Detection of Arecanut Bunch Yield Count using Detectron2

Anitha A C, Shrinivasa Naika C L, Krishna A N, Hanush Gowrav Gowda K S

***Abstract*—Arecanut have an important representation in social, cultural and religious events in India and is widely used in natural and animal health. Proper growth monitoring of arecanut requires precise segmentation of the arecanut bunch, eliminating unnecessary background information. Estimating yield before harvest enables farmers to plan for storage and trade. Segmentation is carried out using U-Net, MRCNN, U2-Net, and Graph Cut Methods. The graph cut method gives better segmentation results involving regional and global information, acquiring minute details and the shape of objects. Segmentation performance gives 85.78% IoU and 93.15% Dice score and is superior, correlating with benchmarks. The segmentation output is the input for the yield count detection model called Detectron2. Yield count is determined for segmented ripe and unripe images using Detectron2. Yield count emulation with 3.23% Mean Absolute Percentage Error (MAPE) for ripe images and 4.1% for unripe images, which is very good compared to the yield count for unsegmented images of 6.6% MAPE.**

***Index Terms*—Arecanut, Segmentation, Graphcut, U-Net, MRCNN, U2-Net, Yield, Detectron2**

I. Introduction

Arecanut is one of the most profitable and economically important crops in India. The profitability of arecanut plantation farmers is solely dependent on the market price of arecanuts, which in turn depends on the quality and maturity level of arecanuts. Arecanut's usual usage is chewing with betel leaves. Arecanut is widely used in homoeopathic and animal health. Experts have mainly suggested using arecanuts to remove tapeworms and other intestinal parasites. Chewing arecanut with betel leaf acts as a natural breath enhancer and a purgative. It also helps in metabolism, is a diuretic, enhances cardiac tissues and controls menses. Arecanut contributes to social, cultural, and religious events, and the fiscal life of human beings in the country. Arecanut contributes to manufacturing toothpaste, soap, tea powder, vita, and wine [1].

Plantations of arecanuts are established on 5,000 to 10,000 hectares of agricultural land each year in India. A total of 7.06 lakh tons of arecanuts are harvested on 4.73 lakh hectares (HA) of land each year, approximately, which is the highest both in terms of the area (47%) and the yield (47%). India yield is at 1.27 tonnes/hectare, which matches the global output. In India, arecanut plantation mainly spreads in Karnataka, Meghalaya, Assam, Tamil Nadu, Kerala and West Bengal. Kerala and Karnataka are the states that produce the most arecanuts in the world [2]. Arecanut will grow in temperatures ranging from a minimum of 14 to 36 degrees Celsius and in regions that receive rainfall of 750 mm to 4,500 mm annually. Its plantation can be found extensively in laterite soils of type red clay. It can also grow in clay loam soils. The arecanut yielded in India is mostly utilised within the country. Approximately 15 to 20 cm thick, the round bark stem of the arecanut plant reaches a height of up to 30 to 40 feet. Approximately 1200 plants in one hectare of land with 4 to 5 bunches of arecanuts per plant make it difficult for monitoring growth, recognition and harvest. Arecanut farming is burdensome and tiring, and requires regular monitoring for better yield.

Production forecasting facilitates the farmers and food industries to plan for harvest, warehouse and trading. Obtaining a reasonably accurate yield estimate before harvest is crucial for appropriate intervention if low yields are estimated [3]. Yield estimation based on expert knowledge of farmers, professionals, or from previous years' yield maps is subjective and often inaccurate. In addition, the yield from historical data might vary from year to year. Also, the manual process of production forecasting is labour-intensive and costly. Vision-based production forecasting became more useful. Vision-based precision agriculture is necessary to assist farmers in monitoring the health, quality, and maturity at the time of harvesting and yield of arecanut. Segmenting arecanut bunches from an input image is an essential component of an automated system for determining the health, quality, maturity and yield of arecanuts. An error-free computerised and vigorous segmentation technique is needed as manual segmentation is difficult, burdensome, subjective and usually leads to fallacy. Divergence in crop colour, silhouette and inter-reflection in the outdoors as the daylight changes makes the segmentation difficult. Segmenting the immature crop bunch is much harder because their colour is commonly green and looks like foliage [4]. A review of different segmentation and yield detection techniques is very much connected to our problem.

The primary focus of this work is the segmentation and determining the yield count of an arecanut bunch. The remaining part of the article is ordered as follows: A survey of the available works connected to segmentation and yield count is presented in section II. The methodology for segmentation and yield count is described in division III. Details of the experimentation are presented in section IV. Results and performance details are presented in division V. The last division summarizes the work done and the future avenues.

Manuscript received Jan 8, 2025; revised Aug 18, 2025.

Anitha A C is an associate professor of Computer Science and Engineering Department, Government SKSJ Technological Institute, Bangalore, Karnataka, India (e-mail: anithakrishna2008@gmail.com).

Shrinivasa Naika C L is a professor of Computer Science and Engineering Department, University B.D.T. College of Engineering, Davanagere, Karnataka, India (e-mail: naika2k6@gmail.com).

Krishna A N is a professor of Computer Science and Engineering Department, SJB Institute of Technology, Bangalore, Karnataka, India (corresponding author phone: +91 9481189899; e-mail: drkrishnaan@gmail.com).

Hanush Gowrav Gowda K S is a postgraduate student of State University of New York, Binghamton, New York, USA (e-mail: hanushgowrav66@gmail.com).

II. Survey

Image segmentation is a crucial stage in any vision-based system and is defined as an intuitive clustering of pixels depending on closeness and proximity [5]. Accurate segmentation is essential for the fruitful derivation of image features and subsequent recognition, and its achievement directly impacts the conduct of the whole vision system. Yield estimation can be done at different stages of crop growth: (1) Blooms, (2) grain/fruit filling, (3) Maturity. Flower density is used to assess the yield. Bloom stage yield estimation is mostly incorrect due to unpredictable environmental conditions and the long duration between bloom and harvest. Also, fruit setting, thinning and fruit drop make the estimation difficult and impact on the accuracy [6]. Many reviews exist for segmentation and yield estimation of apples, mangoes, grapes, tomatoes, and almonds, like crops as compared to arecanut. A survey of existing work on the above crops is directly related to our topic of interest.

A review of techniques for pre-processing, detecting and recognising fruits is presented by Carlos et al. [7]. A review of methods for vineyard yield estimation, management, monitoring for disease detection and evaluation of bunch compactness and maturity is presented by Phooi et al. [8]. Grapes come in two colours: red and white. Red grapes can be easily distinguished from the leaves surrounding them and thus relatively easier to segment as compared to white grapes. Ashfaqur et al. [9] presented an approach to segmenting seasoned grape clusters by primarily finding the gradient contours and then detecting circles using the circular Hough transform. The circles are then distinguished as grapes or scenes by making use of a classifier. Since grapes tend to continue to be in the spatial vicinity of each other, circles with no neighbour within two times the size of their diameter are treated as isolated and are evacuated. The remaining circles are grouped using k-means clustering.

Different pixel-based approaches: Colour, threshold, Mahalanobis distance, Bayesian classifier, Linear colour model, and Histogram-based segmentation have been presented by Davinia Font et al. [10] to identify reddish grapes. Segmentation using thresholding of the H layer yielded a better result for non-occluded reddish grapes. These approaches are simple and fast but incorporate noise. Region-based approaches using edge finding and shape fitting: cotton detection [11], maize tassel segmentation [12], and Rice grains segmentation [13] are proposed to solve the problem of noise incorporation. A very few attempts for arecanut segmentation and yield count includes maximum similarity-based region merging (MSRM) with superior outcomes compared to thresholding, clustering, and Watershed [14], morphological operations and active contours [15], YCgCr color model and erosion and closing [16], YUV, YCbCr, YCgCr, YPbPr and HSV models [17], structured matrix decomposition model [18], morphological segmentation [19] and deep learning techniques [20] [21]. Watershed method experiences over-segmentation and accords connected components at the cost of computation time. MSRM is computationally slow as it needs human intervention, but gives better results than other techniques.

Object detection methods act as a fusion of image categorisation and object finding. These methods generate one or more bounding boxes for the given input image and label each bounding box. These methods can deal with multi-class categorisation and localisation of objects with varied instances. Various methods for identification of objects, including Retina-Net, Single-Shot MultiBox Detector (SSD) and Fast RCNN, tackle the threats like data limitation and object detection. Detectron was developed by Facebook AI Research (FAIR). It comprises implementations for the ensuing object identification: Mask R-CNN, Retina Net, RPN, Fast R-CNN, Tensor Mask, Point Rend, Dense Pose, and more [22]. Detectron2 is a total rewrite of the first Detectron, which was launched in 2018. Detectron2 supports object identification, including human pose prediction, bounding boxes and instance segmentation masks. Beyond that, Detectron2 supports semantic and panoptic segmentation (a task that amalgamates semantic and instance segmentation). Moving the entire training pipeline to GPU made Detectron2 speedy compared to the first Detectron. The use of multiple GPU servers for training makes it much easier to scale the training to large data sets.

Instance Detection refers to classifying and localising an object with a bounding box around it. In contrast to using the classifier to detect and classify the object, the entire operation is performed with one network. A single neural network discovers the bounding box and group probability in one scanning, which brings its accomplishment to an entirely new height. Because the entire operation is a single pipeline, it may be optimised further [23]. The realisation of Detectron2 also includes a model zoo which includes pre-trained models for object detection, semantic segmentation, and keypoint detection. Detectron2 is designed to enhance learning by offering fast training and attending to the challenges companies face going from research to production. Many research works have been carried out for object identification using technologies like CNNs. However, a study of different approaches has shown that Detectron2 and EfficientDet are promising. The above survey concludes that yield estimation requires accurate segmentation and more research is required to identify the maturity levels of the arecanut and estimate the yield.

III. Methodology

This section presents the methodologies followed for segmenting the arecanut bunch from the input image and the method used to estimate the yield. Methodologies followed to segment the input image are: U-Net, Mask R-CNN (Mask Region-Based Convolutional Neural Network), U2-Net and the Graph cut. The output of segmentation is routed to Detectron2 to determine the yield count of the segmented arecanut bunch.

U-Net: U-Net depicted in Figure 1 is primarily a CNN developed for segmentation of biomedical images. CNN has been developed primarily for classification purposes, and CNN outputs a unique class label for each image. The expected output of image partitioning is a precise localisation of the point of interest and the allocation of class labels for every pixel. The belief in developing U-Net is centred around this problem. U-Net is built on top of a Fully Convolutional Network (FCN) and has been extended to work with fewer training examples with improved segmentation performance. U-Net can be conceptually thought of as an encoder network followed by a decoder. The first half of the architecture is an encoder, an experienced classification network like VGG/

Residual Network (ResNet), where max-pool down sampling follows convolution blocks to map the input image onto feature vectors with many different levels. The second half of the architecture is a decoder, which necessarily performs the opposite of down-sampling. It is the balanced expansion path to extrapolate meaningful distinguishing features derived from the encoder onto the pixel space to obtain exact localisation, applying transposed convolutions. Hence, it is an end-to-end FCN. U-Net consists only of convolutional layers and no dense layer, so it can receive images of any size [20].

Mask R-CNN: Mask R-CNN, shown in Figure 2 is a deep neural network variant that performs instantaneous segmentation. It finds objects in an image concurrently and provides a bounding box, class label and segmentation mask for each occurrence of an object. The Mask R-CNN framework consists of two stages. The first stage is the Region Proposal Network (RPN), which examines the image and outputs proposals i.e. places where an object could be present. The second stage is the binary mask classifier, which recognises the proposals and outputs bounding boxes and masks. Mask R-CNN is an expansion of Faster R-CNN with the incorporation of RoIAlign, which performs pixel-to-pixel alignment. Faster R-CNN also contains two stages: RPN, the first stage proposes bounding boxes. The second stage derives features from bounding boxes and achieves classification [20].

U2-Net: The U2-Net depicted in Figure 3, a deep learning model, improves the effectiveness of segmentation by automatically extracting features from input images. The U2-Net architecture is realised as a coder followed by a decoder structure. Many U-Net architectures are arranged jointly to form a (U-n Net), where n denotes the number of U-Net units. Here, n is set to 2. U2-Net is a two-level ingrained U-structure which includes three major parts: a six-stage encoder, a five-stage decoder, and a salience fusion unit. Each stage contains a precise residual U-block (RSU) to derive intra-stage multi-scale attributes. Hence, ingrained U-structure obtains intra-stage multi-scale features and aggregates inter-stage multi-level features [21].

Graph Cut Segmentation: The basic idea of graph cut segmentation is that the user marks the object and surroundings interactively on the input image, providing a hint about what the end user intends to segment. Segmentation is carried out by obtaining the superpixels of image elements using simple linear iterative clustering (SLIC) to reduce computational time and amount of noise. A graph is then constructed using superpixels as nodes and the marked regions as the two end vertices. An undirected graph is constructed using edges obtained by taking the difference between the histograms of the two neighbouring superpixels. This modifies the pixel-based representation to a superpixel-based representation of an image. Image segmentation is treated as a labelling problem in a graph. It assigns labels to each superpixel in an image, so that elements in a specific region with similar features concerning attributes such as colour, intensity, or texture have the same label and elements with different features have different labels. The image is automatically segmented by determining a global optimal cut of all segmentations. Graph cut is employed to obtain a mask image using a min-cut/max-flow algorithm to solve the energy function. The segmented image is obtained by performing pixel multiplication of the mask and the input image [24]. The flow of the work done is depicted in Figure 4.

Detectron2: We experimented with Detectron2's implementation of Faster R-CNN with FPN, which is a primary bounding box locator. The FPN backbone casts the framework as a multi-scale detector, which may result in very good accuracy for large and small objects, making it more powerful. The architecture of the Detectron2 model is depicted in Figure 5 [25]. FPN draws feature maps from input images at different scales and outputs as P2, P3, P4, P5, and P6. These feature maps are then given to RPN, which produces 1000 box proposals along with confidence scores. The features P2, P3, P4, and P5 are given to the box head, which produces 100 boxes using non-max suppression. Box Head crops and warps attribute maps into many preset-size features, utilising proposal boxes and obtains fine-tuned box locations and recognition results using fully connected layers. It combines RPN and the classification phase into a unique network, resulting in a much more compact architecture with improved object detection precision and capability, making it applicable for real-time detectors. Detectron2 predicts bounding boxes and respective classes using a unique feed-forward network when compared to earlier region proposal-based detector that detects in a two-stage pipeline [26].

IV. Experimentation

Experimentation has been carried out on a data set of 1017 images provided by R. Dhanesha et al. [16], which includes 388 ripe and 629 unripe images of 4160x3120 resolution, jpeg format, to assess the methodologies. Deep learning techniques require more data samples to train compared to machine learning methods. The amount of training samples may be increased using data augmentation. Details of the experimental setup and assessment of the segmentation models can be found in [20] [21] [24]. The example segmentation results obtained for both ripe and unripe images are shown in Figure 6.

Detectron2 is a pre-trained model with the COCO dataset. It has been trained using the Detectron2 model zoo. Training samples were annotated by drawing a bounding box for each arecanut of the ground truth images using the graphical image annotation tool, following an LWYS (label what you see) approach. Sample annotations are displayed in Figure 7. Detectron2 accepts data in COCO format. The COCO format accepts a JSON file that includes information about image size, and the format used is BoxMode.XYWH_ABS. The object detection model we have used is ResNet faster_rcnn_R_50_FPN_3X, a backbone network used to derive features from the input image among different models of Detectron2. Detectron2 is trained using 1017 segmented images, of which 80% are used for training and the remaining 20% are considered for validation. Image samples are resized to a resolution of 416x416. The training samples are increased by data augmentation, applying transformations to the available images. It is done using domain-specific methods. Five types of augmentations are applied to image data: Image shifts via width_shift_range and height_shift_range, flip via horizontal_flip and vertical_flip, rotations, brightness and zoom. ImageDataGenerator class of the Keras library has been used for data augmentation.

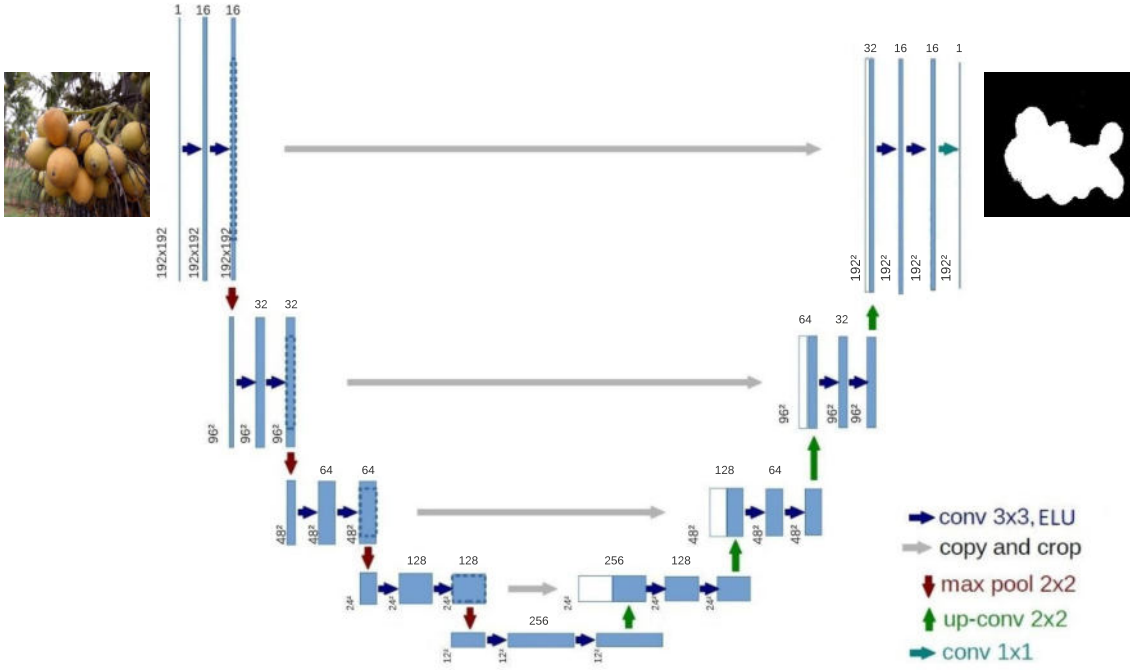

Fig. 1. U-Net Architecture

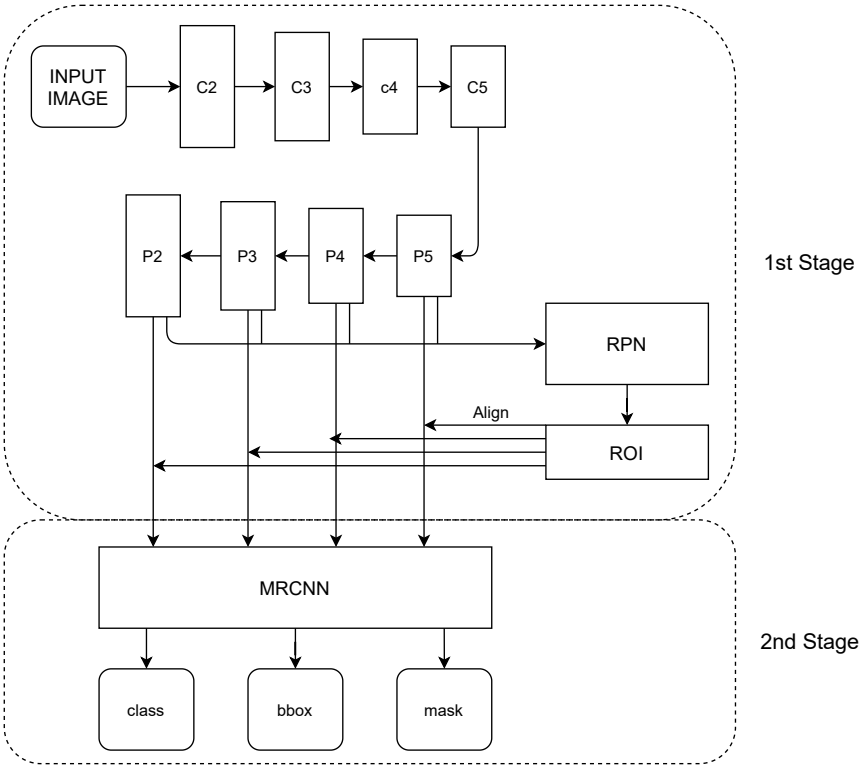

Fig. 2. Architecture of MRCNN

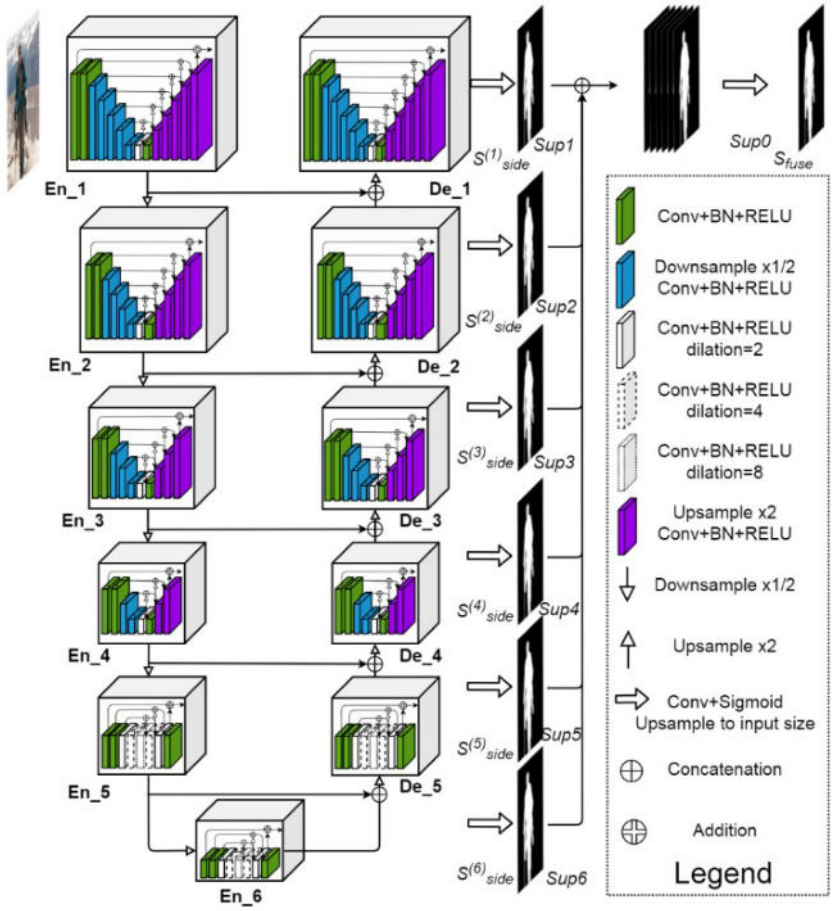

Fig. 3. U2-Net Architecture

TABLE I
TRAINING PERFORMANCE OF DETECTRON2

Image	Precision	Recall	F1-Score	IoU
Unsegmented	73.0%	74.0%	73.0%	57.10%
Segmented	90.0%	82.0%	85.0%	68.64%

Detectron2 is trained with a learning rate of 0.001, momentum of 0.0005, weight decay of 0.9 and a maximum batch size of 6000. The training performance of Detectron2 is summarised in Table 1. Training and testing were done on a machine with the following configurations: Intel Xeon(R) 64-bit, 2.3GHz CPU, 16GBRam, 12GB NVIDIA Tesla T4 GPU, CUDA 11.2, cuDNN v7.6.5, OpenCV v4.2.0.

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

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

$$F1 - Score = 2 \bullet \frac{Precision \bullet Recall}{Precision + Recall} \tag{3}$$

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{4}$$

where TP - True Positive FN - False Negative and FP - False Positive

V. RESULTS

The segmentation performances are evaluated using four standard metrics: Precision, Recall, F1-Score and IoU, given in (1) to (4). Table II describes the test performance comparison with other methods. Mean Average Percentage Error (MAPE) given by (5) is used to measure the performance of the Detectron2 yield count. MAPE is found to be 3.23% for segmented ripe images (Table III) and 4.1% for segmented unripe images (Table IV), indicating improved performance due to the removal of irrelevant surrounding data. Table V shows the classification performance and yield count for 15 images. The sample yield outputs for unsegmented and segmented images are presented in Figure 8.

$$MAPE = \frac{1}{N} \sum_{i=1}^{N} \frac{|n_a - n_b|}{n_a} \tag{5}$$

VI. CONCLUSIONS

This article presents three deep learning techniques and a graph cut method for segmentation of an arecanut bunch. The graph cut technique gives better outcomes than other methods. The output of the graph cut segmentation is routed to Detectron2 to determine the yield count of the segmented arecanut bunch. The yield count experimentation is carried out with Detectron2 on both ripe and unripe segmented images. Detectron2 gives a better accuracy of 96.77% yield count from segmented images, which is very good compared

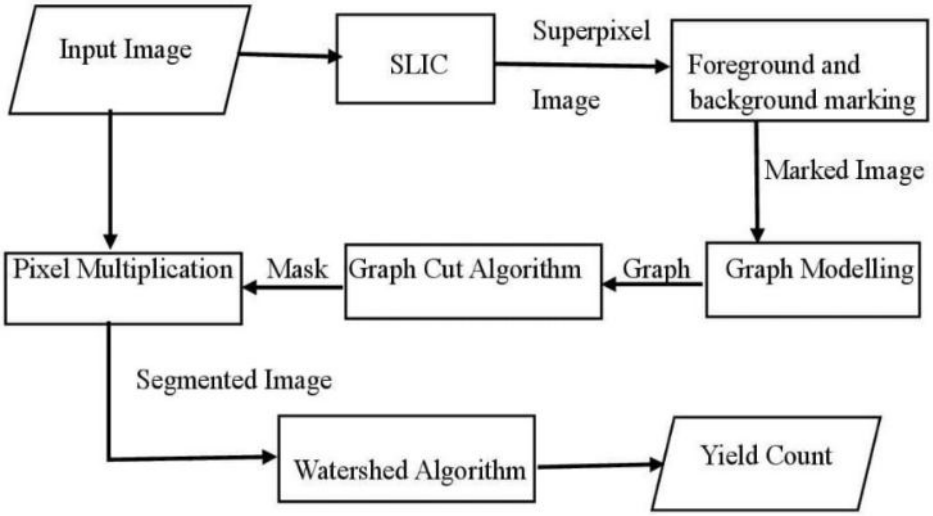

Fig. 4. Flow Diagram of Graph Cut Segmentation and Yield Count

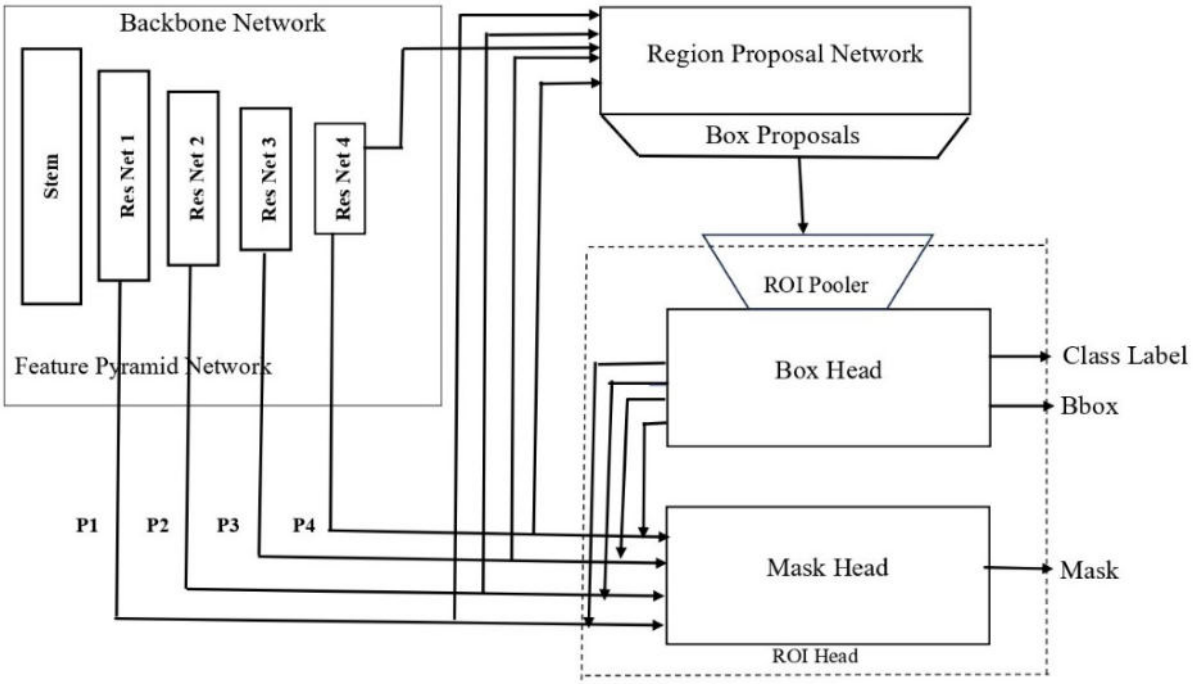

Fig. 5. Architecture of Detectron2

TABLE II
VALIDATION SET SEGMENTATION PERFORMANCE

Author	Method	IoU	Pr	Re	F1
Dhanesha et al [15]	YCgCr				53.54%
Dhanesha et al [17]	YCgCr	72.77%			83.62%
	HSV	66.58%			79.0%
Anitha et al [20]	U-Net Ripe	54.61%	61.53%	87.07%	68.26%
	U-Net Unripe	58.07%	74.71%	77.15%	72.95%
	MRCNN Ripe	61.01%	73.57%	81.84%	72.95%
	MRCNN Unripe	65.98%	89.86%	73.14%	78.68%
Anitha et al [21]	U2-Net Ripe	71.24%	93.07%	69.23%	83.21%
	U2-Net Unripe	65.74%	89.42%	72.73%	79.32%
Anitha et al [24]	Graph cut	85.78%			93.15%

MRCNN - Mask Region-Based Convolutional Neural Networks

Fig. 6. Representative results of segmentation

Fig. 7. Sample annotated images

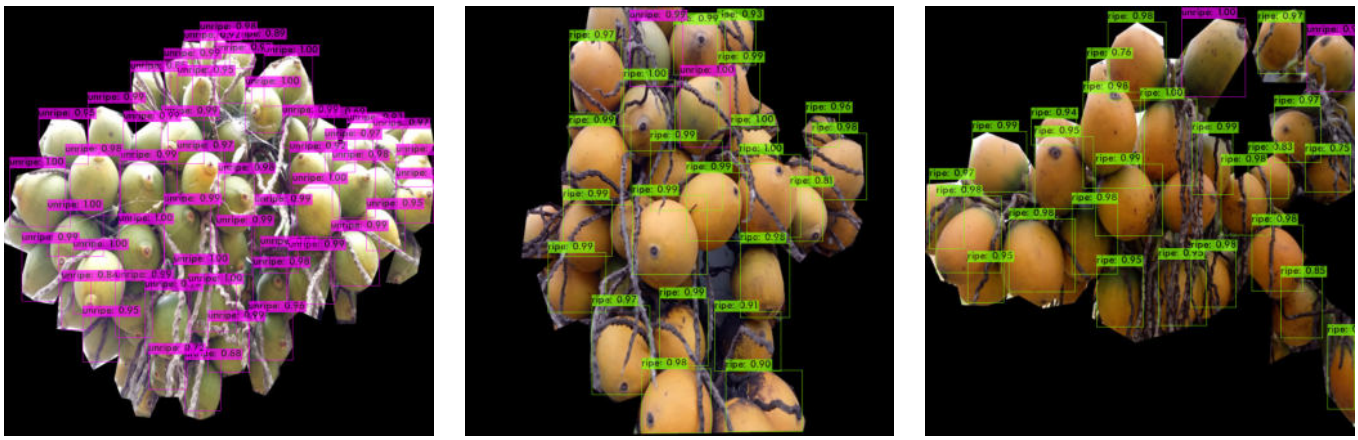

Fig. 8. Representative results of yield count

TABLE III
ERROR OF YIELD COUNT FOR RIPE IMAGES

Image	Actual n_a	Detected n_b	Error (%) $\frac{\lvert n_a - n_b \rvert}{n_a}$
1	46	44	4.34
2	26	26	0
3	42	39	7.14
4	25	24	4
5	41	39	4.87
6	34	33	2.94
7	14	14	0
8	40	41	2.5
9	33	33	0
10	45	44	2.2
11	46	42	4.34
12	26	26	0
13	70	68	2.85
14	24	26	8.33
15	40	42	5.0
		MAPE	3.23

TABLE IV
ERROR OF YIELD COUNT FOR UNRIPE IMAGES

Image	Actual n_a	Detected n_b	Error (%) $\frac{\lvert n_a - n_b \rvert}{n_a}$
1	36	36	0
2	49	47	4.08
3	35	34	2.85
4	35	35	0
5	44	41	6.81
6	38	40	5.26
7	38	37	2.63
8	44	40	9.09
9	38	36	5.26
10	42	42	0
11	44	43	2.27
12	21	20	4.76
13	44	45	2.27
14	24	22	8.33
15	38	41	7.89
		MAPE	4.1

to the yield count of unsegmented images of 95.9%. The yield count performance may be enhanced further using additional training samples and removing the inflorescence (male flowers).

REFERENCES

[1] "https://www.tssindia.in/index.php/other1/about-arecanut-menu/uses-of-areca-and-components."

[2] "Success story: Areca referral laboratory at uahs(university of agriculture and horticulture science), shivamogga," 2017.

TABLE V
CLASSIFICATION AND YIELD COUNT OF ARECANUTS

Image	Actual		Identified	
	Ripe	Unripe	Ripe	Unripe
1	12	70	7	72
2	12	39	12	41
3	23	22	25	24
4	0	48	3	48
5	11	24	11	23
6	0	71	0	70
7	4	55	4	57
8	0	72	2	65
9	45	7	48	2
10	0	56	6	44
11	0	51	0	53
12	60	22	61	18
13	14	41	14	45
14	26	18	26	19
15	28	21	30	16

[3] S. Bargoti and J. P. Underwood, "Image segmentation for fruit detection and yield estimation in apple orchards," *Journal of Field Robotics*, 2017.

[4] H. L. et al., "Region-based colour modelling for joint crop and maize tassel segmentation," *Biosystems Engineering*, vol. 147, pp. 139–150, 2016.

[5] L. Z. Bo Peng and D. Zhang, "A survey of graph theoretical approaches to image segmentation," *Pattern Recognition*, vol. 46, pp. 1020–1038, 2013.

[6] H. M. Philipe A. Dias, Amy Tabb, "Multispecies fruit flower detection using a refined semantic segmentation network," *IEEE Robotics and Automation Letters*, vol. 3, pp. 3003–3010, 2018.

[7] R. M. Carlos S. Pereira and M. J. C. S. Reis, "Recent advances in image processing techniques for automated harvesting purposes: A review," *Intelligent Systems Conference*, pp. 566–575, 2017.

[8] L. M. S. S. Y. R. Kashfaqurah Phooi Seng, Li-Minn Ang, "Computer vision and machine learning for viticulture technology," *IEEE Access*, vol. 6, pp. 67 494–67 510, 2018.

[9] A. A. Hellicar, "Identification of mature grape bunches using image processing and computational intelligence methods," *IEEE*, 2014.

[10] D. F. et al., "Vineyard yield estimation based on the analysis of high resolution images obtained with artificial illumination at night," *Sensors*, vol. 15(4), pp. 8284–8301, 2015.

[11] Y. L. et al., "In-field cotton detection via region-based semantic image segmentation," *Computers and Electronics in Agriculture*, vol. 127, pp. 475–486, 2016.

[12] Q. W. et al., "Automated crop yield estimation for apple orchards," *Experimental Robotics, STAR*, vol. 88, pp. 745–758, 2013.

[13] M. N. Reza, "Rice yield estimation based on k-means clustering with graph-cut segmetation using low-altitude uav images," *Biosystems Engineering*, vol. 177, pp. 109–121, 2016.

[14] S. K. N. Siddesha S and V. N. M. Aradhya., "A study of different color segmentation techniques for crop bunch in arecanut," *Handbook of Research on Advanced Hybrid Intelligent Techniques and Applications*, 2016.

[15] R. Dhanesha and S. N. C. L, "A novel approach for segmentation of arecanut bunches using active contouring," *Studies in Computational Intelligence*, vol. 771, pp. 677–682, 2019.

[16] S. N. C. L. R Dhanesha and Y. Kantharaj, "Segmentation of arecanut bunches using ycgcr color model," *1st Int. Conf. on Advances in Information Technology (ICAIT)*, p. 5053, 2019.

[17] D. R, "Segmentation of arecanut bunches: A comparative study of different color models," *IEEE Mysore Sub Section International Conference (MysuruCon)*, p. 752758, 2021.

[18] M. S. H Chandrashekhara, "Classification of healthy and diseased arecanuts using svm classifier," *Int. Journal of Computer Science and Engineering*, p. 544548, 2019.

[19] R. M. S. Kusumadhara S, "Segmentation of white chali arecanuts using soft computing methods," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 9(7), pp. 1049–1055, 2020.

[20] A. A. C. et al., "Arecanut bunch segmentation using deep learning techniques," *Int. Journal of Circuits, Systems and Signal Processing*, vol. 16, pp. 1064–1073, 2022.

[21] K. A. N. et al., "Segmentation and yield count of an arecanut bunch using deep learning techniques," *IAES Int. Journal of Artificial Intelligence (IJAI)*, vol. 13, p. 542553, 2024.

[22] "https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e."

[23] C. P. V. Pham and T. Dang, "Road damage detection and classification with detectron2 and faster r-cnn," *IEEE Int. Conf. on Big Data (Big Data)*, pp. 5592–5601, 2020.

[24] S. N. C. L. Anitha A C and K. A. N, "Graph cut segmentation and watershed algorithm for yield count of an arecanut bunch," *Ingnierie des Systmes dInformation*, vol. 29(6), pp. 2425–2431, 2024.

[25] F. M. W.-Y. L. R. G. Y. Wu, A. Kirillov, "A pattern recognition strategy for visual grape bunch detection in detectron2. https://github.com/facebookresearch/detectron2," 2019.

[26] G. P. R. Singh, S. Shetty and P. J. Bide, "Helmet detection using detectron2 and efficientdet," *12th Int. Conf. on Computing Communication and Networking Technologies (ICCCNT)*, pp. 1–5, 2021.