. YOLO is a model that is exceptionally quick and useful for object detection. Many other versions of YOLO have been released throughout the years since its first release. Some mentionable updates are YOLOv3 [20] by Joseph Redmond himself, and YOLOv5 [21] by a company Ultralytics and YOLOX [10].

Throughout the years, YOLO has been included in several different research works related to plant disease detection. For example, in their study [9], the authors proposed a version of YOLOv5 for detection. They applied different techniques to reduce the number of parameters and calculations. The loss function was also updated from 'Generalized Intersection over Union' to 'Efficient Intersection over Union'. In the end, they managed to achieve a 5.4% higher mAP score than the base YOLOv5 model. In paper [22], Achyut Morbekar et al. presented a system that uses the object detection model YOLOv3 for detecting plant diseases. Images from the PlantVillage dataset were used for this study. In paper [23], Vijayakumar Ponnusamy et al. presented a wearable device that uses the YOLOv3 model to detect damaged plant leaves in real-time.

In this study, two of the recent versions of YOLO are used. They are YOLOv5 and Scaled YOLOv4.

2.3 SSD (Single Shot Multi-Box Detector)

Wei Liu [12], together with others, presented SSD: Single Shot Multi-Box in 2016 - a method that detects objects in pictures employing a single deep neural network. In this method, a set of default boxes is discretized from the yield space of bounding boxes. The set ranges over diverse aspect proportions and scales per feature map area. During prediction, the presence of each object category in each default box is given scores and produces alterations to the box to superior coordinate the object shape. Furthermore, the arrangement combines predictions from different feature maps with diverse resolutions to handle objects of different sizes. SSD is straightforward relative to other strategies that require object recommendations in several terms. Firstly, it dispenses fully with the proposition era and highlights resampling stages after ensuing pixels. Also, it typifies all computation into a single organization. As a result, SSD becomes simpler to train and direct to coordinate into frameworks in requirement of a detection component.

2.4 Comparative Study on Object Detection Models

In their paper [24], Vu Thanh Nguyen et al. conducted a comparison of several objection models for detecting plant leaf disease. YOLOv3, RetinaNet, Faster RCNN, and Mask RCNN were the models employed in this research. The findings suggest that YOLO is the fastest approach for data processing but has a low mAP of 60.5%. Mask RCNN has the highest mAP of 82.9%, but it takes the longest to process. To balance the speed, time, and resources required, YOLO is the ideal option for developing applications. In [25], Min Li et al. conducted a study of comparison between Faster R-CNN, YOLOv3, and SSD models for the detection of Agricultural Greenhouses (AGs). Considering the mean average precision (mAP) and frames per second (FPS) metrics, YOLOv3 delivered the most significant results in terms of accuracy and efficiency. The SSD outperformed the Faster R-CNN in detection speed, with an FPS twice as high, despite the mAP being similar on the test set. The authors concluded that YOLOv3 was superior in both accuracy and computing efficiency.

In their work [26], Jeong-ah Kim et al. performed a study that compares the performance of Faster-RCNN, YOLOv4, and SSD on an automobile training dataset. The YOLOv4 model outperformed the other two techniques, reaching a 93% accuracy rate in detecting vehicle models. In [27], R Deepa et al. performed a comparative study on the object detection algorithms YOLO, SSD, and Faster RCNN for real- time tennis ball tracking. The SSD model outperformed the other two models in the research, according to the authors. While performing the job of tennis ball tracking, the SSD model exhibited less processing time, more accuracy, and efficiency. In [28], Mohamed Lamine Mekhalf et al. performed a comparison between the detection models named Detection Transformers (DETR), YOLOv5, and EfficientDet for the purpose of crop circle detection. In conclusion, the authors agreed that DETR has restricted performance while object detection was more accurate through YOLOv5.

2.5 Limitations of Earlier Studies

A lot of the studies mentioned above used images from a single dataset to train their models [14, 18, 22, 24]. While training on a single dataset such as the PlantVillage dataset might result in models with higher accuracies, the lack of diversity in the training data could cause the models to fail in real-world settings. All the images present in the PlantVillage dataset are close-up images of plant leaves taken in a lab environment. In order to address this issue, this study combines images from both the PlantVillage and PlantDoc datasets to create a custom dataset. The images from the PlantDoc dataset are taken in more diverse real-world environments. Additionally, this study performs experiments with plant diseases that are common occurrences in the Indian sub-continent, providing future researchers with valuable information regarding which detection model to utilize under similar circumstances.

3 Proposed Methodology

3.1 Datasets

Data on agricultural issues is difficult to come by, and real-time photos are a key concern. Deep learning in the field of agriculture has not progressed as quickly as in other fields due to a lack of available data. One of the main reasons for this is that there

aren't enough publicly available agricultural datasets. The most popular public dataset available is the 'PlantVillage' dataset. The 'PlantVillage' dataset contains over 54000 images of individual plant leaves (see Fig. 1). There are 14 different crop species with 26 different diseases present in the dataset.

Fig. 1. Images from PlantVillage Dataset

The dataset contains pictures of individual leaves that were taken in a controlled environment with a gray or black background. Even though the photos in the plant village collection are very useful for disease classification, they are not ideal for detection because each image only comprises a single leaf.

Another publicly available dataset is the 'PlantDoc' dataset. This dataset was created specifically for the purpose of detecting plant diseases (see Fig. 2). There are 2,598 photos in total in the dataset, which span 13 plant species and up to 17 disease groups. All the images in this dataset represent real-life scenarios.

For this study, it was decided to take images from both the PlantVillage and PlantDoc datasets. However, all the images and classes present in both datasets were not used. Instead, a total of 3154 photos from the disease categories that are relevant to this research were chosen.

The custom dataset contained images of 3 plant species: Corn, Potato, and Tomato. There was a total of 8 diseased leaf classes and 3 healthy leaf classes. The names of these 11 classes are given in Table. 1.

Table 1. Classes Present in the Custom Dataset

Corn Diseases	**Potato Diseases**	**Tomato Diseases**	
Common Rust	Early Blight	Bacterial Spot	Healthy
Healthy	Healthy	Early Blight	Mosaic Virus
Northern Leaf Blight	Late Blight	Late Blight	

Fig. 2. Images from PlantDoc Dataset

For training all the models, we divided our dataset into two sets, the training set, and the validation set. The training and validation set ratio was 80:20. So, there were a total of 2524 training images and 630 validation images.

3.2 Annotations

Annotation is a fundamental step in computer vision research that involves assigning labels or identifiers to objects represented in images in order to train and evaluate detection models. Images must be annotated carefully and correctly because the detection model's accuracy is dependent on it. One of the most popular methods of annotation is to draw rectangular bounding boxes around the objects of interest within the image, thereby informing the model about the exact location of an object inside the picture. However, alternative image annotation techniques like segmentation, landmarking, lines and splines, and so on are also available. In the context of this study, the bounding box annotation technique was used to manually annotate the images within the dataset. The task was performed using the open-source annotation tool called CVAT (Computer Vision Annotation Tool) [29].

3.3 YOLO (You Only Look Once)

YOLO is a one-stage detector and one of the most well-known and state-of-the-art object detectors to date. The initial version of YOLO was introduced by Redmond et al. in 2015[19], and it was regarded as a breakthrough in real-time object recognition. Object detection models like R-CNN, applied region proposal techniques to build potential bounding boxes in an image before running a classifier on them. The model would then modify the bounding boxes and eliminate duplicates after classification. Although these frameworks have had a lot of success, this multi-stage approach introduced complexity and computational inefficiencies. In contrast, YOLO simplifies the object detection task by approaching it as a regression problem and combining all essential components in a single framework. This simplified approach significantly accelerates inference, as it eliminates the need for multiple processing stages.

The YOLO algorithm resizes images to the same size and divides them into a S x S grid (see Fig. 3). When an object's center falls inside a grid cell, that grid cell is tasked with the detection of the object. Bounding boxes and the confidence ratings for those boxes are predicted in each grid cell. There are five predictions in each bounding box: x, y, w, h, and confidence. However, there might be several boxes for the same object. In this scenario, the bounding box with the highest confidence rating is selected by the model by applying Non-Maximum Suppression (NMS), which eliminates any duplicate bounding boxes.

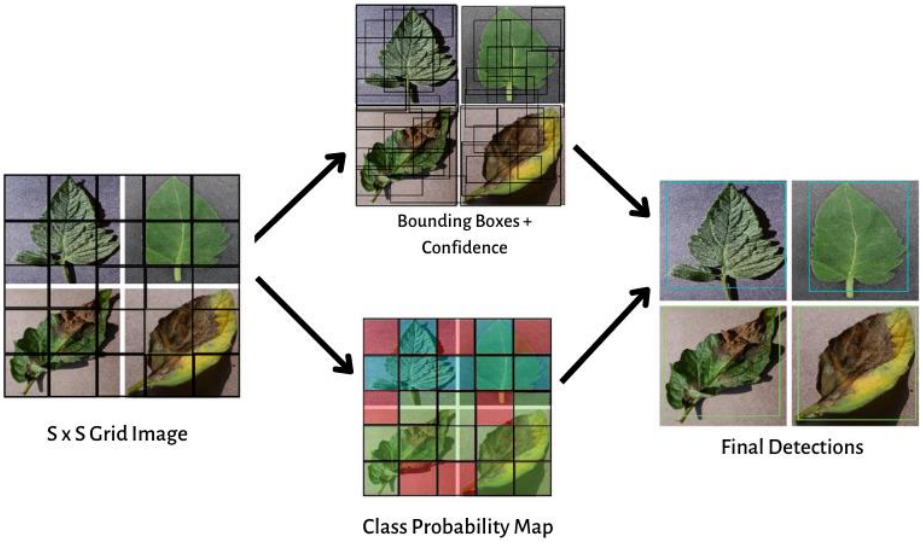

Fig. 3. The Object Detection Stages of YOLO

The confidence score of the bounding box is the IoU (Intersection Over Union) between the predicted box and any ground truth box. IoU is calculated by the equation shown in Fig. 4.

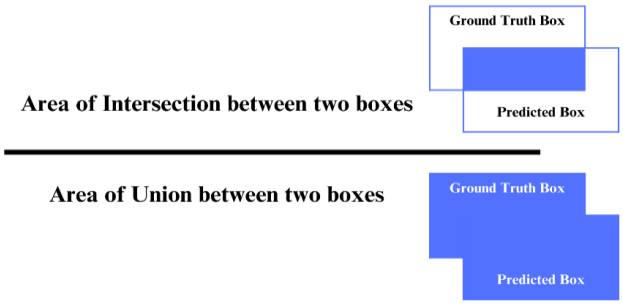

Fig. 4. IoU

The original version of YOLO experienced some generalization and localization problems when the objects in the image were too small, or the image had varied dimensions. The second version of YOLO tried to reduce these errors by introducing new features such as batch normalization and anchor boxes. For bounding box prediction, all fully connected layers are replaced by anchor boxes which makes the entire model a fully convolutional network. The increase in detection accuracy was one of the main goals of YOLOv3. A variation of Darknet-53 with 106 layers is used in

YOLOv3. Several techniques were also included, including skip connections, residual blocks, and up-sampling. These improved the performance of the model with small objects significantly. It does, however, come with a higher processing cost.

For YOLOv4, CSPDarknet53 serves as the backbone. CSP- Darknet53 also includes a spatial pyramid pooling algorithm to increase the receptive field and differentiate contextual information. Another notable aspect of YOLOv4 is its ability to run training on a single GPU. YOLOv5 is one of the most recent versions of YOLO. CSPNet is used as a backbone for this version. YOLOv5's key features include its user-friendliness, speed, performance, and ease of use. It has five different variants nano, small, medium, large, and Xlarge.

For this study, three different YOLO models will be utilized. These models include:

YOLOv5s. YOLOv5 offers multiple different variations of pre-trained models. YOLOv5s is one of the smallest and fastest models in the YOLOv5 family. Since the custom dataset used in this study is relatively small, the training process was initiated using pre-trained weights. These pre-trained weights are provided with the YOLOv5 model.

YOLOv5x. YOLOv5x is the largest variant in the YOLOv5 family. This model is supposed to produce better results in nearly all cases. But the tradeoff of this model is it takes longer to train and requires a lot of CUDA memory to train. Here, pre-trained weights were also used to initiate the training process.

Scaled YOLOv4. Scaled YOLOv4 was proposed by Chien-Yao Wang and others in 2020[11]. This model improves the YOLOv4 model by scaling the network's design and scale with efficiency. This model presents a network scaling strategy that alters not just the network's depth (number of convolutional layers in a CNN), width (number of convolutional filters in a convolutional layer), and resolution, but also its structure. For this study, the Scaled YOLOv4 model was used which scales the YOLOv4-large model. Training the Scaled YOLOv4 model is very resource-expensive as it scales the large YOLOv4 model.

3.4 SSD

As it moves through the neural network, the YOLO feature map's final stage has only brief information, which could restrict its accuracy. To address this shortcoming in YOLO, SSD takes a different approach by utilizing the convolutional feature map at the beginning, resulting in more detailed highlights. Furthermore, using the Faster R-CNN anchor concept, different sizes of objects may be detected. In each feature map, the SSD computation generates a default box with a specific proportion and scale. It constructs the final bounding box by applying the model's predicted coordinates and class values to the default box. Various feature maps can be used to generate predictions of differing sizes.

The SSD architecture employs a complex neural network. These neural networks rely on loss functions to learn effectively. In this case, there are two types of loss functions at play: confidence loss and location loss. Confidence loss assesses the algorithm's confidence in identifying whether an image's bounding boxes contain certain objects or classes. Location loss quantifies the difference between the network's predicted bounding boxes and the actual ground truth values used for training the computer. An

alpha term is used to balance the influence of location loss. The final loss is represented by the following formula-

$$\text{MultiBox Loss} = \text{confidence loss} + \text{alpha} * \text{location loss}$$

4 Evaluation Metrics

4.1 Precision and Recall

True Positive (TP) refers to a collection of positive attributes that are correctly recognized as positive attributes. In contrast, True Negative (TN) refers to a collection of negative attributes that are accurately recognized as negative. False Positives (FP) are negative attributes but are predicted as positive. Similarly, False Negatives (FN) are positive attributes but are predicted as negative.

Precision represents the proportion of correct positive predictions among all positive predictions. The True Positive (TP) and False Positive (FP) values are used to calculate precision (see Fig. 5).

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

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

Fig. 5. Precision and Recall

For example, when the model has a precision score of 0.947 for “Potato Late Blight,” it means that when the model predicts a disease as “Potato Late Blight,” it is correct 94% of the time. On the other hand, recall evaluates what proportion of actual positives are predicted correctly by the model. The True Positive (TP) and the False Negative (FN) values are used to calculate recall (see Fig. 5). For example, when the model has a recall score of 0.90 for “Potato Late Blight,” it means that when the model predicts a disease as “Potato Late Blight”, it correctly predicts 90% of all the "Potato Late Blight" samples present in the dataset.

4.2 Mean Average Precision (mAP)

The metric "Mean Average Precision" (mAP) is frequently used for evaluating how effectively object detection algorithms perform. A higher mAP value indicates better performance of the model. The mAP score is calculated by taking the mean of the Average Precision (AP) values across all classes present in the dataset. AP is calculated for each class by computing the precision-recall curve and then calculating the area under the curve (AUC). For this study, the precision-recall values were derived using the IoU (see Fig. 4) threshold value of 0.5.

4.3 Experimental Setup

The test setup included an Intel Center i7 7700K processor, 16 GB of DDR4 memory, an 8GB Nvidia GTX 1070 graphics card, and a Windows 10 operating system. The

Pytorch framework was utilized for training the object detection model. Additionally, the CVAT [29] annotation tool was utilized for performing data annotations.

5 Results

5.1 YOLO

In order to train the YOLOv5 model, transfer learning was employed. The developers of YOLOv5 recommended the use of pre-trained weights, especially when dealing with a relatively small dataset. Consequently, pre-trained weights were utilized for both the YOLOv5s and YOLOv5x models. However, in the case of training the Scaled YOLOv4, pre-trained weights were not employed, as Scaled YOLOv4 does not provide pre-trained weights out of the box like YOLOv5. As a result, the Scaled YOLOv4 model was trained from scratch. Table. 2, contains the training parameters that were kept consistent during the training of the YOLO models.

Table 2. Training Parameters

Parameter	Value
Batch Size	8
Epochs	50
Image Size	256x256

YOLOv5s. On the YOLOv5s model, the best mAP (Mean Average Precision) score achieved on the custom dataset was 0.932. The training process was completed in approximately 29 minutes. Detailed mAP, precision, and recall scores are shown in Table. 3 and Fig. 6. Furthermore, the final mAP scores for each specific class are shown in Table. 6.

Table 3. YOLOv5-S Results

Epochs	Precision	Recall	Best mAP
0-9	0.788	0.791	0.835
10-19	0.896	0.813	0.911
20-29	0.899	0.870	0.927
30-39	0.893	0.860	0.929
40-49	0.937	0.857	0.932

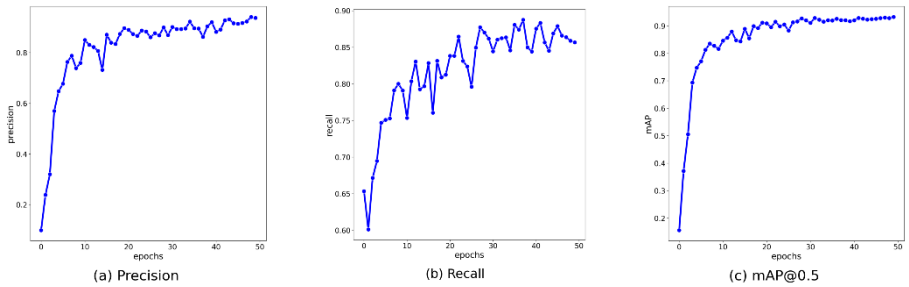

Fig. 6. Precision, Recall and mAP curves obtained from YOLOv5-S

YOLOv5x. On the YOLOv5x model, the best mAP (Mean Average Precision) score attained for the custom dataset was 0.941. The training process was completed in approximately 137 minutes. Detailed mAP, precision, and recall scores are shown in Table. 4 and Fig. 7. Additionally, Table. 6 showcases the final mAP scores for each specific class.

Table 4. YOLOv5-X Results

Epochs	Precision	Recall	Best mAP
0-9	0.867	0.845	0.896
10-19	0.898	0.851	0.919
20-29	0.915	0.869	0.931
30-39	0.901	0.894	0.937
40-49	0.936	0.889	0.941

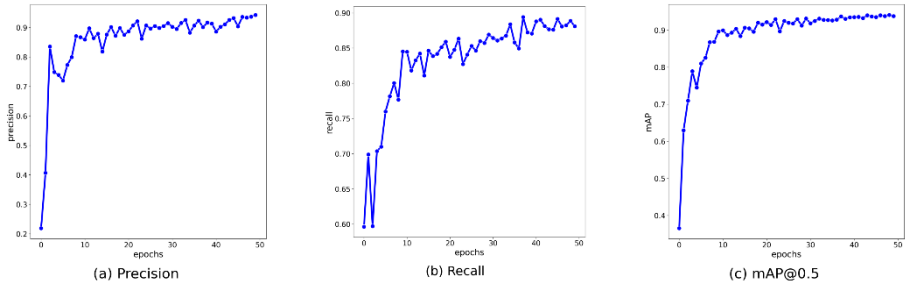

Fig. 7. Precision, Recall and mAP curves obtained from YOLOv5-X

Scaled YOLOv4. On the Scaled YOLOv4 model, the best mAP (Mean Average Precision) score of 0.643 was obtained after 50 epochs of training. The training process took approximately 87 minutes. Detailed mAP, precision, and recall scores are given in Table. 5 and Fig. 8. However, this model did not provide the final mAP scores for each class.

Table 5. Scaled YOLOv4 Results

Epochs	Precision	Recall	Best mAP
0-9	0.198	0.657	0.282
10-19	0.233	0.759	0.440
20-29	0.265	0.764	0.498
30-39	0.276	0.784	0.596
40-49	0.306	0.797	0.643

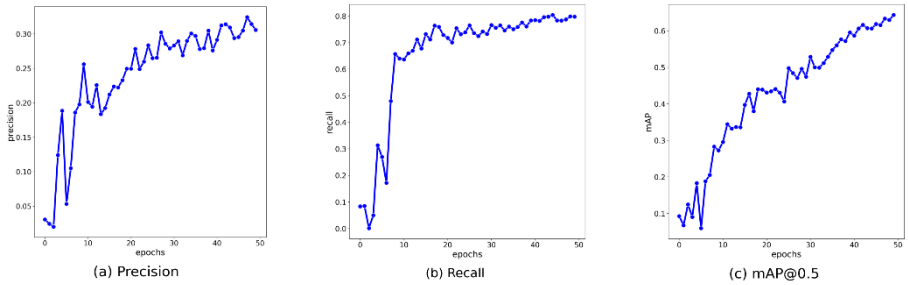

Fig. 8. Precision, Recall and mAP curves obtained from Scaled YOLOv4

5.2 SSD

The same training parameters as those provided in Table. 2 were used to train the SSD model. The best mAP (Mean Average Precision) score achieved during the training process was 0.905. The training process took approximately 160 minutes to complete. Detailed mAP scores are shown in Table. 6.

Table 6. Detailed mAP Scores of YOLOv5-S, YOLOv5-X, and SSD

Class	YOLOv5-S	YOLOv5-X	SSD
all	0.932	0.941	0.905
Corn Common Rust	0.970	0.958	0.899
Corn Healthy	0.971	0.975	0.909
Corn Northern Blight	0.887	0.877	0.902
Potato Early Blight	0.918	0.920	0.905
Potato Healthy	0.995	0.995	0.909
Potato Late Blight	0.912	0.941	0.904
Tomato Bacterial Spot	0.994	0.995	0.986
Tomato Early Blight	0.756	0.855	0.897
Tomato Healthy	0.930	0.935	0.906
Tomato Late Blight	0.899	0.905	0.897
Tomato Mosaic Virus	0.995	0.995	0.909

6 Result Analysis and Discussion

Out of the three models of YOLO, YOLOv5s had the fastest training time of approximately 29 minutes (see Table. 7). This was expected as it is the smallest model out of the three. On the other hand, the YOLOv5x was the largest model out of the three. Consequently, it required more CUDA memory and was slower to train. The YOLOv5x model took the longest time to train, taking about 137 minutes (see Table. 7). Scaled YOLOv4 also took a fairly long time to train, approximately 87 minutes (see Table. 7). Comparing the mAP scores, we found that both YOLOv5s and YOLOv5x had almost similar mAP scores. Even Though YOLOv5x is a very large model compared to YOLOv5s, it only produced a 0.01 higher mAP value than the small model, equivalent to an improvement of 1%. Multiple factors could have influenced such close results, such as the size of the dataset, the size of the images, and the training environment. Smaller models tend to perform more efficiently on smaller datasets than

larger models. The relatively small size of the custom dataset used in this study could have been one of the reasons why the small model performed so well.

In contrast to the YOLOv5 models, the Scaled YOLOv4 model yielded a relatively lower mAP score of 0.643. However, it can be observed from Table. 5 and Fig. 8 that the mAP score continued to show significant improvement even in the final ten epochs of training. This trend of improvement indicates that if the model had been trained for a longer amount of time by increasing the number of epochs, then the model could have achieved better results. A primary factor behind the slow training of the Scaled YOLOv4 model was the absence of pre-trained weights. This model does not provide readily available pre-trained weights like the YOLOv5 models. As a result, the model had to be trained from scratch, which extended the training time required for achieving the optimal mAP score.

Table 7. Comparison of models based on the mAP score and training time

Model	Best mAP	Training Time (in minutes)
YOLOv5-Small	0.932	29
YOLOv5-Xlarge	0.941	137
Scaled YOLOv4	0.643	87
SSD	0.905	160

On the other hand, the SSD model utilizes a VGG backbone, making it a large and strong model. Despite its strength, the SSD model achieved a lower mAP score than the other YOLO models. It also took the longest time to train, approximately 160 minutes (see Table. 7). After training for 50 epochs, the SSD model did not demonstrate any significant increase in the mAP score. The likely explanation for this is that the SSD model has some limitations when it comes to identifying small items effectively [30]. Given that plant the diseases present in plant leaves are usually smaller in nature, the model could have struggled to detect them accurately. Additionally, the SSD model had higher computational costs than the other three models, resulting in significant CPU, GPU, and RAM usage during the training phase.

Finally, some sample test images that weren't part of the training or validation sets were used to test the models. The trained models exhibited strong performance in accurately identifying the type of leaf diseases as well as the specific leaf type in most cases. However, in certain instances, the models did face some challenges in correctly identifying the type of leaf. For example, in Fig. 9, the image on the left is of the given label, and the image on the right is the prediction of the model. It can be seen that the model correctly identified the type of disease as "Early Blight." However, it mislabeled the leaf type as Potato leaf instead of Tomato leaf. Overall, aside from these minor flaws, the models did a great job at detecting plant illnesses. Fig. 10 contains images of some of the successful predictions of the YOLOv5-X model on test images. From the figure, it can be seen that the model was successful at detecting diseased leaves from the test images taken in both lab (PlantVillage) and natural (PlantDoc) environments. Therefore, the use of the custom dataset did help the model with generalization and improved its capability to detect diseases from images taken in diverse settings.

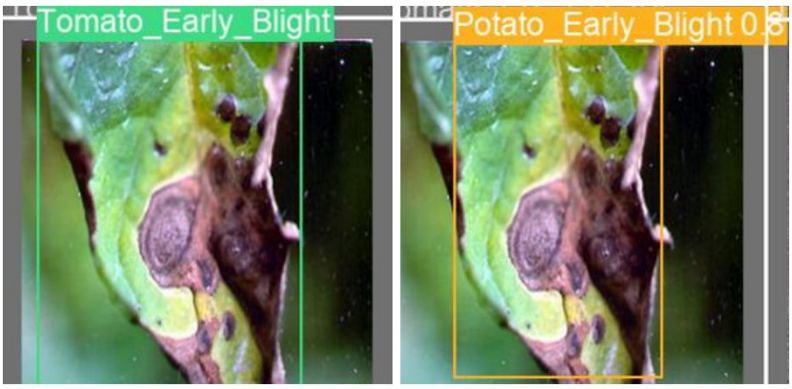

Fig. 9. Ground Truth (left) vs Predicted Result (Right)

7 Limitations

One of the main limitations of this research is the imbalance between the number of lab images (PlantVillage) and real-world images (PlantDoc) present in the custom dataset. The PlantDoc dataset has significantly less number of pictures when compared to the PlantVillage dataset. As a result, the custom dataset also contained a very high amount of pictures that were taken from the PlantVillage dataset. However, even with this imbalance present in the dataset, the models trained on the custom dataset were able to successfully detect diseased leaves on the test images that contained multiple target objects. Therefore, in the future, researchers need to focus on developing large plant disease datasets that contain high-quality images taken in real-world environments and that are more suitable for the task of object detection. Alternatively, sampling techniques such as undersampling and oversampling could be used to address the imbalance problem.

8 Conclusion

Early and accurate disease detection in plants is a pivotal task in the cultivation process to ensure a sustainable food supply chain. There are several research works in this domain. However, a comparison of these methods in the Indian sub-continent agriculture context was missing. This study evaluated the performance of various object detection models and their variations in the context of plant disease recognition, considering elements like training speed and accuracy. The results presented in this study are not only applicable to the region of the Indian sub-continent but also meaningful to understand the performance comparison in a broader concept.

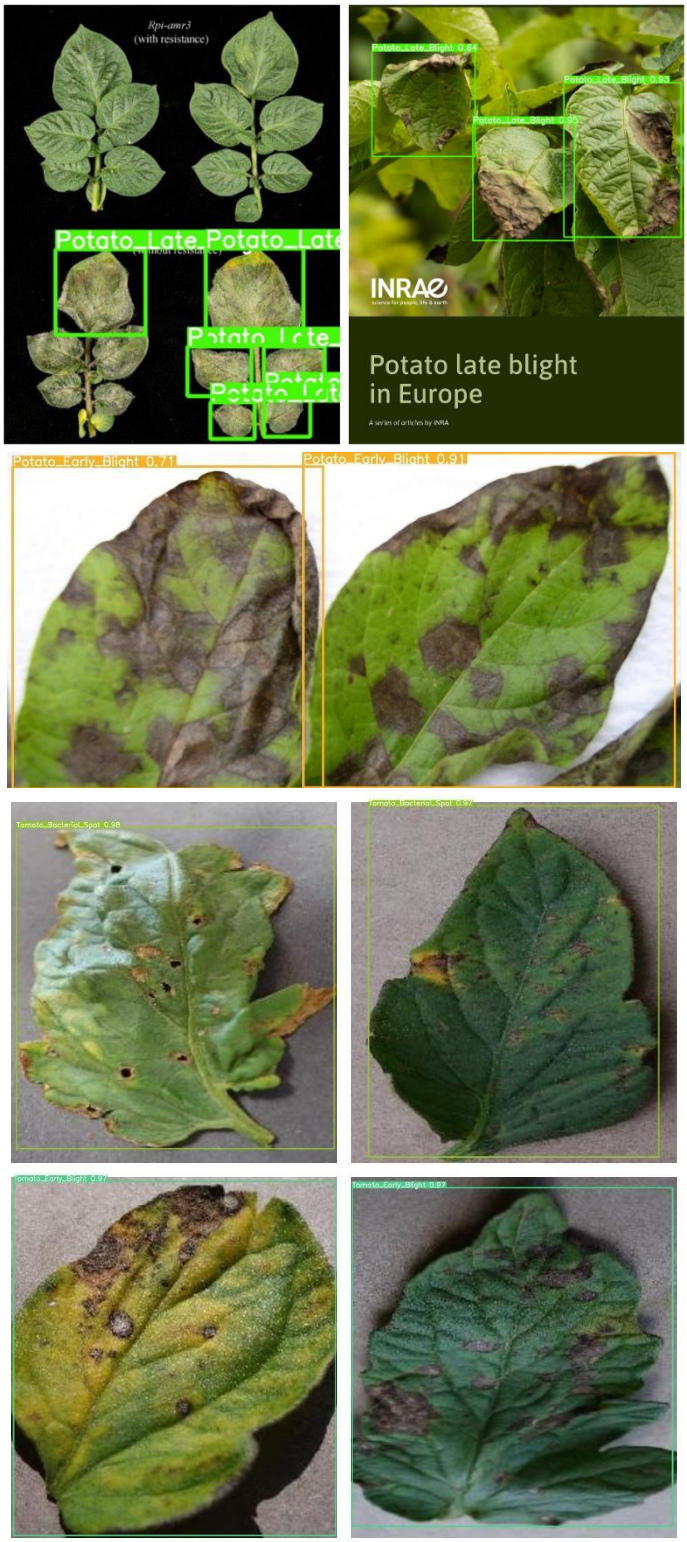

Fig. 10. Successful predictions on test images

The results showed that the YOLOv5 model was more reliable for the task of plant disease detection due to its lightweight and effective architecture. In contrast, despite the SSD model's decent mAP score, its heavy VGG-16 backbone has rendered this approach to be less advantageous in terms of training time and computational cost. Even the largest YOLOv5 model had more accuracy and less training time than SSD. Additionally, the small YOLOv5s model performed surprisingly well, almost matching the performance of the larger YOLOv5x model. However, another variant of YOLO, the Scaled YOLOv4, did not demonstrate a decent performance compared to the other models. The primary contributing factor was the absence of pre-trained weights for this model. Given the relatively small dataset used, the use of pre-trained weights played a huge role in enhancing model performance. In summary, YOLOv5x emerged as the most accurate among the four models examined, while YOLOv5s proved to be the fastest in terms of training speed.

The outcomes of this study will offer valuable insights to future researchers in determining which object detection model to utilize for plant disease detection. The decision largely depends on the user's particular needs. If speed is the most important factor, YOLOv5s is the ideal choice. On the other hand, YOLOv5x is the best option for those who seek higher accuracy. The dataset should also be considered while making a choice, as smaller models tend to perform well with smaller datasets, as evidenced by the nearly equivalent performance of YOLOv5s when compared to YOLOv5x. Transfer learning should also be considered for smaller datasets to achieve optimal results.

9 References

1. Food And Agriculture Organization: Food Security: Risks and Responses, FAO Publications (2015)
2. Iverson, K. and others: Frontier Technology Issues, UN Department of Economic and Social Affairs, Economic Analysis (2021)
3. Ristaino J. and others: The persistent threat of emerging plant disease pandemics to global food security. Journal of National Academy of Science, USA 118(23), doi: 10.1073/pnas.2022239118 (2021)
4. Shurtleff, M., Pelczar, M., Rita M. and Kelman, A.: “Plant Disease”. Encyclopedia Britannica. https://www.britannica.com/science/plant-disease, last accessed 2023/09/12.
5. IBM: What is Deep Warning, https://www.ibm.com/topics/deep-learning, last accessed 2023/09/12.
6. Tugrul, B., Elfatimi, E., Eryigit, R.: Convolutional Neural Networks in Detection of Plant Leaf Diseases: A Review. Advances in Agricultural Engineering Technologies and Application, 12(8), 1192, https://doi.org/10.3390/agriculture12081192 (2022).
7. Emmanuel, T.: PlantVillage Dataset. Kaggle resource: https://www.kaggle.com/datasets/emmarex/plantdisease, last accessed 2023/09/12
8. Singh, D., Jain, N., Jain, P., Kayal, P., Kumawat, S., & Batra, N.: PlantDoc: A dataset for visual plant disease detection. In Proceedings of the 7th ACM IKDD CoDS and 25th COMAD, pp. 249-253 (2020).
9. Chen, Z., Wu, R., Lin, Y., Li, C., Chen, S., Yuan, Z., Chen, S. and Zou, X.,: Plant Disease Recognition Model Based on Improved YOLOv5. Agronomy, 12(2), p.365 (2022)
10. Ge, Z., Liu, S. and others: YOLOX: Exceeding YOLO Series in 2021. Computer vision and Pattern Recognition, Cornell Archive, https://arxiv.org/abs/2107.08430 (2021)
11. Wang, C., Bochkovskiy, A., Liao, H.: Scaled-YOLOv4: Scaling Cross Stage Partial Network, Cornell Archive, https://arxiv.org/abs/2011.08036 (2021)

12. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. and Berg, A.: “SSD: Single shot multibox detector”. In: Proceedings of the ECCV, pp. 21–37, 2016.
13. Anjanadevi, B., Charmila, I., Ns, A., & Anusha, R.: An improved deep learning model for plant disease detection. Int. J. Recent Technol. Eng.(IJRTE), 8(6), 5389-5392 (2020).
14. Hassan, S., Maji, A., Jasinski, M., Leonowicz, Z. and Jasin´ska, E.: Identification of Plant-Leaf Diseases Using CNN and Transfer-Learning Approach. Journal of Electronics, 10(12), p.1388 (2021).
15. Xiang, S., Liang, Q., Sun, W., Zhang, D. and Wang, Y.: L-CSMS: novel lightweight network for plant disease severity recognition. Journal of Plant Diseases and Protection, 128(2), pp.557-569 (2021).
16. Kakade, N. and Ahire, D.: Real Time Grape Leaf Disease Detection, International Journal of Advance Research and Innovative Ideas (IJARIIE), 1(4) pp. 598-610 (2015).
17. Fuentes, A., Yoon, S., Kim, S. and Park, D.: A Robust Deep- Learning-Based Detector for Real-Time Tomato Plant Diseases and Pests Recognition. Intl Journal of Sensors, 17(9), p.2022 (2017).
18. Mohanty, S., Hughes, D. and Salathe, M.: Using Deep Learning for Image-Based Plant Disease Detection. Frontiers in Plant Science, 7, 1419 (2016).
19. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A.: You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779-788 (2016).
20. Redmon, J., & Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018).
21. Ultralytics: YOLOv5: The friendliest AI architecture you'll ever use. Developed by Ultrlytics Inc. https://ultralytics.com/yolov5, last accessed 2023/09/12.
22. Morbekar, A., Parihar, A., & Jadhav, R.: Crop disease detection using YOLO. In 2020 international conference for emerging technology (INCET) (pp. 1-5). IEEE (2020).
23. Ponnusamy, V., Coumaran, A., Shunmugam, A. S., Rajaram, K., & Senthilvelavan, S.: Smart glass: real-time leaf disease detection using YOLO transfer learning. In 2020 International Conference on Communication and Signal Processing (ICCSP) (pp. 1150-1154). IEEE (2020).
24. Nguyen, V. T., Duong, T. Q., Le, T. D., & Nguyen, A. T. D: Deep learning-based methods for plant disease. In Future Data and Security Engineering. Big Data, Security and Privacy, Smart City and Industry 4.0 Applications: 7th International Conference, FDSE 2020, Quy Nhon, Vietnam, November 25–27, 2020, Proceedings 7 (pp. 166-177). Springer Singapore (2020).
25. Li, M., Zhang, Z., Lei, L., Wang, X. and Guo, X.: Agricultural Greenhouses Detection in High-Resolution Satellite Images Based on Convolutional Neural Networks: Comparison of Faster R-CNN, YOLO v3 and SSD. Sensors, 20(17), p.4938 (2020).
26. Kim, J. A., Sung, J. Y., & Park, S. H.: Comparison of Faster-RCNN, YOLO, and SSD for real-time vehicle type recognition. In 2020 IEEE international conference on consumer electronics-Asia (ICCE-Asia) (pp. 1-4). IEEE (2020).
27. Deepa, R., Tamilselvan, E., Abrar, E. S., & Sampath, S.: Comparison of yolo, ssd, faster rcnn for real time tennis ball tracking for action decision networks. In 2019 International conference on advances in computing and communication engineering (ICACCE) (pp. 1-4). IEEE (2019).
28. Mekhalfi, M., Nicolo, C., Bazi, Y., Rahhal, M., Alsharif, N. and Maghayreh, E.: Contrasting YOLOv5, Transformer, and Efficient- Det Detectors for Crop Circle Detection in Desert. IEEE Geoscience and Remote Sensing Letters, 19, pp.1-5 (2022).
29. Cvat.org: Computer Vision Annotation Tool. https://cvat.org/, last accessed 2023/09/12.
30. Kang, S. H., & Park, J. S. (2023). Aligned Matching: Improving Small Object Detection in SSD. Sensors, 23(5), 2589.