International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Performance Evaluation of YOLOv8 and Segment Anything Model for Auto Annotation of Crop and Weed Images in Pigeon Pea Production System

Vaibhav D. Dhore 1[1*], Vijay K. Sambhe[2], Mohan S. Khedkar[3], Shrinivas A. Khedkar[4] ,Seema C. Shrawne[5]

Submitted:13/03/2024 **Revised**: 28/04/2024 **Accepted**: 05/05/2024

Abstract: India's agricultural industry generates more than $375 billion every year. India ranks second in agricultural output. To improve the agricultural output Precision Agriculture is used. But one of the most important concerns in Precision Agriculture is crop and weed detection. As a result, robotic weeding techniques are utilized to control weeds. In robotic weeding, accurate crop and weed detection and localization in the unstructured field remains a substantial challenge, necessitating supervised modeling using annotated data. The process of creating annotated data is quite time-consuming. Also, the dataset for all the crops and weeds is not present.

In this work, an attempt is made to collect a real-time dataset of the Pigeon Pea dataset using a mobile camera and drone. There are a total of 1727 images in the dataset. Initially, 137 images are manually annotated using roboflow.com which are then used to train all of the YOLOv8 variants for epochs ranging from 10 to 1,000. The YOLOv8n variant has the shortest inference time at 2.8ms which is selected to train the remaining 1590 unannotated images. The predicted bounding boxes of YOLOv8n are given input to the segment anything model that generates the annotations.

The manually and automatically annotated images are merged to create the new dataset. Again, all of the YOLOv8 variants are trained and tested on the new dataset for epochs ranging from 10 to 1,000. Following the inclusion of automated annotations, the values of accuracy, recall, mean average precision@50, and mean average precision@50-95 rose by 9.79%, 38.63%, 13.99%, and 18.43%, respectively. The YOLOv8n provides the shortest inference time of 3.8ms on a new data set. Also, using automatically annotated data, approximately 132.5 hours required for annotation of unlabelled images are saved. This effort will contribute to the advancement of crop and weed detection studies in the pigeon pea production system, including disease prediction, yield prediction, and automated weed removal.

Keywords: *Auto Annotation, Crop and Weed Detection, YOLOv8 model, Segment Anything Model (SAM)*

1. Introduction

By 2050, the population worldwide is supposed to be increased to nine billion. To fulfill the anticipated demand, agricultural production must expand by almost 70% [1]. Nowadays, the agricultural industry is facing numerous difficulties, which include climate change, the depreciation of arable land, and water scarcity. Strategies to address these problems can be found in precision agriculture or smart agriculture. [1][2][3].

Weeds can negatively affect crop yields and quality and spread swiftly and unintentionally [4]. Crops and weeds compete for food, water, sunlight, and growth space [5]. Farmers must therefore use resources to lessen weeds. Numerous factors influence the methods used to reduce the effect of weeds.

Automation of weed management has become attractive as labor costs have increased and people's concerns about their health and the environment have grown [7]. Automatic weed management techniques can be advantageous from an economical and environmental perspective. Utilizing a machine to eliminate weeds lowers labor expenses, and selective spraying methods can reduce the quantity of herbicides used [8].

Detecting and identifying weeds is the necessary initial step in the development of an Autonomous weed management system [7]. It is difficult to identify weeds in crops since they frequently have similar shapes and colors as a crop. Figure 1 depicts different crops with weeds growing among them and corresponding problems like occlusion, texture, color similarity effect of lighting conditions, etc.

The four main processes of a typical weed detection

[1] *Research Scholar, Department of Computer Engineering, Veermata Jijabai Technological Institute, Mumbai, India*
ORCID ID : 0000-0003-1650-6535
[2] *Assistant Professor, Department of Computer Engineering, Veermata Jijabai Technological Institute, Mumbai, India*
ORCID ID : 0000-0003-4224-8840
[3] *Research Scholar, Department of Computer Engineering, Veermata Jijabai Technological Institute, Mumbai, India*
ORCID ID : 0009-0000-0596-267X
[4] *Research Scholar, Department of Computer Engineering, Veermata Jijabai Technological Institute, Mumbai, India*
ORCID ID : 0000-0002-1967-4799
[5] *Research Scholar, Department of Computer Engineering, Veermata Jijabai Technological Institute, Mumbai, India*
ORCID ID : 0009-0002-0281-5003
** Corresponding Author Email: vddhore@ce.vjti.ac.in*

system are data collection, preprocessing, feature extraction, and classification of weeds [10]. The use has accomplished these steps of several developing technologies. The identification and classification of weeds is the most important phase. The automatic detection of weed species has been more popular in recent years due to advancements in technology, specifically in graphical processing units (GPUs), and the application of Machine Learning (ML) approaches [11] [12] [13].

Deep Learning (DL) is a significant subpart of ML. DL models perform better than ML models for image segmentation, object detection, etc. It might be challenging to identify and choose differentiating traits using ML approaches because crops and weeds can sometimes be similar. Based on robust feature learning abilities, DL models can successfully address this issue. Crop and weeds are automatically classified using RGB, multispectral, and hyperspectral images. A large number of images are required to be collected and then needs to be annotated. The dataset of a few crops and weeds is present. Currently, there is no dataset of Pigeon Pea is publicly available. So, we collected the dataset of Pigeon Pea. After data collection, it needs to be annotated. The annotation process is very time-consuming and tedious. Hence it is very much required to annotate the dataset automatically.

Recently, Meta AI proposed the Segment anything model in April 2023. It automatically segments the image. But it works on the prompt engineering principle. It is not able to segment the crop and weed images correctly. There is a need to provide objects bounding boxes to the SAM which leads to the improvement in the segmentation [14]. These bounding boxes are provided by the trained YOLOv8 to the SAM.

The YOLOv8 model was released in January 2023 and it is a state-of-the-art model for object detection and image segmentation created by Ultralytics. The YOLOv8x model attains 53.9 as mAP@50-95 value on the MS COCO 2017 benchmark dataset [15].

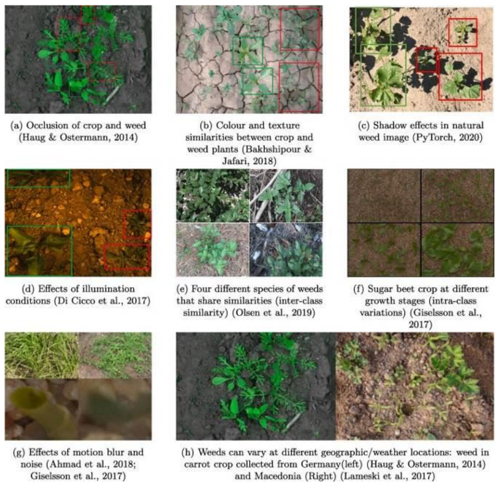

(a) Occlusion of crop and weed (Haug & Ostermann, 2014)
(b) Colour and texture similarities between crop and weed plants (Bakhshipour & Jafari, 2018)
(c) Shadow effects in natural weed image (PyTorch, 2020)
(d) Effects of illumination conditions (Di Cicco et al., 2017)
(e) Four different species of weeds that share similarities (inter-class similarity) (Olsen et al., 2019)
(f) Sugar beet crop at different growth stages (intra-class variations) (Giselsson et al., 2017)
(g) Effects of motion blur and noise (Ahmad et al., 2018; Giselsson et al., 2017)
(h) Weeds can vary at different geographic/weather locations: weed in carrot crop collected from Germany(left) (Haug & Ostermann, 2014) and Macedonia (Right) (Lameski et al., 2017)

Fig.1. Problems in Crop and Weed Detection [12]

In research [16], the YOLOv8 model was trained on images from public datasets and the internet. The promising results are observed. However, there is a lack of diversity of images in the dataset. The authors in the study [17] presented CCCS-YOLO as a lightweight model by making changes in the YOLOv5 architecture. They integrated a faster block with a C3 module in the neck, enhanced the context aggregation module, replaced the Upsample module with a Lightweight context-aware ReAssembly Feature (CARAFE), and replaced the NMS and CIoU module with SOFT-NMS-EIOU module. The customized model is trained on two publicly available datasets Seasame and Sugabeet. The CCCS-YOLO achieved 79.5% and 58.6% as mAP and mAP50-95 values respectively. The performance of variants of YOLOv7, YOLOv8, and Faster RCNN is evaluated in [18]. The YOLOv7- tiny model attained the highest mAP of 88.5% and lowest inference time of 2.7ms. The improved YOLOv7 is proposed in [19]. The author used F-ReLU as an activation function of a convolutional module and appended the maxpool multihead self-attention (M-MHSA) module. The precision, recall, and mAP of the model were 94.96%, 91.25%, and 96.62% respectively. The BSS-YOLOv8 is proposed in [20]. The authors modified the YOLOv8 model. The authors integrated the BoTNet module into the backbone network, links local and global features through multi-head self-attention mechanism (MHSA), and attached a detection layer in the neck of the model. The model attained a mAP of 92.6%, a precision of 91.1%, and a recall of 86.7%. The authors in [21] suggested lightweight YOLOv7 model (LW-YOLOv7). They modified YOLOv7 architecture. They used GhostNet as a backbone network and replaced the path aggregation network (PAN) with a Bidirectional feature pyramid network (Bi-FPN) to enhance location and semantic information. The performance of LW-YOLOv7 is compared with YOLOv3 to YOLOv8 and Faster RCNN. It achieved an 89.3% Precision, an 85.5% Recall, and 93.2% mAP. The performance of YOLOv8 is compared with all the YOLO models in [22]. Among all the YOLO models YOLOv8 reported the highest mAP of 84.7%.

For a generation of automatic annotation [23] used the Segment Anything Model (SAM) and Grounding DINO. The authors in [24] used the Hough transform and Convolutional neural network to automatically annotate the images of Sugar-beet.

After going through the literature four observations are made. First, there is no dataset available for the Pigeon Pea. Second, the researchers modified the different YOLO architectures. Third, YOLOv8 is the state-of-the-art model for object detection. Fourth, automatic annotation is not explored by many researchers.

Based on observations of the literature the main contribution of this paper is finalized which is given below:

1. Collect the dataset of Pigeon Pea and corresponding weeds.
2. Annotate the Pigeon Pea dataset.
3. Train the YOLOv8 variants on annotated data.
4. Automatically generate the annotations for unannotated data using the trained YOLOv8 model and Segment Anything Model.
5. Evaluate the performance of YOLOv8 variants on manually annotated images and automatically generated images.

This paper is organized into five sections. Section 2 has details of dataset collection, preparation, and models used for training. The experimental result and discussion are covered in section 3. The conclusion is presented in section 4.

2. Material and Methods

This section covers details of dataset collection, preprocessing, experimental environment, and the proposed methodology.

2.1. Dataset

In India, the Pigeon Pea is one of the important crops. So, the pigeon pea crop is selected for data collection. Afterwards, three fields are selected from the Vidharbha region. Then the devices' mobile camera and drone are finalized for data acquisition. After which the data is prepossessed and annotated. In this section data collection, prepossessing, and data annotation process are described in detail.

2.1.1. Data acquisition

For flexibility drone, DJI Mavic Air 2S and mobile camera are selected for data acquisition. The drone has a 20-megapixel CMOS camera sensor. The images are captured by drone at 20 cm height. The images are captured using a drone on 19th July 2023. The number of captured images is 93. The mobile has a 64-megapixel camera. The images are captured using a mobile camera on the 16th, 17th, 18th, 20th, and 22nd of July 2023. The number of images captured by mobile cameras is 1634. The image captured at different height ranging from 10 cm to 30 cm. Three different fields are selected for data collection. Their details are given in Table 1. The sample images collected in different fields are given in Figure 2.

Fig. 2 a. Crop Images from different fields.

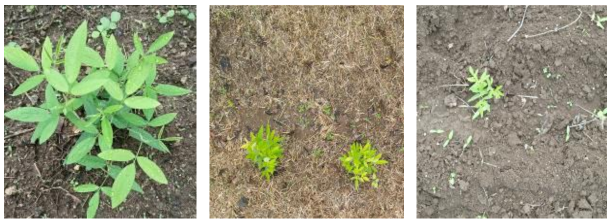


Fig. 2 b. Weed Images from different fields.

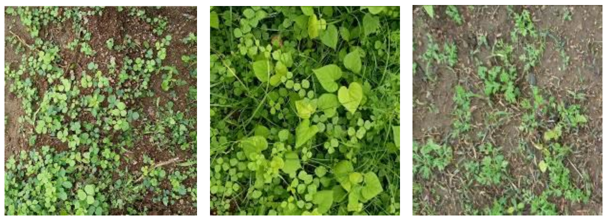


Fig.2 c. Three different fields were selected for data collection.

2.1.2. Data Preprocessing

The raw images have resolutions 2016 x 4480, 4480 x 2016, and 5472 x 3648. The raw images are preprocessed. The various preprocessing operations performed are given below:

2.1.2.1. Noise removal

Image denoising is an essential image processing technique that may be used as a standalone procedure or as a component in another. There are several methods for denoising a picture. It is solved using several algorithms. As a result, noises are identified with nearby information and eliminated utilizing optimum filtering algorithms that do not degrade picture quality while also enhancing the smoothness of the image collected for analysis. In this work Gaussian filter is used for noise removal.

2.1.2.2. Image Normalization

In image normalization pixel intensity range is modified. It helps in reducing the execution time. For grayscale images, the intensity range is 0 to 255. It contains one channel. So, the range is changed from 0 to 1. On the other hand, RGB images contain three channels having a pixel intensity range of 0 to 255. In this case, the three channels' pixel intensity value range is changed from 0 to 1.

2.1.2.3. Image resizing

The images captured using a mobile camera have resolutions as

2016 x 4480 and 4480 x 2016. The drone-captured images have a resolution of 5472 x 3648. All the images are resized to 640 x 640.

Fig. 3. DJI MAVIC AIR 2S Drone.

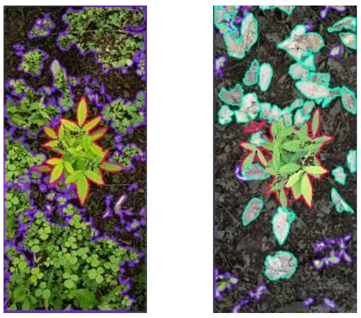
Table 1. Distribution of images collected

SN	Date	Quantity	Capturing Device
1	16th July 2023	137	Mobile Camera
2	17th July 2023	422	Mobile Camera
3	18th July 2023	440	Mobile Camera
4	19th July 2023	93	Drone
5	20th July 2023	256	Mobile Camera
6	22nd July 2023	379	Mobile Camera

2.2. Data Annotation

Initially, 137 images are annotated using roboflow.com. Then these images are used to train YOLOv8 variants. The data is annotated using a smart polygon tool. After annotating the images annotation file is downloaded in YOLOv8 format. The sample annotated images are given in Figure 4.

Fig.4. Annotated of images



2.3. Experimental Environment

The YOLOv8 variants are designed by considering different training environments and inference speeds. They are trained and tested on commodity hardware as well as high-end servers. The environment used in this work is presented in Table 2.

Table 2. System configuration

Item	Description
CPU	Intel Xeon-S 4210R Kit for DL380 Genl0
GPU	HPE NVIDIA Tesla T4
Operating System	Ubuntu 22.04 LTS
Accelerated Environment	CUDA 10.2 CUDNN 7.6.0
Development Environment	Visual Studio code, Python 3.9
Random Access Memory	224 GB

2.4. Model Training

In this experiment, the image pixels of the input network were set to 640 × 640. Adam was used as the model optimizer. While gradient descent optimizations, the updating of model parameters is based on the learning rate. Hence, the learning rate is one of the important hyperparameters. The initial learning rate was 0.01. The batch size indicates the number of images processed simultaneously in each iteration of training. An increase in batch size may lead to a decrease in training time. Sometimes overfitting may occur. Here batch size of 8 is selected. The other parameters are given in the Table 3.

Table 3. Training Parameters

Parameter Name	Value of Range
Image Size	640x640
Learning Rate	0.01
Learning rate frequency	0.1
Momentum	0.937
Batch Size	8
Patience	100
Maximum Training Epochs	500
Decay	0.0005

2.5. YOLOv8 Model

You only look once version 8 was released in January 2023. It can be used for segmentation, classification, and object detection. The detailed architecture of YOLOv8 is given in Figure 5. There are two major networks present in it i.e., backbone, and head. The

The backbone part contains five Convolutional layers, four coarse-to-fine (C2F) modules, and one Spatial pyramid pooling fast (SPPF) layer. The head module contains four C2F modules, four Concat layers, two Convolutional layers, two Upsample layers, and three detect modules. Three C2F modules of the backbone part are connected to three Concat layers of the head part. The three C2F modules of the head part are connected to three Convolutional layers of the detect module. In the C2F module, two bottleneck layers are present in which concatenation of output occurs. It leads to a speed-up of the training process and improvement in gradient flow. In the SPPF layer, two Convolutional, three Maxpooling, and one Concat layer are present. This is an optimized version of the spatial pyramid pooling (SPP) layer. Due to this same output size images are generated for the different input size images. Multiple factors contribute to improving the performance of the YOLOv8 model which includes Mosaic Data Augmentation, decoupled head, C2F module, and SPPF module. There are five different variants of YOLOv8 are available. The variants are nano, small, medium, large, and extra-large. The number of parameters and number of floating-point operations (FLOPS) present in each variant is given in Figures 6 and 7. The parameters and FLOPS are increasing from nano to extra-large variant.

Fig. 6. Number of parameters in YOLOv8 variants

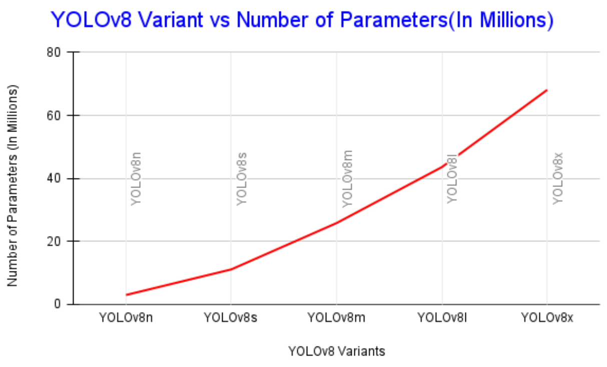
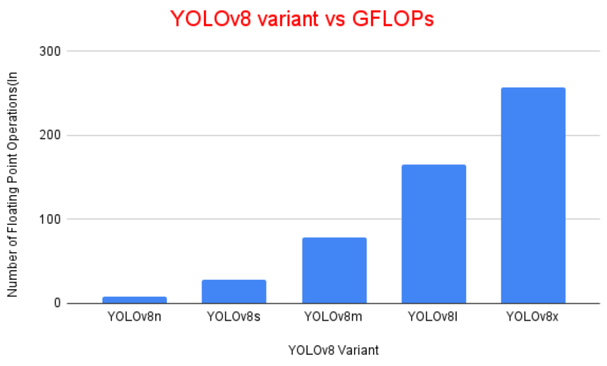


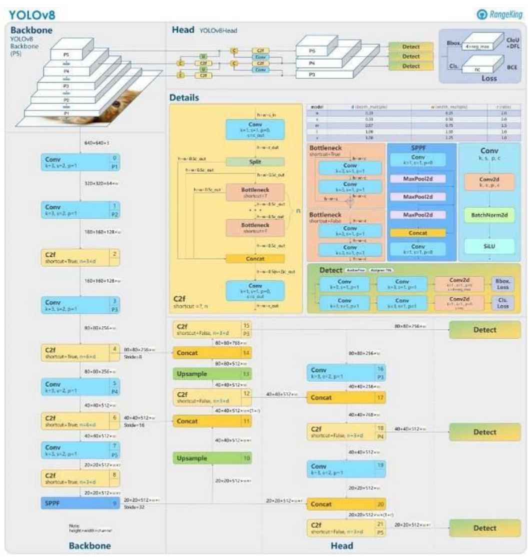
Fig. 7. Floating point operations in YOLOv8 variants



2.6. Segment Anything Model (SAM)

The SAM was proposed by Meta AI in April 2023. It has been trained on a dataset having 11 million images and 1.1 billion masks. Due to this large dataset SAM's performance on zero-shot for different segmentation tasks. The SAM takes points or boxes as input and produces masks for a variety of objects. The architecture of SAM is given in Figure 8. The SAM has three components which are an image encoder, prompt encoder, and mask decoder. The Image Encoder is a pre-trained model that generated one-time embedding. These embedding can be used before prompting the model. The Prompt Encoder is responsible for encoding background points, text, and masks into an embedding vector. The Mask Decoder takes input as embedding from the both Image Encoder and Prompt Encoder and produces segmentation masks. The generated segmentation masks are used to improve the model by updating model weights. In this way, with time model improves and becomes efficient.

Fig.5. YOLOv8 Architecture



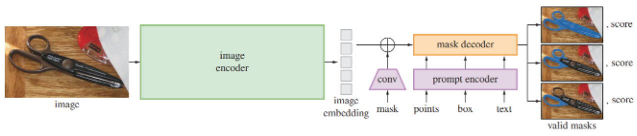


Fig. 8. Segment Anything Model Architecture [17]

2.7. Proposed Methodology

Initially, the 137 images were manually annotated using roboflow.com, and YOLOv8 was trained on those images. Then, the remaining 1590 unannotated images are given input to the trained YOLOv8 model which generates the bounding boxes of prediction. It is then fed to the segment anything model which generates the annotations for the remaining images in YOLOv8 format. The new dataset is created by merging manual annotation and automatic annotations. Later the YOLOv8 is trained and tested on the new dataset. The detailed workflow is given in Figure 9.

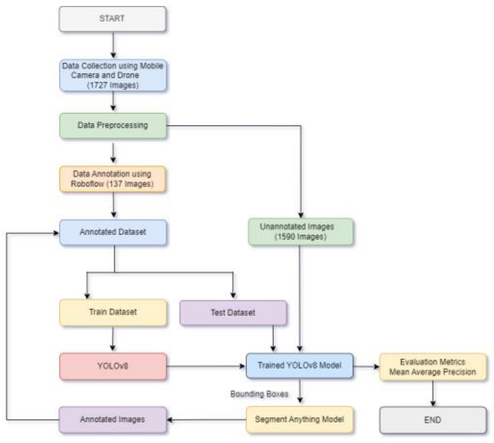

START
Data Collection using Mobile Camera and Drone (1727 Images)
Data Preprocessing
Data Annotation using Roboflow (137 Images)
Annotated Dataset
Unannotated Images (1590 Images)
Train Dataset
Test Dataset
YOLOv8
Trained YOLOv8 Model
Evaluation Metrics Mean Average Precision
Bounding Boxes
Annotated Images
Segment Anything Model
END

Fig. 9. Proposed Methodology

2.8. Performance Metrics

The five variants of YOLOv8 are trained on the dataset and their performance is evaluated based on metrics such as Precision(P), Recall(R), Mean Average Precision, Mean Average Precision at 50% Intersection of Union threshold(mAP@50), and Mean Average Precision at 50 to 95% Intersection of Union threshold(mAP@50-95).

Precision: It represents the percentage of correct positive predictions.

$$Precision = \frac{TruePositive(TP)}{TruePositive(TP) + FalsePositive(FP)} \tag{1}$$

Recall: It represents the percentage of actual positive correct predictions.

$$Recall = \frac{TruePositive(TP)}{TruePositive(TP) + FalseNegative(FN)} \tag{2}$$

Intersection of Union (IoU): It measures the accuracy of an object detector's localization on a given dataset. It determines how much the predicted bounding box coordinates overlap the ground truth box coordinates.

Fig. 10. Intersection Over Union

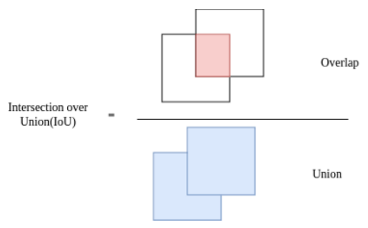
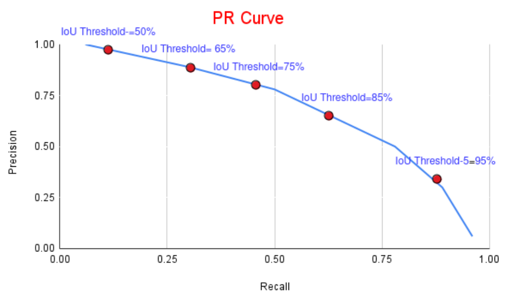


Fig.11. PR-Curve



Average Precision: Average Precision is calculated as the weighted mean of precision at each threshold, where the weight represents the increase in recall over the previous threshold.

Mean Average Precision(mAP): The mAP is calculated by calculating the average precision (AP) for each class and then averaging it over many classes.

$$mAP = \frac{1}{N}\sum_{K=1}^{N} AP_k \tag{3}$$

mAP@50: The mAP@50 is calculated by calculating the average precision (AP) for each class where the threshold of IoU is 50%.

mAP@50-95: The mAP@50-95 is calculated by calculating the average precision (AP) for each class where the threshold of IoU ranges from 50%. to 95% at an interval of 5%.

Precision-Recall Curve (PR-curve): There is a trade-off between precision and recall. So, the PR curve is plotted for different values of the threshold of IoU ranging from 50 to 95%. There is an interval of 5%.

3. Experimental Results

Initially, the 137 images are manually annotated using the roboflow website tool and the YOLOv8 model is trained on those images. The five different variants of YOLOv8 are used for the training. The different version of YOLOv8 are YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x. The training is carried out for different values of epochs such as 10,20,50,100,200,500, and 1000.

3.1. Performance evaluation of YOLOv8 variants on a manually annotated dataset

To evaluate the performance of YOLOv8 variants, they are trained on the manually annotated dataset for different values of epochs ranging from 10 to 1000. The training parameters used are given in Table 3. The experimental results are presented in Table 4. It shows that the epoch value has a significant effect on the performance of detection results. The various variant is compared based on inference time. Among all the variants, YOLOv8n has the least inference time 2.8ms after training for 500 epochs. The model starts overfitting after the 500 epochs. The YOLOv8m achieves highest recall of 0.422, mAP@50 of 0.486, and mAP@50-95 of 29.3. The YOLOv8m starts overfitting after 100 epochs. The least inference time of YOLOv8m is 10.7 ms for epoch value 500. The highest precision 0.6467 is attained by YOLOv8l after training for 200 epochs. Based on the results YOLOv8n weights are used in the prediction of unlabelled dataset.

Figure 14,15. and16 represents the confusion matrix, PR-Curve, and validation and training loss for the YOLOv8n. From the confusion matrix, it can be concluded that the background class is reducing the performance. So, there is a need to add more instances of background in the dataset. The PR-Curve indicates that the AP for different classes and mAP@50. The mAP@50 for all the classes is 0.486.

The effect of epoch values on training time and inference time is represented in Figures 12 and 13. Training time increases with the epochs. The patience value is 100 epochs is used. It indicates that if there is no change in losses for the last 100 epochs, then it stops training. So, in some cases for higher epoch value has less training time as compared to the lower epoch value. In Table 4, the training time YOLOv8m for 1000 epochs is 0.25 hrs which is less than the training time for 500 epochs i.e., 0.269 hrs.

The inference time starts decreasing with the increase in epoch value. But it starts increasing if overfitting occurs. From Figure 14, it can be observed that the inference time of the YOLOv8n model starts decreasing till epoch value 50, and then it starts increasing.

Fig. 12. Inference time for YOLOv8 variants in different epochs

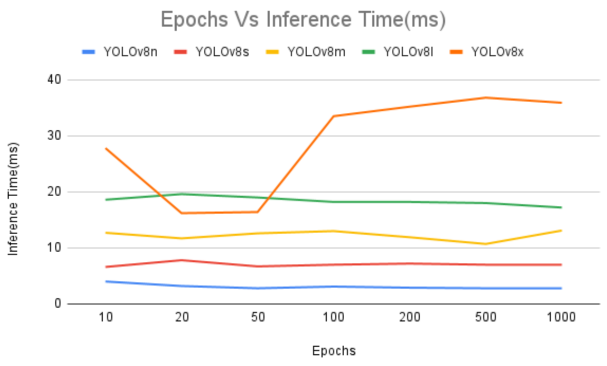
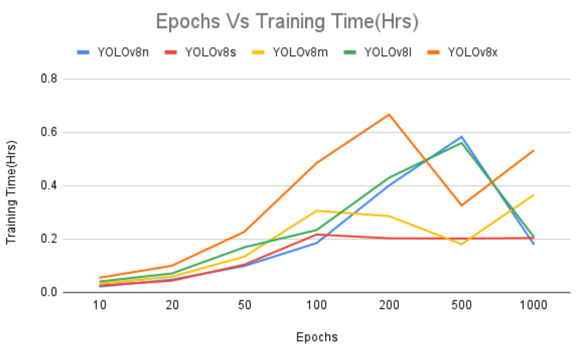


Fig. 13. Training time of YOLOv8 variants in different epochs



3.2. Performance evaluation of YOLOv8 variants on manually and automatically annotated dataset

The manually annotated dataset and automatically annotated dataset by YOLOv8n and SAM are merged to create the new dataset. The different YOLOv8 variants are trained and tested on the newly created dataset. The corresponding experimental results are given in Table 5. Also, the training parameters used are given in Table 3. Table 5 describes the impact of epochs on the performance of various YOLOv8 variants. The least inference time of 3.8 ms is attained by YOLOv8n after training for 50 epochs. After 50 epochs, YOLOv8m starts overfitting. Among all the YOLOv8m performs better. The precision of 0.71, recall of 0.58, mAP50 of 0.556, and mAP50-95 of 0.358 is attained by it. The least time of inference for YOLOv8m is 6.3 ms for epoch value 50. After 50 epochs, YOLOv8m starts overfitting.

Table 4. YOLOv8 variants result in a manually annotated dataset

SN	Variants	Epochs	Precision	Recall	mAP@50	mAP@50-95
		10	0.546	0.35	0.327	0.209
		20	0.568	0.371	0.39	0.257
		50	0.583	0.381	0.399	0.262
1	**YOLOv8n**	100	0.541	0.394	0.402	0.265
		200	0.5906	0.36126	0.38012	0.24888
		500	**0.5711**	**0.35674**	**0.486**	**0.25127**
		1000	0.503	0.383	0.383	0.255
		10	0.558	0.375	0.381	0.244
		20	0.573	0.388	0.408	0.268
2	YOLOv8s	50	0.543	0.386	0.409	0.279
		100	0.573	0.384	0.409	0.274
		200	0.551	0.419	0.411	0.27

		500	0.565	0.393	0.411	0.27
		1000	0.582	0.39	0.408	0.271
		10	0.51	0.364	0.38	0.24
		20	0.542	0.422	0.419	0.283
		50	0.557	0.37	0.411	0.285
3	**YOLOv8m**	**100**	0.588	**0.422**	**0.486**	**0.293**
		200	0.568	0.411	0.419	0.282
		500	0.602	0.392	0.421	0.28
		1000	0.59009	0.36679	0.39134	0.26479
		10	0.473	0.365	0.36	0.229
		20	0.539	0.387	0.403	0.27
		50	0.529	0.409	0.416	0.287
4	**YOLOv8l**	100	0.553	0.387	0.405	0.283
		200	**0.64677**	0.34524	0.36899	0.25552
		500	0.6123	0.34955	0.37664	0.26428
		1000	0.571	0.35	0.386	0.272
		10	0.511	0.36	0.364	0.235
		20	0.552	0.397	0.407	0.271
		50	0.563	0.387	0.419	0.287
5	YOLOv8x	100	0.538	0.397	0.419	0.281
		200	0.63127	0.40565	0.38556	0.26203,
		500	0.53	0.402	0.398	0.269
		1000	0.55998	0.36456	0.36805	0.24509

Fig. 14. Confusion Matrix after training YOLOv8 on sample images

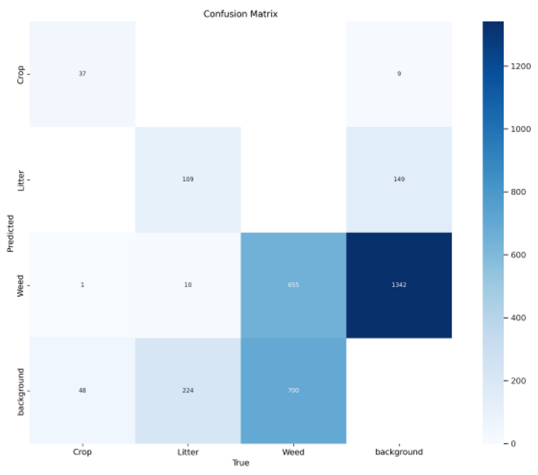


Fig. 15. PR-Curve after training YOLOv8 on sample images

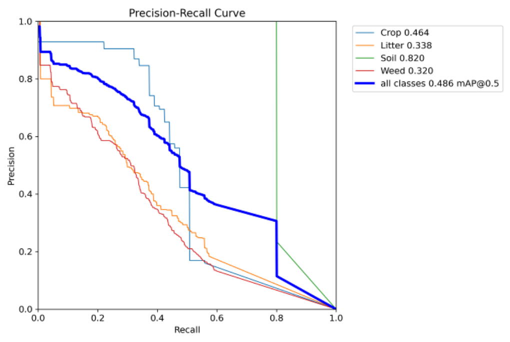
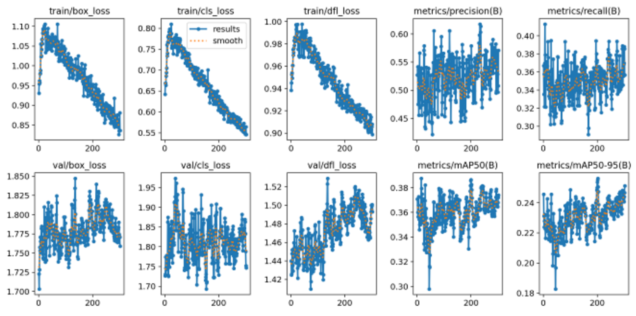


Fig. 16. Results training YOLOv8 on sample images



Figures 17,18 and 19 represent the confusion matrix, PR-

Curve, and validation and training loss for the YOLOv8n. From the confusion matrix, it can be concluded that the background class is reducing the performance. So, there is a need to add more instances of background in the dataset. The PR-Curve indicates that the AP for different classes and mAP@50. The mAP@50 for all the classes is 0.534.

The effect of epoch values on training time and inference time is represented in Figures 20 and 21. Training time increases with the epochs. The patience value is 100 epochs is used. It indicates that if there is no change in losses for the last 100 epochs, then it stops training. So, in some cases for higher epoch value has less training time as compared to the lower epoch value. The inference time starts decreasing with the increase in epoch value. But it starts increasing if overfitting occurs. From Figure 22, it can be observed that the inference time of the YOLOv8n model starts decreasing till epoch value 50, and then it starts increasing.

Table 5. YOLOv8 variants result in manually and automatically annotated dataset

SN	Variant	Epochs	Precision	Recall	mAP@50	mAP@50-95
1	YOLO v8n	10	0.71	0.426	0.458	0.288
		20	0.552	0.519	0.512	0.335
		50	0.598	0.484	0.534	0.351
		100	0.607	0.514	0.554	0.347
		200	0.67	0.52	0.55	0.33
		500	0.6	0.54	0.56	0.34
		1000	0.58	0.56	0.57	0.36
2	YOLO v8s	10	0.593	0.46	0.467	0.303
		20	0.598	0.453	0.524	0.335
		50	0.569	0.483	0.516	0.331
		100	0.596	0.489	0.512	0.327
		200	0.677	0.412	0.515	0.329
		500	0.65	0.42	0.52	0.33
		1000	0.63	0.45	0.53	0.34
3	**YOLO v8m**	10	**0.717**	0.408	0.446	0.297
		20	0.569	0.489	0.498	0.331
		50	0.548	0.511	0.54	0.348
		100	0.525	**0.585**	0.544	**0.358**
		200	0.646	0.46	**0.556**	0.352
		500	0.62	0.48	0.56	0.37
		1000	0.6	0.51	0.55	0.36
4	YOLO v8l	10	0.697	0.432	0.461	0.302
		20	0.575	0.558	0.52	0.343
		50	0.646	0.549	0.531	0.338
		100	0.528	0.52	0.521	0.345
		200	0.55	0.4	0.38	0.38
		500	0.54	0.45	0.423	0.39
		1000	0.532	0.47	0.421	0.381
5	YOLO v8x	10	0.708	0.418	0.463	0.308
		20	0.722	0.409	0.487	0.32
		50	0.517	0.573	0.536	0.348
		100	0.559	0.519	0.516	0.334
		200	0.578	0.491	0.563	0.342
		500	0.54	0.53	0.58	0.36
		1000	0.52	0.57	0.556	0.35

Fig. 17. Confusion Matrix after training YOLOv8 on all the images

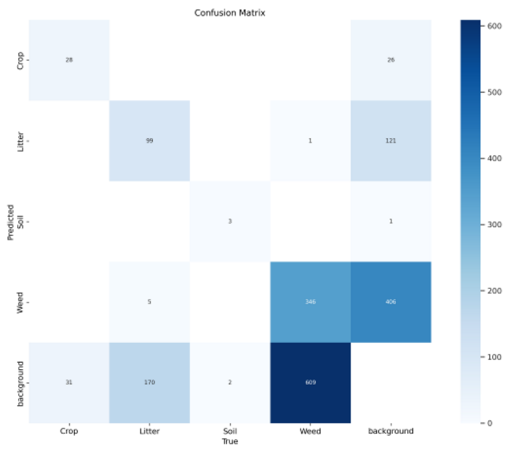


Fig. 18. PR-Curve after training YOLOv8 on all the images

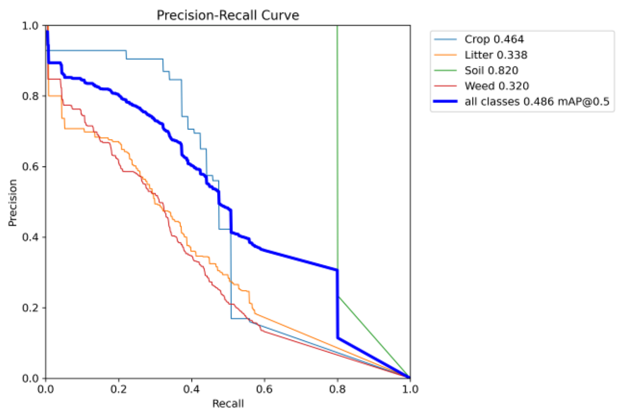


Fig. 19. Results after training YOLOv8 on all the images

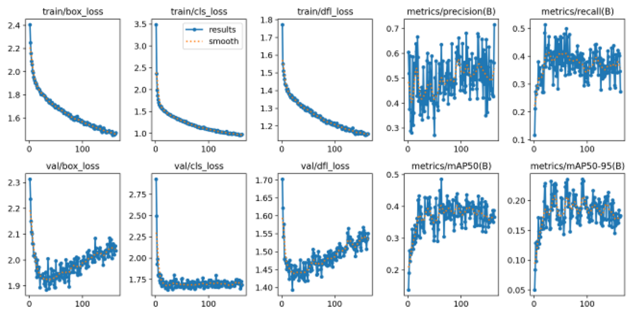


Fig. 20. Effect of epochs on training time in the case of the new dataset

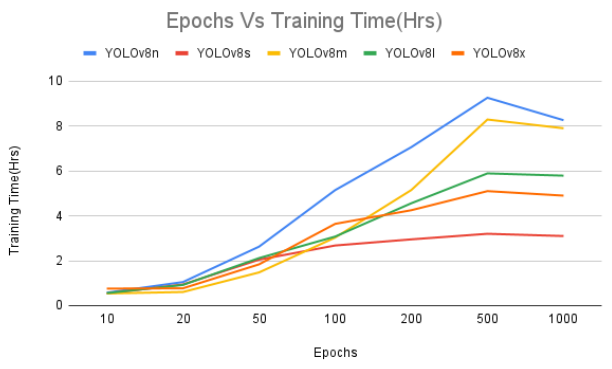
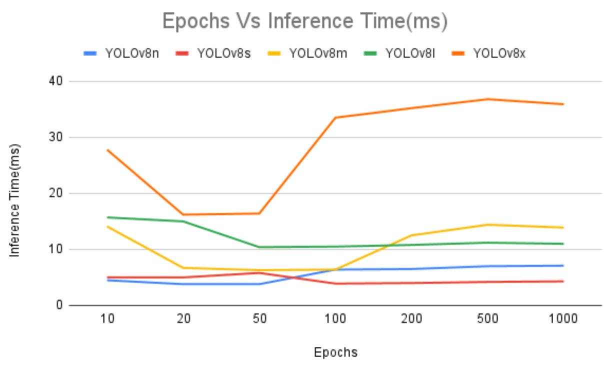


Fig. 21. Effect of epochs on inference time in the case of the new dataset



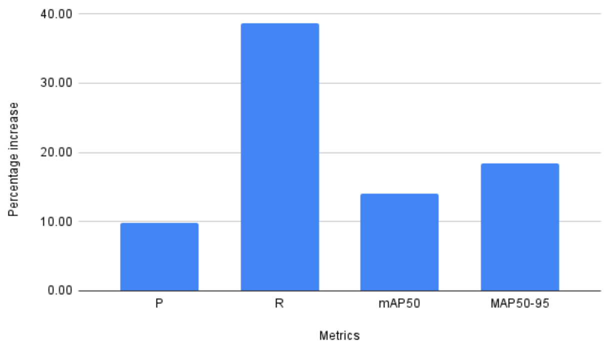
3.3. Effect of addition of automatically annotated dataset

The new dataset is created by merging the manually annotated images and automatically annotated images. Due to the increase in the number of images the performance of the YOLOv8 variant improves. In this process, the large amount of annotation time is saved. To annotate an image an average of 5 minutes is required. To annotate 1590 images, it will take approximately 132.5 hours. Due to the automatic annotation, 132.5 hours of annotations are saved. After the addition of automatic annotations, the values of precision, recall, mean average precision@50, and mean average precision@50-95 are increased by 9.79%, 38.63%., 13.99%, and 18.43% respectively. The change in performance after the addition of automatically annotated images is depicted in Figure 22. The values of precision, recall, mAP@50, and mAP@50-95 obtained after training on the two datasets are given in Table 6.

Table 6. Comparison of performance of YOLOv8 on two datasets

Metrics	Manually Annotated Dataset	Manually + Automatically Annotated Dataset
Precision	0.6467	71
Recall	0.422	58.5
mAP@50	0.486	55.4
mAP@50-95	0.293	34.7

Fig. 22. Percentage change in performance metrics after training on the new dataset



3.4. Visualization of results

Two random images are selected and given input to the best performing YOLOv8 variant i.e., YOLOv8n. of both the cases. The actual and predicted images are given in Figure 25. From the prediction images, it can be inferred that YOLOv8n trained on the new dataset performs better than the YOLOv8n trained on manually annotated datasets.

Fig. 23 a. Actual images

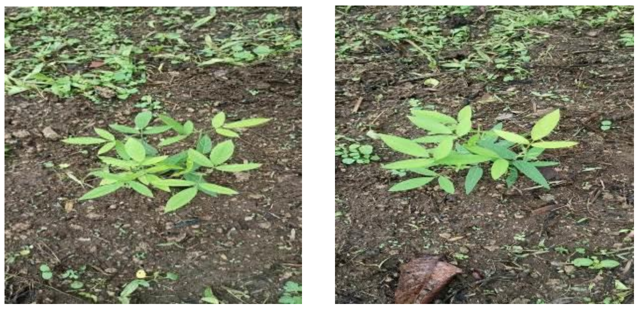


Fig. 23 b Prediction results of YOLOv8n trained on a manually annotated dataset

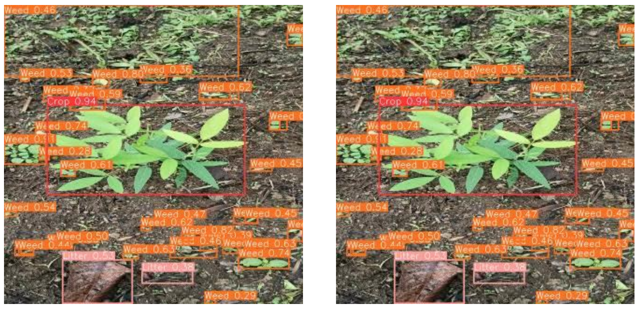
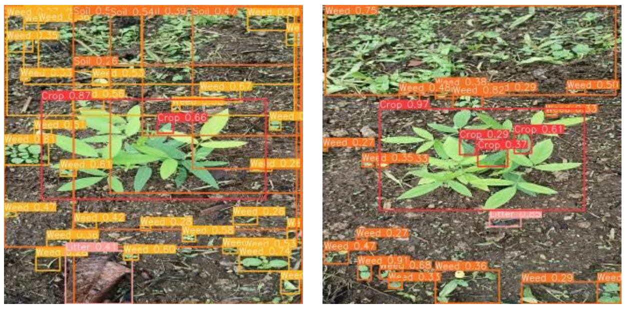


Fig. 23 c. Prediction results of YOLOv8n trained on the new dataset



4. Conclusion

To address the challenge of annotation of crops and weeds in Pigeon Pea, this study first collected the corresponding real-time data from the three different fields on different days using a mobile camera and drone. Secondly, the 137 images are annotated manually. These images are used to train different YOLOv8 variants.

The training is carried out for epoch values ranging from 10 to 1000. Among all the variants YOLOv8n attains the least inference time as 2.8ms. On the other hand, YOLOv8m achieves the highest recall of 0.422, mAP@50 of 0.486, and mAP@50-95 of 0.293. The weights of YOLOv8n are selected for prediction bounding boxes of the unlabelled data. The trained YOLOv8n and SAM generated the annotations for the unlabelled data. The manually annotated and automatically created datasets are merged to create a new dataset. By using automatically annotated data approximately 132.5 hours are saved.

After training YOLOv8 variants on the new dataset, YOLOv8n again achieved the least inference time at 3.8ms. Again, the highest precision of 0.71, recall of 0.558, mAP@50 of 0.554, and mAP@50-95 of 0.347 is attained by YOLOv8m. The performance is improved after adding the automatically annotated dataset. After the addition of automatic annotations, the values of precision, recall, mean average precision@50, and mean average precision@50-95 are increased by 9.79%, 38.63%., 13.99%, and 18.43% respectively. Also, approximately 132.5 man-hours are saved due to automatic annotation.

After analyzing the confusion matrix, it has been observed that the background is impacting the overall performance. So, there is a need to add more background images to the dataset. This work can be extended further by performing augmentation on the newly created dataset to increase the instances of various classes. This work will help to improve the research in crop and weed detection in the pigeon pea production system for disease prediction, yield prediction, automatic weed removal, etc.

Acknowledgments

We thank our colleagues from Veermata Jijabai Technological Institute, Mumbai who provided insight and expertise that greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

Conflicts of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work carried out in this paper.

References

[1] P. Radoglou-Grammatikis, P. Sarigiannidis, T. Lagkas, and I. Moscholios, “A compilation of UAV applications for precision agriculture,” Computer Networks, vol. 172, p. 107148, 2020.

[2] R. Lal, “Soil structure and sustainability,” Journal of sustainable agriculture, vol. 1, no. 4, pp. 67–92, 1991.

[3] S. K. Seelan, S. Laguette, G. M. Casady, and G. A. Seielstad, “Remote sensing applications for precision agriculture: A learning community approach,” Remote sensing of environment, vol. 88, no. 1-2, pp. 157–169, 2003.

[4] D. Patel and B. Kumbhar, “Weed and its management: A major threats to crop economy,” J. Pharm. Sci. Biosci. Res, vol. 6, pp. 453–758, 2016.

[5] N. Iqbal, S. Manalil, B. S. Chauhan, and S. W. Adkins, “Investigation of alternate herbicides for effective weed management in glyphosate-tolerant cotton,” Archives of Agronomy and Soil Science, 2019.

[6] J. S. Holt, “Principles of weed management in agroecosystems and wildlands1,” Weed Technology, vol. 18, no. sp1, pp. 1559–1562, 2004.

[7] B. Liu and R. Bruch, “Weed detection for selective spraying: A review, “Current Robotics Reports, vol. 1, no. 1, pp. 19–26, 2020.

[8] P. Lameski, E. Zdravevski, and A. Kulakov, “Review of automated weed control approaches An environmental impact perspective,” in International Conference on Telecommunications, pp. 132–147. Springer, 2018.

[9] A. M. Hasan, F. Sohel, D. Diepeveen, H. Laga, and M. G. Jones, “A survey of deep learning techniques for weed detection from images,” Computers and Electronics in Agriculture, vol. 184, p. 106067, 2021.

[10] S. Shanmugam, E. Assuncao, R. Mesquita, A. Veiros, and P. D. Gaspar, “Automated weed detection systems: A review,” KnE Engineering, pp. 271–284, 2020.

[11] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” nature, vol. 521, no.7553, pp. 436–444, 2015.

[12] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang,G. Wang, J. Cai, et al., “Recent advances in convolutional neural networks, “Pattern recognition, vol. 77, pp. 354–377, 2018.

[13] J. Yu, S. M. Sharpe, A. W. Schumann, and N. S. Boyd, “Deep learning for image-based weed detection in turfgrass,” European Journal of Agronomy, vol. 104, pp. 78–84, 2019.

[14] Kirillov, Alexander, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, et al. "Segment anything." arXiv preprint arXiv:2304.02643 (2023).

[15] Terven, J.; Cordova-Esparza, D. A Comprehensive Review of YOLO: From YOLOv1 to YOLOv8 and Beyond. arXiv 2023, arXiv:2304.00501

[16] Wang, Y. A Deep Learning-based Approach for Vision-based Weeds Detection.

[17] Zhang, C., Liu, J., Li, H., Chen, H., Xu, Z., & Ou, Z. (2023). Weed Detection Method Based on Lightweight and Contextual Information Fusion. Applied Sciences, 13(24), 13074.

[18] Hasan, A. M., Diepeveen, D., Laga, H., Jones, M. G., & Sohel, F. (2024). Object-level benchmark for deep learning-based detection and classification of weed species. Crop protection, 177, 106561.

[19] Li, J., Zhang, W., & Li, Q. (2024). Weed detection in soybean fields using improved YOLOv7 and evaluating herbicide reduction efficacy. Frontiers in Plant Science, 14, 1284338.

[20] Guo, B., Ling, S., Tan, H., Wang, S., Wu, C., & Yang, D. (2023). “Detection of the Grassland Weed Phlomoides umbrosa Using Multi-Source Imagery and an Improved YOLOv8 Network’. Agronomy, 13(12), 3001.

[21] Zhao, K., Zhao, L., Zhao, Y., & Deng, H. (2023). “Study on lightweight model of maize seedling object detection based on YOLOv7”. Applied Sciences, 13(13), 7731.

[22] Khalid, S., Oqaibi, H. M., Aqib, M., & Hafeez, Y. (2023). “Small Pests Detection in Field Crops Using Deep Learning Object Detection”. Sustainability, 15(8), 6815.

[23] Rana, S., Gerbino, S., Barretta, D., Carillo, P., Crimaldi, M., Cirillo, V & Sarghini, F. (2024)” Rafanoset: Dataset of Manually and Automatically Annotated Raphanus Raphanistrum Weed Images for Object Detection and Segmentation” Available at SSRN 4720646.

[24] Boysen, J. J., & Stein, A. (2022), “ AI-supported data annotation in the context of UAV-based weed detection in sugar beet fields using Deep Neural Networks”. In GIL Jahrestagung (pp. 63-68).