Smart Agricultural Technology 7 (2024) 100386

Contents lists available at ScienceDirect

Smart Agricultural Technology

journal homepage: www.journals.elsevier.com/smart-agricultural-technology

Enhanced rendering-based approach for improved quality of instance segmentation in detecting green gram (*Vigna Rediata*) pods

Nagaraj V. Dharwadkar [a,*], RajinderKumar M. Math [b]

[a] Department of Computer Science, School of Computer Science, Central University of Karnataka, Kalaburagi, India
[b] Department of Electronics and Communication Engineering, B.L.D.E. Association's V.P Dr. P.G. Halakatti College of Engineering and Technology, Ashram Road, Vijayapur, India

ARTICLE INFO

Keywords:
Object detection
Detectron2
Pointrend
Green gram pods
Instance segmentation

ABSTRACT

The emergence of Artificial Intelligence, deep learning, and current computer vision algorithms are the main contributors to innovations in the agricultural domain. The most recent detection algorithms capable of giving real-time detections at the edge nodes tackle most agricultural problems, such as disease, pest or insect detections, and maturity level detection of crops (fruits and vegetables). Modern harvesters and fruit-picking robots rely heavily on the detection capability of the algorithms used. Various detection algorithms have been proposed and used in literature, having good performance in terms of mean average precision. Still, the current agricultural systems require not only high mean average accuracy but also algorithms should have high inference speeds. The research proposes a Detectron2-based framework with PointRend (Point-based Rendering), capable of providing enhanced, high-quality pixel-level instance segmentation in identifying and detecting green gram pods or Mung Bean (*Vigna Radiata*) in natural field conditions rendering crisp and smooth boundaries for accurately locating the green gram pods. The results indicate that the proposed framework outperforms the famous Mask R-CNN model to obtain higher mean average precision and improved quality of detections.

1. Introduction

Recent developments in Artificial Intelligence (AI), advanced computer vision algorithms, availability of spatial and temporal data from the agricultural field, satellite imagery, and affordable cloud-based computational resources have made commendable progress in achieving agriculture sustainability. Modern agrarian practices aim at effectively utilizing the available agricultural resources to maximize farm productivity that fetches higher consumer returns. This agricultural practice method is called Precision Agriculture (PA). Crop health monitoring plays a vital role in ensuring superior quality of yields.

The popularity of precision agriculture attracted many researchers and engineers (ML/DL and computer vision) to develop robust models capable of providing precise identification, detection, classification, and localization of essential crop parts to retrieve vital growth features. The earlier implementations of deep learning techniques in the agricultural domain focused on solving classification problems related to crops. These encompassed disease classification [1], pest classification [2], detection [3], grading, and classification of fruits based on maturity levels, such as coffee beans [4], as well as beehive monitoring [5].

The introduction of modern computer vision algorithms provided the "vision" to computers to classify agricultural commodities and pinpoint their locations in the image, either with the bounding boxes or a polygon around the object [6].

Though there are numerous implementations of detecting fruits and vegetables for developing real-time systems enabling the automation of fruit picking during harvesting, there is always a need for pixel-level accuracy in detecting and identifying fruits and vegetables under study.

Prior to the popularity of machine learning algorithms, the identification and categorization of fruits and vegetables relied heavily on image-processing techniques. The traditional or classical segmentation techniques include three widely used methods: region-based, threshold-based, and edge-based. These techniques were prevalent and were mainly used in detection systems to identify, detect and classify various crops under study. One method by [7] includes two ways based on edge and colour segmentation in the case of orange fruit. The colour-based segmentation method obtained an accuracy of 85% in detecting the oranges. An automatic threshold-based method [8] for detecting damages caused by insects on soybean used hyperspectral images to obtain an accuracy of insect-damaged samples of 91.7%. Another

* Corresponding author.
E-mail address: dharwadkarn@cuk.ac.in (N.V. Dharwadkar).

https://doi.org/10.1016/j.atech.2023.100386
Received 30 September 2023; Received in revised form 2 December 2023; Accepted 15 December 2023
Available online 16 December 2023

Fig. 1. Sample raw dataset images consisting of green gram pods collected from the field.

threshold-based method [9] uses the OTSU method for depth-based segmentation and recognition of clustered tomatoes. The developed method achieved a tomato recognition accuracy of 87.9%. Another popular traditional image processing technique widely used is based on edge detection. Modified Canny edge detection for the detection and classification of fruits (Apples and Oranges) by [10] by extracting features corresponding to the colour and the shape. Proposed comparative analysis of various edge detection algorithms such as Prewitt, Laplacian, and Sobel for detecting rice diseases and pests [11].

Further, the authors developed two CNN-based models to show how deep learning algorithms can outperform traditional image processing techniques for image segmentation. Soon the researchers started to realize the need for segmentation techniques capable of providing better object segmentation while requiring low computational time. A watershed-based model [12] for segmenting cotton leaves shows an enhancement in the computational speed of segmentation, with the success rate of segmentation reaching a level of 97% when compared to the manually extracted leaf area. The clustering algorithms used for the segmentation purpose have the edge over the other techniques in providing good segmentation with the advantage of being quick in obtaining the segmentation results. Two prevalent cluster-based segmentation unsupervised methods developed include K-means and Fuzzy C-means clustering. An adaptive K- means clustering algorithm [13] for segmenting diseased tomato leaves was able to achieve higher precision and efficiency when compared to the other techniques, such as traditional K-means, DBSCAN, and mean-shift techniques. A cognitive fuzzy C-means (CFCM) algorithm [14] can precisely identify the diseased and normal parts of rice crops and apple trees. Various algorithms proposed in the literature utilize CNN backbones for monitoring crops, such as the early detection of crop diseases [15]. The best-known architecture widely used for providing instance segmentation in agricultural and other domains includes the Mask R-CNN [16,17] which effectively addresses the issues arising from overlapping objects.

When considering real-time detection and identification of fruits and vegetables, the Yolo series detectors and the Single Shot Multi-Box Detectors (SSDs) are favoured over Faster R-CNN. The study by [18] demonstrates the implementation of real-time apple fruit detection using an enhanced Yolo V3-based model. Similarly, the research conducted in [19] introduces an optimized improved SSD detector for detecting strawberries, which operates on Raspberry Pi.

Green gram is one of the most widely consumed agricultural commodities in the pulses category. Besides high nutritional value, green grams are known for their soil and nitrogen-enriching characteristics. Also, the studies show that the farmers are getting good returns when using new technologies for green gram cultivation. New technologies such as deep learning and computer vision have shown considerable promise for improving agricultural products' quality. In the literature, we could not find any research contributions purely related to the pixel-level detection and instance segmentation of green gram pods. The possible reason might be the challenge posed by the background (the pod's colour matches the background colour), which makes pod detection a complicated task. Moreover, the small size of the pods also poses a challenge for object detection. This research aims at creating a custom polygon annotated image dataset consisting of green gram pods and developing a detectron2-based framework using a rendering approach (PointRend) that modifies the state-of-the-art Mask R-CNN object detection algorithm to yield higher accuracy (mAP) and improve the detection quality of instance segmentation in the case of green gram pods. The primary contributions of this research include:

- The study introduced a first-of-its-kind dataset consisting of polygon annotated images of green gram pods that can identify, detect, and localize green pods with improved instance segmentation capabilities under complex backgrounds.
- The research led to the development of a detectron2-based framework that uses deep learning baseline models and state-of-the-art computer vision libraries to enhance the quality of instance segmentation.
- The developed approach not only delivers enhanced segmentation precision and accuracy 95 but also improves the quality of the segmentation mask, enabling the capture of finer details 96 and more defined green gram pod boundaries. (Segmentation + improvement in segmentation quality)

Table 1
Dataset details indicating contents of training and validation sets.

SL. No.	Dataset Split	Number of Images	Number of Instances	Image Resolution
1.	Training	1215	3720	1024×1024
2.	Validation	270	1512	

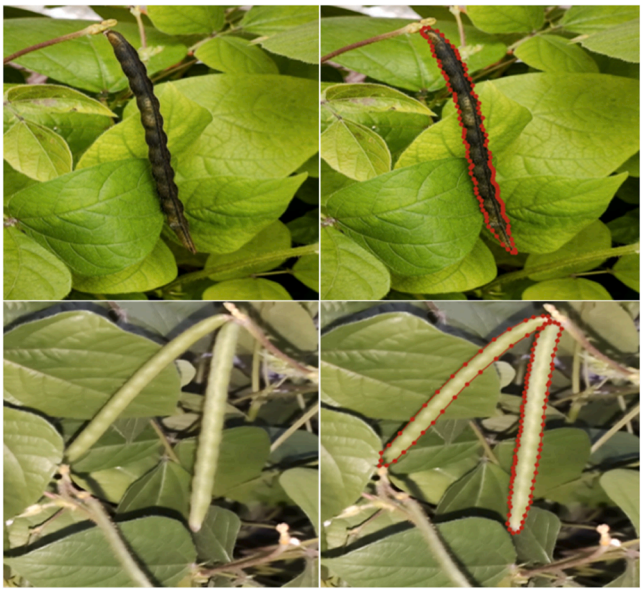

Fig. 2. Green gram sample images from dataset before annotation (left) and after annotation (right).

The rest of the article is organized into four sections. Section 2 details the materials and methods adopted in the study, highlighting dataset details, annotation details, use of augmentations, experimental setup, evaluation metrics, and model architectures. Section 3 provides a detailed tabulation of the training and validation process results, along with the model's predictions on unseen data. Section 4 presents a brief discussion using a comparative analysis of the results and key findings of the experimentation, referring to a similar implementation from the literature. Section 5 concludes the article by summarizing key findings and insights while also discussing potential scope for future research.

2. Materials and methods

2.1. Dataset details

A local field near the Vijayapur district of Karnataka provided the required images for the experimentation. Using a smartphone camera as an imaging device, the raw pictures collected encompass various times of the day (morning, afternoon, evening, and even night). Additionally, the datasets include images representing occluded pods, ensuring high generalizability in various deployment scenarios. In a pre-processing step, rescaling images to a resolution of 1024×1024 provides a balance between detection capability and inference speeds. Several sample images within the dataset displayed in Fig. 1 feature green gram pods set against complex backgrounds (same as the green gram pods), making the detection and localization of pods very challenging.

Table 1 portrays the details of the dataset used for the research in developing an instance segmentation model for detecting green gram pods on the field. The dataset consists of 1215 training images with 3720 training instances representing the green grams. Similarly, the validation set includes 270 images representing 1512 instances of green gram. In the percentage split of training and validation image sets, the ratios are 82% and 18%, respectively.

2.2. Annotation details

To collect the raw images of green gram pods for the dataset, we visited a nearby local farm in the Vijayapur district of Karnataka. A smartphone with a 48 Megapixel camera as an imaging device provided the required images, ensuring the model's generalizability when deployed at the edge mobile devices. The bare pictures had an image dimension of 1846×4000 (portrait) or 4000×1846 (landscape). The images were pre-processed and rescaled to a resolution of 1024×1024 before the training. LabelMe open annotation tool provided the required polygon annotations for the dataset in common objects in context (COCO) JSON format. Fig. 2 represents the sample images of the dataset (before and after annotations).

The pictures in the figure on the left have no annotations (before annotation), while the images on the right show the polygon annotated images (shown in the red-dotted boundary around the green gram pods after annotation).

2.3. Image augmentations

The unique dataset mapper of Detectron2 uses modifications that provide augmentations for expanding the dataset size, preventing the model from overfitting the training data. The augmentation techniques used correspond to resizing (800×600), random brightness (0.8 to 1.8), random contrast (0.6 to 1.3), random saturation (0.8 to 1.4), random rotation (90°), random lighting (0.7), and random vertical flip (with a probability of 0.4). Apart from expanding the dataset, augmentations ensure that the models generalize well when deployed in the production environments.

2.4. Experimental setup

The experimental setup consists of a Detectron2 [20] modular and extensible object detection library that uses the PyTorch [21] framework to provide various computer vision functionalities that include object detection, key point detection, and various types of image segmentation (Semantic, Instance, and Panoptic). The Google Colaboratory Pro-version [22] rendered the required computational resources for model training, including GPU (Tesla P100-PCIE-16GB). The system used for the model development consists of a MacBook Air running on Mojave macOS with Python 3.7 as a programming language. The other essential libraries used include cv2, Pillow, and NumPy. The hyperparameters selected for the model training includes number of epochs (2000), batch size (2, 4, 8 images/batch), learning rate ($1e^{-3}$, $1e^{-2}$), data augmentation techniques (random brightness, random contrast, random lighting, random saturation, random rotation, and random vertical flip).

2.5. Performance evaluation metrics

The model's performance evaluation uses metrics consisting of mean average precision (mAP), average recall (AR), and F1-score. Before evaluating the parameters, it is necessary to understand a key parameter known as intersection-over-union (IoU), invariably used for the detection and localization-based models. The IoU ranges from 0 to 1 and is widely used to provide the measure of overlap between the ground truth bounding box and the predicted bounding box. The higher the value, the better the detection capability of the model. If the overlap is higher than the set threshold of IoU, the detection is considered valid. If not, it results in invalid detection. The parameter evaluation includes calculating the true positive (TP), true negative (TN), false positive (FP), and false negative (FN). The metric that uses TPs and FPs is the precision, given by the ratio of the model's TPs to the sum of TPs and FPs, represented by Eq. (1).

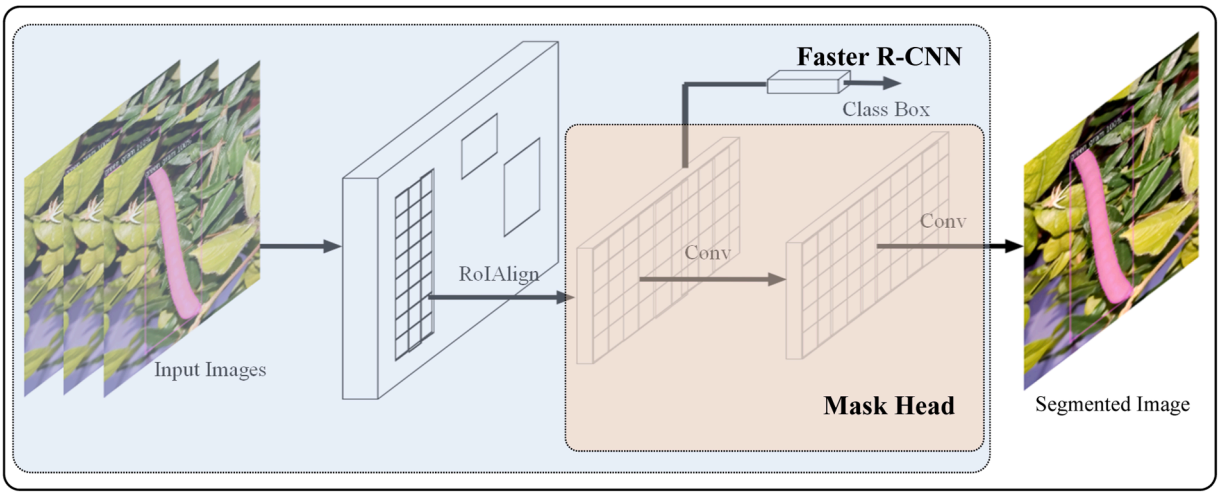

Fig. 3. Detectron2's Mask R-CNN architecture for instance segmentation.

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

The ratio of TPs to the sum of TPs and FNs defines the recall of the model, as indicated in Eq. (2).

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

The F1-score depends on the values of precision and recall and is indicated by Eq. (3).

$$F1 - score = \frac{2\ \times (P * R)}{P + R} \tag{3}$$

The F1-score is a metric generally chosen as a trade-off between precision and recall.

The Average Precision (AP) metric, represented by the area under the precision-recall curve, is obtained by Eq. (4).

$$AP\ = \int_0^1 P(R)\mathrm{d}R \tag{4}$$

The mean average precision is a typical metric that evaluates the object detection model's accuracy. It is determined using average precision (AP) for all the classes and then averaged over *n,* as indicated in Eq. (5).

$$mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i \tag{5}$$

Here, *n* represents dataset classes.

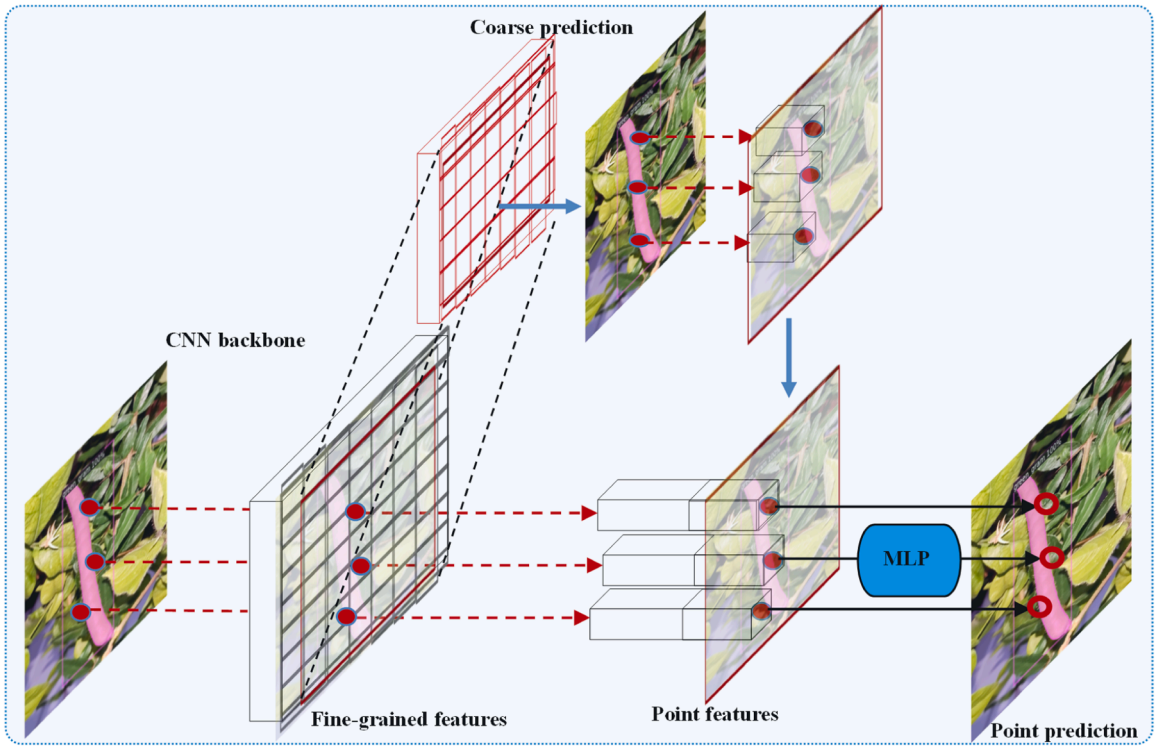

Fig. 4. PointRend architecture for Instance segmentation indicating point proposal and selection for improved class boundaries.

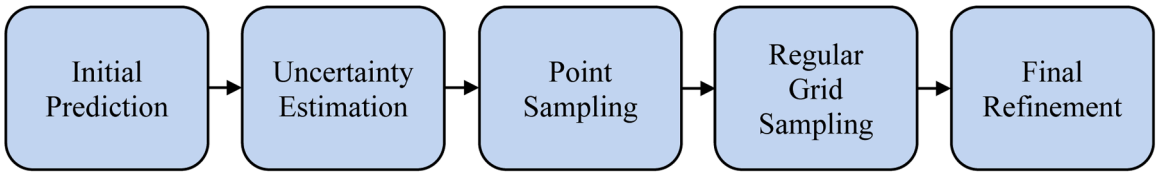

Fig. 5. Point selection strategy adopted in PointRend for improving quality of segmentation mask.

2.6. Model architectures

The PointRend [23] architecture offers flexibility in implementing existing detection algorithms to provide semantic or instance segmentation. One widely used instance segmentation model is the Mask R-CNN, popularly known as a two-stage detector. The Mask R-CNN built over a faster R-CNN includes a mask head for the prediction of object mask in parallel to the standard bounding box prediction. Fig. 3 shows the typical architecture of Mask R-CNN used for providing instance segmentation [24].

The RoIPool layer, used by faster R-CNN, offers coarse spatial quantization during feature extraction. As a result, these improperly aligned features cannot provide a pixel-to-pixel mapping between inputs and outputs. In Mask R-CNN, the RoIAlign layer takes the place of the RoIPool layer to ensure accurate feature extraction and correct alignment. Parallel to the bounding box prediction module, a mask head follows the RoIAlign with two convolutional layers.

The predicted bounding box with class probability and a predicted label boundary around the green gram pods makes up the Mask R-CNN model's final output. In order to obtain good-quality segmentation of the green gram pods, the developed framework shown in Fig. 4 builds on the Mask R-CNN to enable instance segmentation by improving the class boundaries. The standard mask head, consisting of three FC layers (FC1, FC2, and FC3), is replaced by a point mask head and a coarse head, consisting of two FC layers (FC1 and FC2). The model uses coarse features (generated from backbone CNN), which serve as the inputs to the PointRend model resulting in the generation of the fine features with some uncertain points (N) by using the point selection technique.

The smooth boundaries in PointRend are the results of the bilinear upsampling process that is applied to increase the resolution of the prediction boundaries relative to the small number of points at the boundary of the class (green gram pods).

2.6.1. Point selection strategy

The segmentation mask refinement is accomplished through several steps. First, neural networks predict the coarse segmentation mask around each green gram pod in the image. Subsequently, the uncertainty linked to these segmentation masks is estimated using pixel-wise uncertainty measures, such as entropy or confidence scores. Higher uncertainty values indicate ambiguity in the initially predicted masks. The next step is the point sampling, where the points corresponding to the ambiguous region or the region with high uncertainty are chosen for further refinement. These regions are usually the edges of pods or region indicating an abrupt change in the image. Subsequently, regular grid sampling process is undertaken to select a finite number of segmentation points that necessitate refinement, particularly focusing on the boundaries or edges of the pods. These chosen points then undergo a refinement process, resulting in the creation of sharp and well-defined segmentation masks around the pods. This procedure significantly enhances the overall segmentation quality of the pods. The point selection steps are depicted in Fig. 5.

2.6.2. Loss function

The segmentation loss (SL), the point-wise loss (PL) and the total loss are the three loss components that are included in training process to ensure effective model training. The total loss is the weighted sum of SL and PL.

The segmentation loss is a pixel-level loss that measures the pixel-wise inconsistency between the predicted probability distribution and the ground truth labels, which represent each pixel in the image. The commonly used loss function for segmentation tasks is the cross-entropy (CE) loss, calculated by Eq. (6) and represented as:

$$L_{seg} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\cdot\log\left(p_{i,c}\right) \tag{6}$$

Where:

- $p_{i,c}$ is the predicted probability of pixel i belonging to class c
- $y_{i,c}$ is ground truth label for pixel i and class c
- N is the total number of pixels in the image
- C is the number of classes (in our case C=1)

The point-wise loss represents the effectiveness in further refining the mask at selected points within the instance mask. Lower values of this loss ensure more accurate object boundaries, highlighting finer details at the selected points.

The point-wise loss for each selected point m from instance mask is defined using a loss function such as Smooth L1, calculated by Eq. (7) and represented as:

$$L_{point} = \frac{1}{|M|}\sum_{m\in M}\text{SmoothL1}(\hat{y}_m, y_m) \tag{7}$$

Where:

- $|M|$ is the total number of selected points.
- $\hat{y}_m$ is the refined prediction at point m and
- y_m is the ground truth label at point m.

The total loss is the weighted combination of SL and PL, calculated by Eq. (8) and given as:

$$L_{total} = \lambda\cdot L_{seg} + (1-\lambda)\cdot L_{point} \tag{8}$$

Where:

- λ represents the hyperparameter that is responsible for providing control between the SL and PL based on the requirements.
- L_{seg} is the segmentation loss and
- L_{point} is the point loss

Further investigation reveals that the loss parameters associated with PointRend model consists of the following losses in:

(1). *Region of interest (RoI) head*- It consists of RoI classification loss, represented as (L_{cls}^{ROI}) and RoI localisation loss represented as (L_{loc}^{ROI}). The L_{cls}^{ROI} loss provides the measure of model's ability to correctly label the predicted bounding box with the associated or the correct class, while L_{loc}^{ROI} helps to identify how close the predicted bounding boxes are to the true location of the object.

(2). *Region proposal network (RPN)*- This loss consists of RPN classification loss (L_{cls}^{RPN}), and RPN localisation loss (L_{loc}^{RPN}). The L_{cls}^{RPN}

Table 2
Training results for Mask R-CNN model (bounding box) in terms of (mAP) (best result are in bold face).

SL. No.	Baselines	Mean Average Precision (mAP) in%					Training Time (hh: mm: ss)
		mAP	AP_{50}	AP_{75}	AP_M	AP_L	
1.	**X101-FPN**	**52.25**	**67.73**	**59.51**	**45.82**	**78.34**	00:23:30
2.	R101-FPN	50.96	65.57	58.64	42.81	80.50	00:12:05
3.	R50 -FPN	48.49	65.21	55.37	39.76	78.31	00:10:03

provides the measure of how capable the RPN is at identifying the anchor boxes as a foreground or background. In the RPN, L_{loc}^{RPN} is indicative of the loss represented as localisation of the predicted regions.

(3). *Mask head*- The mask loss in mask head is represented as (L_{mask}) and it denotes the correctness of the predicted binary masks.

(4). *Point mask head*- The point loss in the point mask is represented by (L_{Point}) and it demonstrate the model's proficiency in predicting uncertain points within the mask, using cross-entropy loss.

The total loss, denoted as (L_{Total}), is the weighted summation of all these individual losses and is logged during each iteration. Eq. (9) displays the total loss associated with the PointRend model. In contrast, while the losses for Mask R-CNN remain the same as previously mentioned, it lacks a specific loss attributed to the point head.

$$\boldsymbol{L_{Total}} = L_{cls}^{ROI} + L_{loc}^{ROI} + L_{cls}^{RPN} + L_{loc}^{RPN} + L_{mask} + L_{point} \tag{9}$$

3. Results

3.1. Model training and evaluation

The training and validation outcomes were evaluated for both the Mask R-CNN model consisting of the standard head, and for the PointRend model equipped with both a point head and a standard head. The detailed visual comparison indicates the capability of the PointRend model to provide the best quality (smooth and sharp) boundaries across the detected green gram pods. Developing an instance segmentation model, particularly in the case of green gram, poses a challenge owing to the similarity between the background and the object of interest (green gram pods). The proposed model's output assessment includes pixel-level detection of green gram pods with the bounding box and corresponding pixel-level segmentation.

3.1.1. Evaluation of mean average precision (mAP)

The PointRend model trained to detect and provide instance segmentation on the custom dataset uses Google Colaboratory. The experimentation involves training two state-of-the-art segmentation models. The first model is Mask R-CNN+FPN with three baselines (Mask R-CNN with ResNet101(R101-FPN) backbone, Mask R-CNN with ResNeXt101 (X101-FPN) backbone, and Mask R-CNN with ResNet50 (R50-FPN) backbone, and PointRend+FPN with the same baselines, including the point head for the fine-grained prediction for the instance segmentation. The tabulated results further provided the analysis of the models. The training results represent the model metrics consisting of mean average precision (mAP) with IoU threshold values ranging between 0.5 to 0.95, varied in steps of 0.05. The value indicated by AP_{50} corresponds to the average precision with IoU=0.5. Similarly, AP_{75} has IoU=0.75. The other parameters (AP_M and AP_L) indicate the mean average precision values obtained for medium-size instances with the area (32^2 < area < 96^2) and large-size instances with the area (area > 96^2), respectively. The instance segmentation model's evaluation comprises the object's localization and the predicted boundary around the object. The evaluation includes two results, one representing the bounding box prediction and the other segmentation. Table 2 indicates the tabulated mean average precision values obtained for various baselines. For Mask R-CNN training and evaluation for mean average precision (mAP), the X101-FPN backbone outperformed the other two backbones (R101-FPN and R50 -FPN) with a mAP of 52.25%, while R101-FPN and R50-FPN could manage to obtain a mAP of 50.96% and 48.49%, respectively. When considering the training time for 2000 epochs, the X101-FPN took around 23.30 min, which is higher than the time taken for the training of R101-FPN (12.05 min) and R50-FPN (10.03 min). Similarly, Table 3 shows the mean average precision (mAP) evaluated for segmentation after training the Mask R-CNN model for various baselines. The X101-FPN obtained the highest mAP of 39.48%, while R101-FPN and R50-FPN could manage mAP values of 36.84% and 33.07%, respectively. Also, the table represents the other values corresponding to AP_{50}, AP_{75}, AP_M, and AP_L.

Table 3
Training results for Mask R-CNN model (segmentation) in terms of (mAP) (best result are in bold face).

SL. No.	Baselines	Mean Average Precision (mAP) in%					Training Time (hh: mm: ss)
		mAP	AP_{50}	AP_{75}	AP_M	AP_L	
1.	**X101-FPN**	**39.48**	**63.70**	**46.48**	**26.79**	**67.67**	00:23:30
2.	R101-FPN	36.84	62.79	41.90	24.01	64.20	00:12:05
3.	R50 -FPN	33.07	58.02	37.21	18.93	61.76	00:10:03

Table 4
Training results for PointRend model (bounding box) in terms of (mAP) (best result are in bold face).

SL. No.	Baselines	Mean Average Precision (mAP) in%					Training Time (hh: mm: ss)
		mAP	AP_{50}	AP_{75}	AP_M	AP_L	
1.	**X101-FPN**	**53.43**	**68.52**	**61.80**	**45.04**	**80.40**	00:42:54
2.	R101-FPN	51.79	67.30	58.41	43.74	80.89	00:20:46
3.	R50 -FPN	48.35	65.17	56.77	41.14	76.75	00:15:18

Table 5
Training results for PointRend model (segmentation) in terms of (mAP) (best result are in bold face).

SL. No.	Baselines	Mean Average Precision (mAP) in%					Training Time (hh: mm: ss)
		mAP	AP_{50}	AP_{75}	AP_M	AP_L	
1.	**X101-FPN**	**46.15**	**67.49**	**54.63**	**33.13**	**74.03**	00:42:54
2.	R101-FPN	44.45	65.81	53.41	29.68	74.33	00:20:46
3.	R50 -FPN	42.89	66.61	50.04	26.44	72.64	00:15:18

Tables 4 and 5 show the bounding box and segmentation results obtained by the PointRend model, respectively, for mAP.

Fig. 6 shows the plots corresponding to the model evaluation for mAP. In Fig. 6(a) and (b), the results obtained for mAP for the Mask R-CNN model represent the bounding box and segmentation, respectively, while Fig. 6(c) and (d) indicate the results obtained for the PointRend model in terms of mAP.

3.1.2. Evaluation of average recall (AR)

The average recall metrics indicate how the trained model can deal with the false negative (FN) corresponding to the detections made by the model. We consider the AR value with a maximum detection of 10 per image for evaluation ($AR_{max=10}$) both in the bounding box and

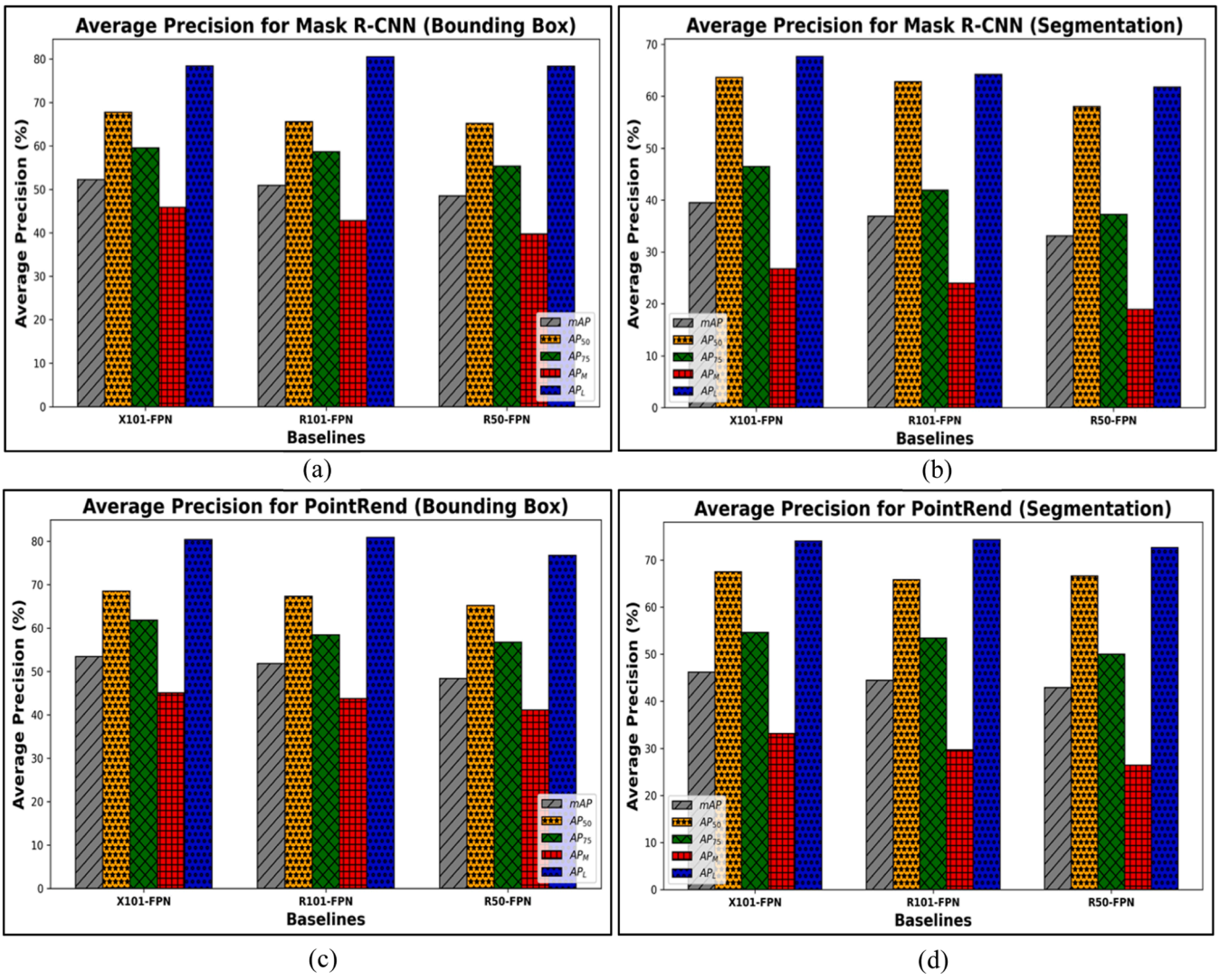

Fig. 6. Plots showing variation in mean average precision (mAP) for (a) Mask R-CNN (Bounding box) (b) Mask R-CNN (Segmentation) (c) PointRend (Bounding box) and (d) PointRend (Segmentation).

Table 6
Training results for Mask R-CNN model (bounding box) in terms of (AR) (best result are in bold face).

SL. No.	Baselines	Average Recall (AR) in%				Training Time (hh: mm: ss)
		$AR_{max=10}$	$AR_{max=100}$	AR_M	AR_L	
1.	**X101-FPN**	**55.80**	**61.10**	**54.70**	**84.30**	00:23:30
2.	R101-FPN	55.00	59.10	51.50	84.40	00:12:05
3.	R50 -FPN	53.30	61.20	55.10	84.10	00:10:03

Table 7
Training results for Mask R-CNN model (segmentation) in terms of (AR) (best result are in bold face).

SL. No.	Baselines	Average Recall (AR) in%				Training Time (hh: mm: ss)
		$AR_{max=10}$	$AR_{max=100}$	AR_M	AR_L	
1.	**X101-FPN**	**43.10**	**45.60**	**36.30**	**72.30**	00:23:30
2.	R101-FPN	42.20	44.40	35.10	70.20	00:12:05
3.	R50 -FPN	37.60	40.90	31.30	67.70	00:10:03

Table 8
Training results for PointRend model (bounding box) in terms of (AR) (best result are in bold face).

SL. No.	Baselines	Average Recall (AR) in%				Training Time (hh: mm: ss)
		$AR_{max=10}$	$AR_{max=100}$	AR_M	AR_L	
1.	X101-FPN	57.00	62.00	55.20	85.55	00:42:54
2.	**R101-FPN**	**57.00**	**62.50**	**55.30**	**86.20**	00:20:46
3.	R50 -FPN	52.30	61.60	55.40	82.70	00:15:18

segmentation case.

The average recall (AR) values obtained for instance segmentation corresponding to the Mask R-CNN model for bounding box and segmentation, are depicted in Tables 6 and 7, respectively. Table 6 clearly shows that, compared to the other two baselines with 55.00% and 53.30% AR values, the X101-FPN baseline obtained the highest AR of 55.80%. In Table 7, the tabulated AR values show that the X101-FPN baseline outperformed the R101-FPN (42.20%) and R50-FPN (37.60%) by reaching the highest value of 43.10%. The table also shows the values obtained for $AR_{max=100}$, AR_M, and AR_L.

Similarly, Table 8 shows the results corresponding to AR (bounding

Table 9
Training results for PointRend model (segmentation) in terms of (AR) (best result are in bold face).

SL. No.	Baselines	Average Recall (AR) in%				Training Time (hh: mm: ss)
		$AR_{max=10}$	$AR_{max=100}$	AR_M	AR_L	
1.	**X101-FPN**	**49.40**	**54.00**	**47.00**	**77.10**	00:42:54
2.	R101-FPN	49.10	52.20	44.00	77.10	00:20:46
3.	R50 -FPN	48.00	53.90	46.90	76.30	00:15:18

Table 10
Calculation of (F1-score) for Mask R-CNN and PointRend (Bounding Box and Segmentation) using mAP and AR (best results are in bold face).

SL. No.	Model	Bounding Box Evaluation (%)			Segmentation Evaluation (%)		
		mAP	AR	F1-score	mAP	AR	F1-score
1.	Mask R-CNN	52.25	54.70	53.45	39.48	36.30	37.82
2.	**PointRend**	**53.43**	**55.20**	**54.30**	**46.15**	**47.00**	**46.57**

box) for the PointRend model. Considering the AR (Table 8), the X101-FPN and R101-FPN baselines yielded the highest value of 57.00%, while R50-FPN could manage a value of 52.30%, as indicated. The table also includes the other parameters indicated by $AR_{max=100}$, AR_M, and AR_L.

In the segmentation results for AR (Table 9), the X101-FPN baseline outwitted the other baselines with a value of 49.40%, which could manage AR values of 49.10% and 48.00%, corresponding to the R101-FPN and R50-FPN baselines, respectively. The table also tabulates the other values indicated by $AR_{max=100}$, AR_M, and AR_L for PointRend segmentation.

Based on the mAP and AR values, the F1-scores calculated for Mask R-CNN and PointRend models are tabulated in Table 10 for both bounding box and segmentation considering only the best-performing baselines. The results indicate that the PointRend model performs better by including a point head for making point-based predictions that are more accurate when compared to Mask R-CNN with the standard head.

Fig. 7 displays the graphs of the AR values obtained in terms of the bounding box and segmentation for Mask R-CNN and PointRend models. Figs. 7(a) and (b) show the variation in AR (bounding box) for Mask R-CNN and PointRend models, while Fig. 7(c) and (d) represent the AR values attained by Mask R-CNN and PointRend models in terms of segmentation. The graphs show that Mask R-CNN and PointRend models obtained higher AR values for bounding boxes than AR's segmentation results. The PointRend model achieved better results regarding the bounding box and the segmentation evaluation, as depicted by the graphs.

Fig. 8 shows the F1-scores of the two trained models (Mask R-CNN and PointRend) along with mAP and AR. Fig. 8(a) represents the values obtained for the models in the bounding box, while Fig. 8(b) gives segmentation results for the models.

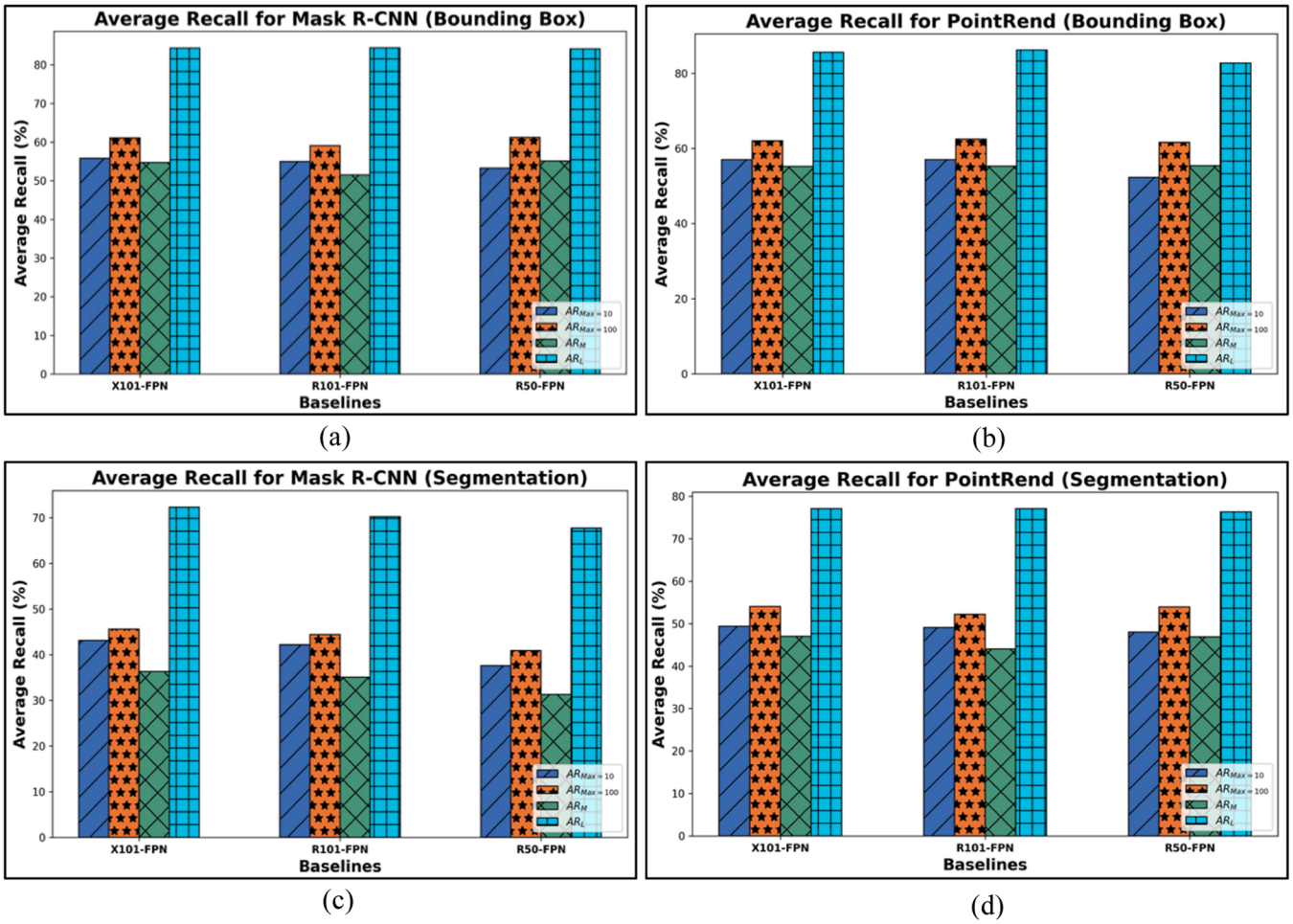

Fig. 7. Plots showing variation in average recall (AR) for (a) Mask R-CNN (Bounding box) (b) PointRend (Bounding box) (c) Mask R-CNN (Segmentation) and (d) PointRend (Segmentation).

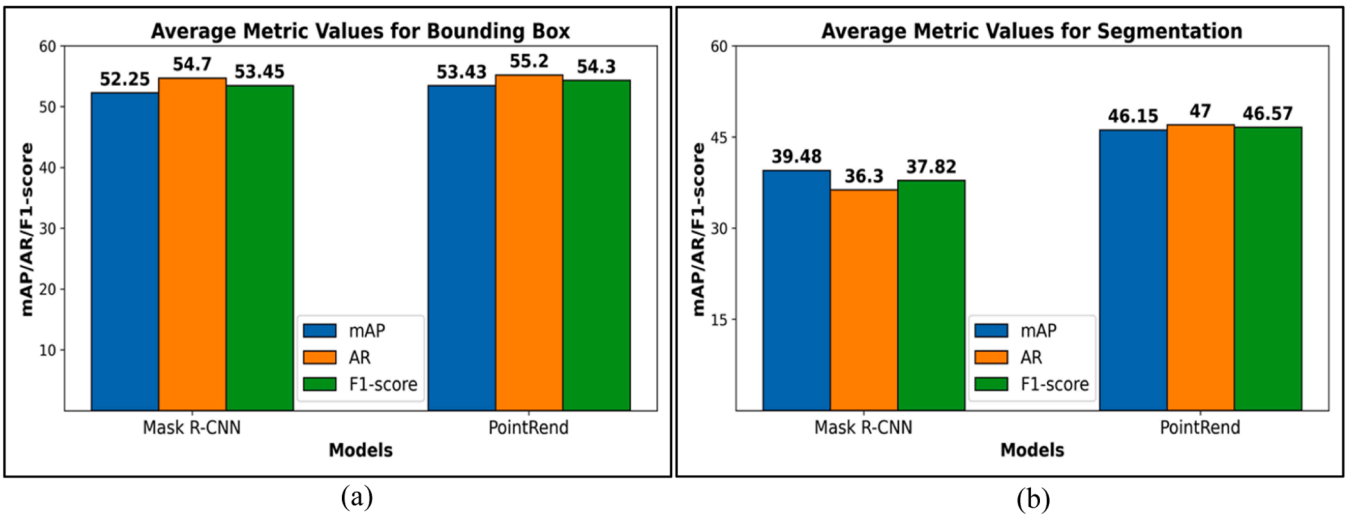

Fig. 8. Variation of mAP, AR, and F1-score for (a) Mask R-CNN and PointRend (Bounding box) and (b) Mask R-CNN and PointRend (Segmentation).

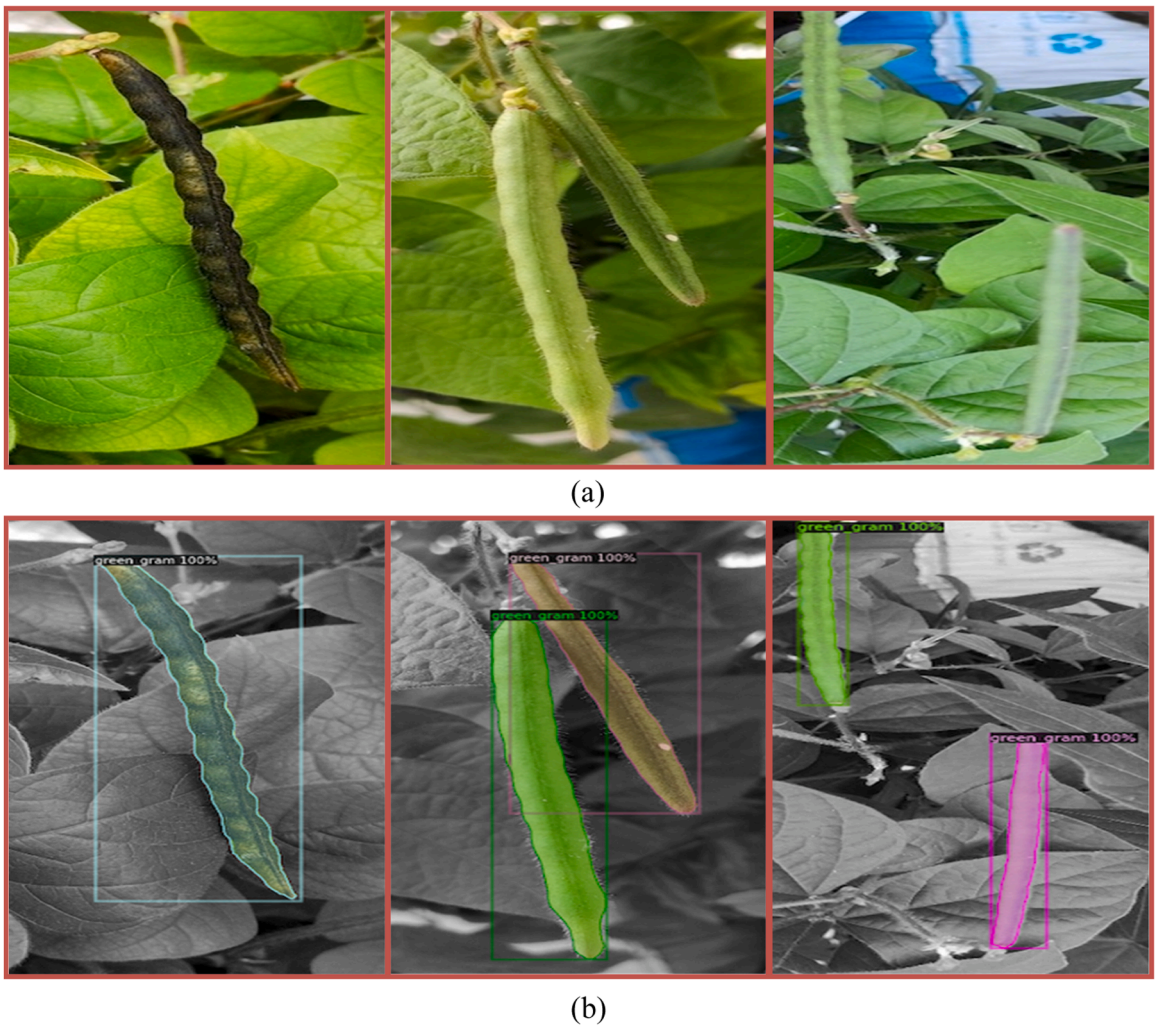

Fig. 9. The model's performance on sample images from the validation set includes: (a) Input images, and (b) Corresponding model-predicted images with bounding boxes and pixel-level class boundaries.

Fig. 9 shows the validation performance of the PointRend-based model. The results suggest the efficacy of the rendering approach used to develop the model to improve the detection capability and the quality of instance segmentation.

The images on the top are the sample images from the validation set, while those on the bottom are the model-predicted output images. The

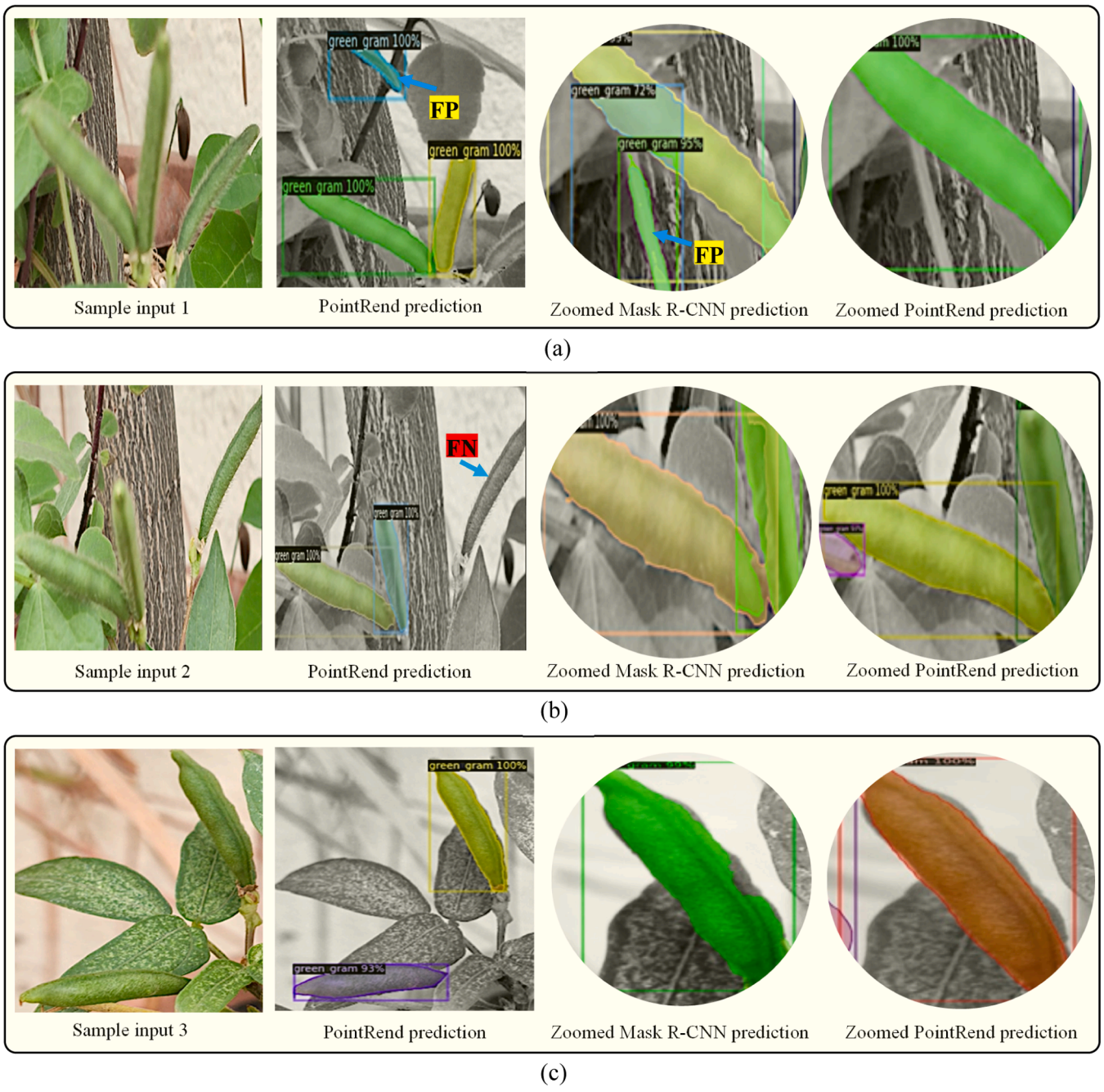

Fig. 10. The visual evaluation of the Mask R-CNN and PointRend trained models on unseen images, the sequence followed is: sample input image-PointRend Prediction-Zoomed Mask R-CNN prediction-Zoomed PointRend Prediction where (A)Sample input 1 (B)Sample input 2 and (C)Sample input 3.

images on the right indicate the model's prediction in terms of the bounding box and the pixel-level segmentation.

Compared to the Mask R-CNN-based model, the model's high confidence level values (with a tight bounding box) and the smooth and sharp edges of the green gram pods show their potential to produce excellent results. Further, to validate the PointRend-based model, the testing was carried out on unseen images to indicate the model's capability to provide improved quality of instance segmentation with fine, smooth, and sharp boundaries across the green gram pods.

Fig. 10 visually represents the results obtained by the model. For simplicity, three input images considered as the unseen sample images (sample images 1, 2, and 3) are depicted in Fig. 10(a), (b), and (c) respectively. The figure clearly shows the predictions by the model that have better quality in terms of well-defined boundaries compared to the predictions made by the Mask R-CNN-based model. By suitably zooming, the predicted images provide fine-grained details on the sample images as indicated. Also, the figure indicates some false positives (FPs) and false negatives (FNs) produced by Mask R-CNN and PointRend models. The overall result indicates that the PointRend model outperforms the Mask R-CNN model in providing a better quality of instance segmentation with smooth and sharp edges around the green gram pods.

3.1.3. Loss function evaluation

Soon after training process, the model's loss functions are logged and evaluated, consisting of various loss parameters as discussed in Section 2.6.2. To streamline comprehension, we have chosen to retain and present solely the loss functions pertinent to the PointRend model. This

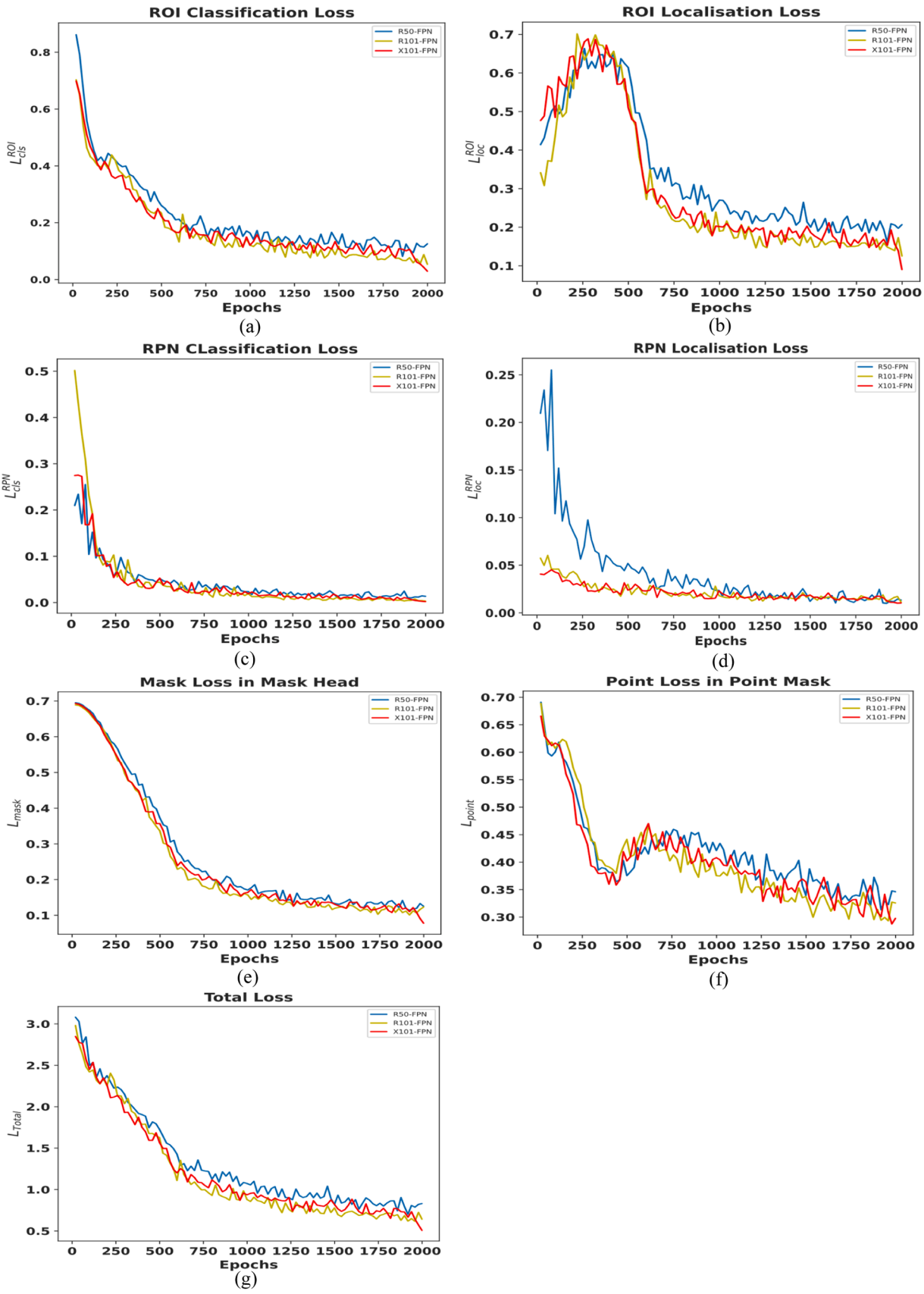

Fig. 11. Plot indicating the loss function evaluation (a) RoI Classification Loss (b) RoI Localisation Loss (c) RPN Classification Loss (d) RPN Localisation Loss (e) Mask Loss (f) Point Loss and (g) Total Loss.

Table 11
Comparison of the proposed work with a similar implementation from the literature (best result are in **bold** face).

SL. No.	Reference	Dataset type	Agricultural commodity under study	Function Rendered	Network	Bounding Box AP_{50}	Segmentation AP_{50}
1.	[26]	Peach Disease Image Dataset (PDID)	Peach disease detection	Segmentation	Cascade R-CNN	0.450	0.243
					Mask Scoring R-CNN	0.301	0.279
					Mask R-CNN	0.294	0.267
2.	[27]	Custom dataset	Cucumber fruit detection	Detection (Bounding Box)	Yolo V5	0.550	–
4.	**Proposed method**	**Custom dataset**	**Green gram pod detection**	**Segmentation+ boundary refinement**	Mask R-CNN	0.677	0.637
					PointRend	**0.685**	**0.675**

evaluation comprises a total of seven distinct losses.

Within the RoI (Region of Interest) proposal network, two essential losses are identified: the RoI classification loss (L_{cls}^{ROI}) and the RoI localization loss (L_{loc}^{ROI}). Correspondingly, the region proposal network (RPN) contributes the RPN classification loss (L_{cls}^{RPN}) and the RPN localization loss (L_{loc}^{RPN}) to the overall evaluation. Furthermore, additional losses comprise the mask loss within the mask head (L_{mask}) and the point loss in the point mask (L_{point}). The weighted summation of these losses yields the total loss (L_{Total}).

The assessment of these distinct losses plays an essential role in comprehensively evaluating the model's performance throughout the training phase. It aids significantly in minimizing errors associated with classification, localization, and point refinement.

Moreover, these losses decide the optimal set of hyperparameters, effectively enhancing the model's performance when fine-tuned. The graphical representation of these losses is depicted in the plot illustrated in Fig. 11. The evaluation of loss function consists R50-FPN, R101-FPN, and X101-FPN as the base models. Fig. 11(a) indicates the variation of the RoI classification loss as a function of number of epochs or iterations. As indicated, this loss decreases with number of iterations, and the loss attains a final value at the end of 2000th epoch. The final value of RoI classification loss attained by various baselines are: R50-FPN=0.1257, R101-FPN=0.0538, and X101-FPN=0.0292. The performance of PointRend model for RoI classification loss shows better results with X101-FPN (0.0292) baseline when compared to R50-FPN (0.1257), and R101-FPN (0.0538) baselines. Similarly, for RoI localisation loss, the performance achieved by X101-FPN (0.09), is better than R50-FPN (0.1253), and R101-FPN (0.2067), as indicated by Fig. 11(b). Further, the loss corresponding to RPN classification and localisation are shown in Fig. 11(c) and (d), respectively. The mask loss and point loss are depicted by Fig. 11(e) and (f), while the total loss, which the weighted sum of these losses is shown in Fig. 11(g). The close observation reveals that the performance of X101-FPN baseline model outperforms the other basslines.

4. Discussion

The results obtained by the PointRend model during the training and the testing process indicate the superiority of the developed PointRend model in delivering unmatched performance compared to the Mask R-CNN model. As mentioned earlier, this study is the first to demonstrate the enhanced rendering-based instance segmentation considering the case study of green gram pods. The first implementation of the PointRend-based model shows the enhancement of the COCO and Cityscapes dataset providing an average precision of 38.2% for segmentation, with a maximum output image resolution of 224×224. The proposed PointRend-based model obtained higher mAP (46.15%) for an output image dimension of 1024×1024 with improved quality of instance segmentation compared to the segmentation quality obtained by the Mask R-CNN (39.48%). The study conducted by [25] addresses a comparable challenge of detecting Cucumber fruits amid intricate backgrounds resembling the object of interest. Their research focused on enhancing instance segmentation using Mask R-CNN. While their findings showcased remarkable performance in contrast to alternative methods (Yolo V2, Yolo V3, Original Mask R-CNN), the evaluation overlooked improvements in the precision of mask boundaries. A separate study conducted by [26] involved experimentation and comparison of two models, incorporating Mask R-CNN, Mask scoring R-CNN, and Cascade R-CNN with various baselines, aimed at providing instance segmentation for detecting diseases in peach fruits. Additionally, in the research conducted by [27], a method based on Yolo V5 was proposed for detecting cucumber fruits in a greenhouse environment by leveraging colour space features. Table 11 compares the performance obtained by the experimented models (Mask R-CNN and PointRend) with the results obtained by other similar implementations used for segmentation of peach diseases and cucumber fruits. The observation reveals that the proposed PointRend model outperforms the experimented Mask R-CNN and the results obtained by the models developed above from the literature (both in terms of the bounding box and segmentation mean average precision). The proposed PointRend-based implementation obtained a 39.10% improvement in the bounding box AP_{50} and a 40.80% increase in segmentation AP_{50}. Apart from providing improved mean average precision, the PointRend model achieves better results in terms of segmentation quality. The RestNeXt101-FPN baseline implemented in the experimentation of Mask R-CNN and PointRend models yielded the best model performance.

5. Conclusion and future scope

Modern computer vision tasks in agriculture require precise detection and localization of agricultural commodities like field crops, fruit crops, and vegetable crops to automate the process that saves the farmers' labor and time. Thus, the detection algorithms must have pixel-level accuracies to ensure error-free operation. The proposed framework addresses this issue by developing a PointRend-based model on top of Detectron2 instance segmentation to demonstrate the class boundaries' improvement by employing point-based refined prediction. Further, the comparison results between the Mask R-CNN-based implementations (with three baseline models) and the PointRend-based model highlight the effectiveness of the proposed model. According to the results, the PointRend-based model provides higher values of mean average precision (both for bounding box and segmentation) along with crisp and smooth class boundaries around green gram pods under natural environments. Improved results obtained for the sample test images (unseen images) indicate high generalizability from the rendering-based approach. The PointRend-based model outperforms the standard Mask R-CNN-based model in terms of accuracy (2.26% increase in mAP for bounding box and 16.90% mAP improvement for segmentation compared to Mask R-CNN results) and quality of detected boundaries across the green gram pods. The methodology developed in this study is the first of its kind to test the rendering-based instance segmentation framework by taking the case study of green gram pods (small object detection) in the field environment, which is challenging owing to the similarity between the pods and the background (leaves). The accuracy

in localizing the objects (pods) offered by the proposed method is higher than the experimented Mask RCNN, original Mask R-CNN and Mask scoring R-CNN with improved segmentation masks. With precise detection of the pods in the field, a precise yield estimation helps effectively plan and manage post-harvest activities to improve farmers' profit margins with enhanced agricultural sustainability. In future work, we plan to expand the dataset with more field images (including diseased pods) to enhance the functionalities of the developed model with improved robustness.

CRediT authorship contribution statement

Nagaraj V. Dharwadkar: Data curation, Formal analysis, Methodology, Resources, Supervision, Writing – review & editing. **Rajinder-Kumar M. Math:** Formal analysis, Validation, Visualization.

Declaration of Competing Interest

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

Data availability

Data will be made available on request.

References

[1] A. Bhujel, F. Khan, J.K. Basak, M. Jaihuni, T. Sihalath, B.E. Moon, J. Park, H. T. Kim, Detection of gray mold disease and its severity on strawberry using deep learning networks, J. Plant Dis. Prot. (2022), https://doi.org/10.1007/s41348-022-00578-8.

[2] K. Thenmozhi, U. Srinivasulu Reddy, Crop pest classification based on deep convolutional neural network and transfer learning, Comput. Electron. Agric. 164 (2019), 104906, https://doi.org/10.1016/j.compag.2019.104906.

[3] R.K.M. Math, N.V. Dharwadkar, Deep learning and computer vision for leaf miner infestation severity detection on muskmelon (*Cucumis melo*) leaves, Comput. Electr. Eng. 110 (2023), 108843, https://doi.org/10.1016/j.compeleceng.2023.108843.

[4] N. Huang, D. Chou, C. Lee, F. Wu, A. Chuang, Y. Chen, Y. Tsai, Smart agriculture: real-time classification of green coffee beans by using a convolutional neural network, IET Smart Cities 2 (4) (2020) 167–172, https://doi.org/10.1049/iet-smc.2020.0068.

[5] S. Kaplan Berkaya, E. Sora Gunal, S. Gunal, Deep learning-based classification models for beehive monitoring, Ecol. Inform. 64 (2021), 101353, https://doi.org/10.1016/j.ecoinf.2021.101353.

[6] P. Barré, B.C. Stöver, K.F. Müller, V. Steinhage, LeafNet: a computer vision system for automatic plant species identification, Ecol. Inform. 40 (2017) 50–56, https://doi.org/10.1016/j.ecoinf.2017.05.005.

[7] R. Thendral, A. Suhasini, N. Senthil, A comparative analysis of edge and color-based segmentation for orange fruit recognition, in: Proceedings of the International Conference on Communication and Signal Processing, 2014, pp. 463–466, https://doi.org/10.1109/ICCSP.2014.6949884.

[8] Y. Ma, M. Huang, B. Yang, Q. Zhu, Automatic threshold method and optimal wavelength selection for insect-damaged vegetable soybean detection using hyperspectral images, Comput. Electron. Agric. 106 (2014) 102–110, https://doi.org/10.1016/j.compag.2014.05.014.

[9] R. Xiang, H. Jiang, Y. Ying, Recognition of clustered tomatoes based on binocular stereo vision, Comput. Electron. Agric. 106 (2014) 75–90, https://doi.org/10.1016/j.compag.2014.05.006.

[10] M.K. Monir Rabby, B. Chowdhury, J.H. Kim, A modified canny edge detection algorithm for fruit detection & classification, in: Proceedings of the 10th International Conference on Electrical and Computer Engineering (ICECE), 2018, pp. 237–240, https://doi.org/10.1109/ICECE.2018.8636811.

[11] P. Muruganandam, V. Tandon, B. Baranidharan, Rice crop diseases and pest detection using edge detection techniques and convolution neural network, J.C. Bansal, A. Engelbrecht, & P.K. Shukla (Eds.). Computer Vision and Robotics, Springer, Singapore, 2022, pp. 49–64, https://doi.org/10.1007/978-981-16-8225-4_5.

[12] C. Niu, H. Li, Y. Niu, Z. Zhou, Y. Bu, W. Zheng, Segmentation of cotton leaves based on improved watershed algorithm, In D. Li & Z. Li (Eds.), in: Proceedings of the Computer and Computing Technologies in Agriculture IX 478, Springer International Publishing, 2016, pp. 425–436, https://doi.org/10.1007/978-3-319-48357-3_41.

[13] K. Tian, J. Li, J. Zeng, A. Evans, L. Zhang, Segmentation of tomato leaf images based on adaptive clustering number of K-means algorithm, Comput. Electron. Agric. 165 (2019), 104962, https://doi.org/10.1016/j.compag.2019.104962.

[14] S. Sampathkumar, R. Rajeswari, An automated crop and plant disease identification scheme using cognitive fuzzy C-means algorithm, IETE J. Res. (2020) 1–12, https://doi.org/10.1080/03772063.2020.1780163.

[15] R.M. Math, N.V. Dharwadkar, Early detection and identification of grape diseases using convolutional neural networks, J. Plant Dis. Prot. 129 (2022) 521–532, https://doi.org/10.1007/s41348-022-00589-5.

[16] U. Afzaal, B. Bhattarai, Y.R. Pandeya, J. Lee, An instance segmentation model for strawberry diseases based on mask R-CNN, Sensors 21 (19) (2021) 6565, https://doi.org/10.3390/s21196565.

[17] D. Wang, D. He, Fusion of mask RCNN and attention mechanism for instance segmentation of apples under complex background, Comput. Electron. Agric. 196 (2022), 106864, https://doi.org/10.1016/j.compag.2022.106864.

[18] Y. Tian, G. Yang, Z. Wang, H. Wang, E. Li, Z. Liang, Apple detection during different growth stages in orchards using the improved YOLO-V3 model, Comput. Electron. Agric. 157 (2019) 417–426, https://doi.org/10.1016/j.compag.2019.01.012.

[19] N. Lamb, M.C. Chuah, A strawberry detection system using convolutional neural networks, in: Proceedings of the IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 2515–2520, https://doi.org/10.1109/BigData.2018.8622466.

[20] Y. Wu, A. Kirillov, F. Massa, W. Lo. https://github.com/facebookresearch/detectron2.

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, S. Chintala, PyTorch: an imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems, 32, Curran Associates, Inc, 2019, pp. 8024–8035. Retrieved from, http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[22] E. Bisong, Google colaboratory, E. Bisong. Building Machine Learning and Deep Learning Models on Google Cloud Platform, Apress, 2019, pp. 59–64, https://doi.org/10.1007/978-1-4842-4470-8_7.

[23] A. Kirillov, Y. Wu, K. He, R. Girshick, PointRend: Image Segmentation as Rendering. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019, pp. 9796–9805.http://arxiv.org/abs/1912.08193.

[24] K. Gkioxari, G. Dollár, P. Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 2017, pp. 2961–2969.

[25] X. Liu, D. Zhao, W. Jia, W. Ji, C. Ruan, Y. Sun, Cucumber fruits detection in greenhouses based on instance segmentation, IEEE Access 7 (2019) 139635–139642, https://doi.org/10.1109/ACCESS.2019.2942144.

[26] N. Yao, F. Ni, M. Wu, H. Wang, G. Li, W.K. Sung, Deep learning-based segmentation of peach diseases using convolutional neural network, Front. Plant Sci. 13 (2022), https://doi.org/10.3389/fpls.2022.876357.

[27] N. Wang, T. Qian, J. Yang, L. Li, Y. Zhang, X. Zheng, Y. Xu, H. Zhao, J. Zhao, An enhanced YOLOv5 model for greenhouse cucumber fruit recognition based on color space features, Agriculture 12 (10) (2022) 1556, https://doi.org/10.3390/agriculture12101556.