Segmentation and yield count of an arecanut bunch using deep learning techniques

Anitha Arekattedoddi Chikkalingaiah[1], RudraNaik Dhanesha[1], Shrinivasa Naika Chikkathore Palya Laxmana[1], Krishna Alabujanahalli Neelegowda[2], Anirudh Mangala Puttaswamy[2], Pushkar Ayengar[2]

[1]Department of Studies in Computer Science and Engineering, University B.D.T College of Engineering,
Constituent College of Visvesvaraya Technological University, Davanagere, India
[2]Department of Computer Science and Engineering, SJB Institute of Technology, Visvesvaraya Technological University,
Bengaluru, India

Article Info

Article history:

Received Jun 26, 2022
Revised Feb 11, 2023
Accepted Mar 10, 2023

Keywords:

Arecanut
Segmentation
U-Net squared model
Yield
You only look once

ABSTRACT

Arecanut is one of Southeast Asia's most significant commercial crops. This work aims at helping arecanut farmers get an estimate of the yield of their orchards. This paper presents deep-learning-based methods for segmenting arecanut bunch from the images and yield estimation. Segmentation is a fundamental task in any vision-based system for crop growth monitoring and is done using U-Net squared model. The yield of the crop is estimated using Yolov4. Experiments were done to measure the performance and compared with benchmark segmentation and yield estimation with other commodities, as there were no benchmarks for the arecanut. U-Net squared model has achieved a training accuracy of 88% and validation accuracy of 85%. Yolo shows excellent performance of 94.7% accuracy for segmented images, which is very good compared to similar crops.

Corresponding Author:

Anitha Arekattedoddi Chikkalingaiah
Department of Studies in Computer Science and Engineering, University B.D.T College of Engineering,
Constituent College of Visvesvaraya Technological University
Davanagere, Karnataka-577004, India
Email: anithakrishna2008@gmail.com

1. INTRODUCTION

Agriculture is humanity's primary source of income and a critical element of each country's economy. As a traditional occupation, agriculture is the backbone of the Indian economy. A strong farming industry offers national food safety, an origin of revenue and job opportunities. Precision agriculture could help with this. Precision agriculture can ease many rising environmental, economic, market and societal problems [1]. Precision agriculture technologies are predicted to positively influence agricultural output concerning two issues: lucrative for farmers and eco-friendly environmental advantages for the general population. Precision agriculture aims to maximize profit, reduce cost and reduce environmental harm by tailoring agricultural techniques to the location's needs. As a result, agricultural and engineering companies are creating cutting-edge machine vision technology to aid farmers in precision farming. Attention-based farming is a crop supervision method aiming to find the required type and quantity of inputs to the actual crop yield for tiny areas in a farm field. Precision agriculture's financial and eco-friendly benefits may be found in the decreased use of aqua, fertilizers and defoliant, and farming tools. Precision agriculture detects site-specific variations in the fields and adapts techniques as necessary.

The standard masticatory nut, often known as arecanut, comes from the arecanut palm. The arecanut is one of Southeast Asia's most significant commercial crops. Arecanut farming is primarily limited to Karnataka, Kerala and Assam in India. Karnataka is the most populous state in terms of both area and output [2]. The areca palm is a tall-stemmed erect palm that grows to different heights depending on the climate. Palms that reach 30 meters or more heights are relatively unusual, making it challenging to distinguish ripe from sick arecanuts while harvesting. Arecanut crop bunch segmentation seeks to determine whether a specific areca plant has diseased nuts, regularly relieving farmers of a load of ascending the towering trees. Arecanut is a profitable crop in southern India. The market sets the levy of arecanut based on the maturity level. Identifying the maturity stage of arecanut before harvesting is essential for increasing profitability. Farmers require knowledge to determine ripeness level; else, their crops would provide less profit. Automated segmentation of arecanut bunch in a given input image can be utilized to find maturity level, health and finally for yield estimation. Very few attempts have been made for the segmentation, disease detection, harvesting and grading of arecanut. This research intends to present an accurate arecanut bunch segmentation and yield-counting technique, which is very efficient for the given input image.

The rest of the paper is detailed as follows. Section 2 presents the work done on the related crop segmentation and yield count. Section 3 describes the method for arecanut bunch segmentation and its performance analysis. Section 4 describes the technique for yield count and its performance analysis. A summary of the work is presented in the last section.

2. RELATED WORK

Segmentation is a significant step in a machine vision system used for the analysis or interpretation of an image. Its achievement primarily impacts the conduct of the whole vision system. Image segmentation is a perceptual grouping of pixels based on similarity and proximity [3]. Automated segmentation is essential as manual segmentation is complicated, consumes time, and is subjective and error-prone. Most existing segmentation techniques focus on a two-class classification approach, i.e. object and background. Background elimination is the primary step and must be done most suitably to avoid misclassification. Segmentation is complex because the color of the crop, shadows, and inter-reflection varies as the illumination changes in the outdoor field. Segmenting childish crops is much more problematic as it is green and resembles the background foliage. Slight variations in different parts of a single crop bunch increase the complexity of crop segmentation. Despite the above limitations, color-based segmentation also has its superiority. Color is the most potent visual cue to discriminate an object from the background. Also, color is primarily unchanging to transition in size, orientation and occlusion [4]. Color-based methods are mostly classified into two categories: pixel-based and region-based methods. Excess green minus excess red index (ExGR) color model, a pixel-based approach, demonstrated better results compared to other color models for the green vegetation segmentation [5].

A good many yield estimation techniques require image segmentation/detection, which includes mango crop yield estimation using (red, green, blue (RGB) and Y is luma (brightness) (YCbCr), Cb is B-Y, and Cr is R-Y) color space and texture information based on pixel adjacency [6], detection of red apples using hue saturation value (HSV) and green apples using hue saturation intensity (HSI) profile [7], threshold-based segmentation of reddish grapes using the Otsu threshold applied to the H layer of HSV color space gave better results compared to the histogram and linear color model, Bayesian classifier and Mahalanobis distance methods [8]. Pixel-based methods are easy and efficient but incorporate noise. Researchers focus on region-based strategies mainly based on edge detection and shape fitting, which includes apple segmentation [9], cotton detection [10], maize tassel segmentation [11], apple identification using thresholding, edge detection, circular Hough transforms, clustering and K-nearest neighbours classification of color and texture features [12]. Citrus detection using HSV space follows thresholding and watershed segmentation [13]. Rice grains are segmented by converting RGB image into lab color space followed by clustering and graph-cut segmentation [14].

An interactive arecanut bunch segmentation using maximum similarity-based region merging (MSRM) gave better results than thresholding, clustering, and watershed [15]. Arecanut bunch segmentation using active contours [16], YCgCr color model [17], different color models [18], and deep learning techniques [19] are the few attempts for the crop areca. The human visual system often combines more than one visual cues to enhance perceptual performance. Segmentation is better when we combine more than one feature [20].

Mango segmentation combining color and texture features can be found in [21]. Often, things have a particular shape. Segmentation based on shape and size is often desired. Support vector machine-radial basis function (SVM-RBF) classifier and density-based spatial clustering of applications with noise (DBSCAN) based grape bunch detection using histograms of oriented gradients (HOG) and local binary pattern (LBP) shape and texture descriptors [22], active contours and scale-invariant feature transform (SIFT) based segmentation of tomatoes using shape and position information [23] are attempted in this direction. Region information is combined with gradient information to approximate the elliptic shape of the tomato boundary.

Machine learning (ML) based segmentation approaches include segmenting matured grape bunches by finding edges, then determining circles and then classifying them as background and grapes using support vector machines [24]. Segmentation using those hand-engineered features is less powerful. Investigators focused on deep learning-based approaches. Though better results were obtained for apple segmentation using multi-scale multi-layered perceptrons (MLP) and convolutional neural network (CNN) and yield estimation using watershed and circular Hough transform [25], they are susceptible to occlusion and illumination. Multiclass (fruit, leaves and branches) almond fruit segmentation using feature learning with a conditional random field (CRF) [26] automatically generates the set of rules from the data rather than make use of pre-defined feature descriptors. That unsupervised feature learning approach automatically captures the most appropriate features from the data. Mango counting applies MagnoNet, a deep CNN-based model, with a contour-based connected object detection model, which gives better results. It is invariant to illumination changes, scaling, contrast and occlusion [27].

3. SEGMENTATION

The section describes the framework of U2-Net for segmentation of arecanut bunch from the input image eliminating the unwanted background information. The architecture of U2-Net shown in Figure 1 can be conceptually viewed as an encoder accompanied by a decoder framework. Multiple U-Net-like arrangements are stacked together to construct flow models and compiled as (U-n Net), n represents a number of U-Net units. The challenge is the increased costs of memory and computation by n times. In this framework, each encoder-decoder U-Net stage includes a ReSidual U-block (RSU), a down and up sampling encoder-decoder. The purpose of this block is to use residuals, a multi-scale features in place of original features. According to researchers, this will introduce the desired effect of keeping the fine-grained details which enforce the network to derive features at multiple scales from a residual block. For example, En_1 is one RSU block. As illustrated in Figure 1, U2-Net consists of three main parts: five stages of the encoder, decoder, a salience fusion unit and the last stage consists of the encoder and a fusion unit.

Encoders En_4, En_3, En_2 and En_1 uses residual blocks RSU-4, RSU-5, RSU-6 and RSU-7 respectively. Digits 7, 6, 5 and 4 represent the height (L) of RSU blocks and are customized by the spatial resolution of the input feature maps. Large L has been used to represent more information about feature maps with large resolutions. The feature maps resolution in En_6 and En_5 are fairly low; further down-sampling of those feature maps results in a loss of contextual information. Hence, a dilated version RSU-4F ("'F'' represents dilated version) is used in both En_6 and En_5. Therefore, all the intervening feature maps of RSU-4F posses the resolution of input feature maps. For En_6, decoder stages possess information related to symmetrical encoder stages. In De_5, the dilated version RSU-4F has been used and is indistinguishable from the one used in En_6 and En_5. The input for each decoder is the sequence of up-sampled feature maps of its preceding stage with its symmetrical encoder phase. The saliency probability map is generated using the final fusion module of the saliency map. Indistinguishable to Holistically-Nested Edge Detection [28], U2-Net begins generating six side output saliency probability maps $S^{(6)}_{side}$, $S^{(5)}_{side}$, $S^{(4)}_{side}$, $S^{(3)}_{side}$, $S^{(2)}_{side}$ and $S^{(1)}_{side}$ from En_6, De_5, De_4, De_3, De_2 and De_1 using 3×3 convolution layer, a sigmoid function and then up-samples the saliency maps abovementioned to the size of the input image. Further, fuse all the saliency maps using a concatenation operation accompanied by a 1×1 convolution and a sigmoid function to produce the finishing saliency probability map S_{fuse}. To summarize, U2-Net is built on RSU blocks with no pre-trained backbones. It permits deep networks with opulent multi-scale features with comparatively low computing and memory costs.

3.1. Training

Deep learning-based techniques need vast data sets and proper labelling, thereby longer training and less testing time than other machine learning-based methods. It is not feasible to collect enormous amounts of data for training. The training data size can be increased using a technique called data augmentation by applying transformations to the database images. ImageDataGenerator class of the Keras library has been used for data augmentation. The labelling task is difficult and requires experts to annotate input images. The maximum current annotation tools are based on polygonal approximation of the object boundaries. The object boundary in each input image is encoded as a mask, a set of polygon points. Annotations were completed using the Labelme tool, and each annotation was stored as a JSON file. JSON was converted to binary images using labelme2voc-white representing the arecanut area and background represented by the black. A sample input image, labelling and masking are shown in Figure 2.

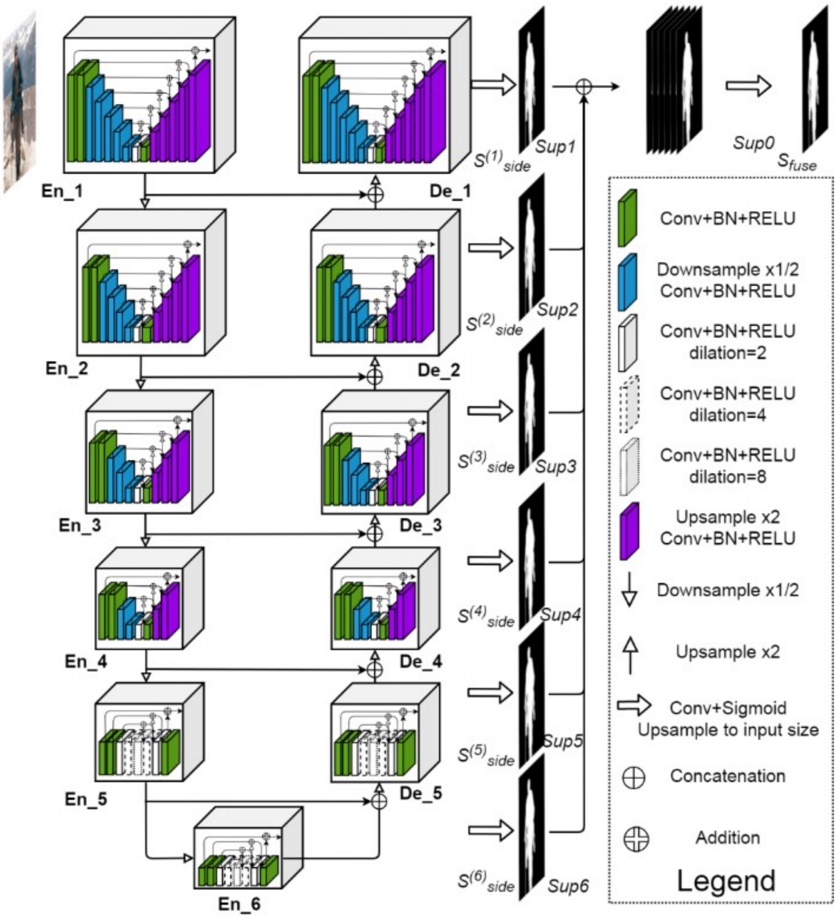


Figure 1. Architecture of U2-Net

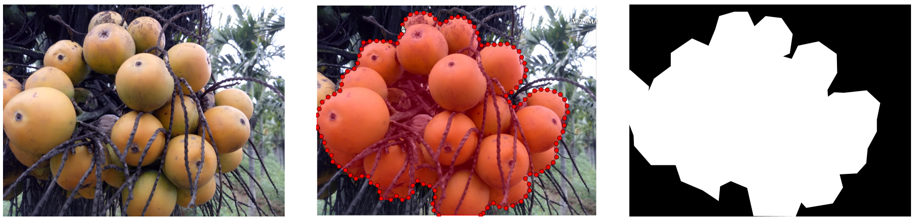
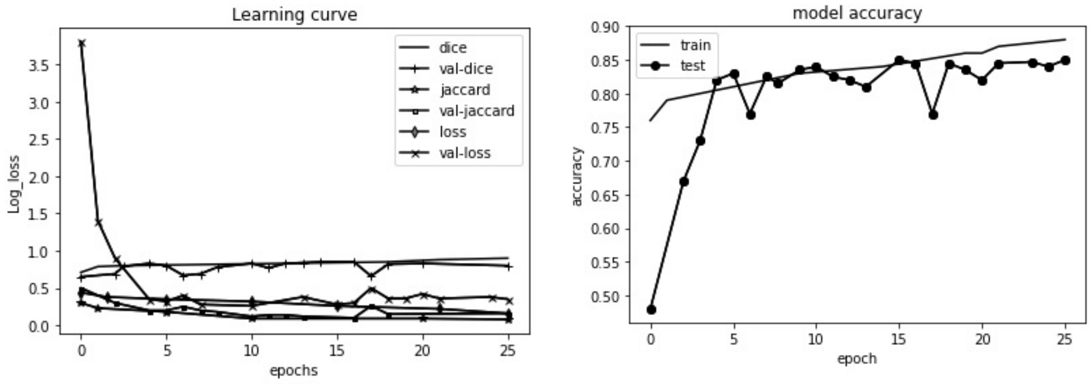


Figure 2. Labelling and masking of sample input images

The model has been trained and examined using the data set [17] that consists of 388 ripe and 629 unripe images. All images are of 4160×3120 resolution and stored in jpeg format. The model has been trained with 310 images and validated with 78 images with a total of 388 ripe data sets (80:20 split) and evaluated the accomplishment on both ripe and unripe image sets. Input and the corresponding mask images have been used to train the model. The model has been trained with resized images of 720×720 resolution, randomly flipping vertically and trimming to 400×400. The network and all the convolutional layers have been trained from scratch. The loss weights $w_{side}^{(m)}$ and w_{fuse} are all adjusted to 1. The network has been trained using Adam optimizer, and all the hyperparameters are adjusted to default initial values (weight decay=0, learning rate lr=1e-3, eps=1e-8 betas= (0.9, 0.999)). The model has been trained for about 20 hours with a batch size of 12, and the loss converges after 5k iterations. Images are resized to 720×720 during testing and fed to the network to generate saliency maps. The predicted 400×400 saliency maps are resized to the original input image of 720×720. Bi-linear interpolation has been used for the resizing process. The model learning curve concerning dice, jaccard and loss for training and validation is shown in Figure 3(a), and the accuracy plot is shown in Figure 3(b). A summary of the training performance is depicted in Table 1. The model has achieved training and validation accuracy of 88% and 85%, respectively.



(a)

(b)

Figure 3. Model performance: (a) learning curve and (b) accuracy plot

Table 1. Segmentation performance calculated over validation set

Loss	Accuracy	Dice
0.2652	0.8506	0.8594

3.2. Performance analysis

Experimentation has been carried out to determine the segmentation performance using both the data sets: ripe and unripe. Four standard measures, namely: precision (Pr), recall (Re), F1-score (F1), and IoU given by (1) to (4), have been used to judge the accomplishment of the model. There are minimal attempts made for arecanut segmentation [16]–[19], and the evaluation of achievement has been concluded for very few images [15], [16]. Table 2 summarizes the test performance of the segmentation. The higher values indicate more remarkable performance. The appropriateness of this method is evidenced by its high segmentation performance against both ripe and unripe data sets that differ in terms of color. The sample segmentation outputs achieved by U2-Net model for both ripe and unripe images is shown in Figure 4. The results obtained are better than other methods. The model has been implemented using Pytorch 0.4.0. The entire training and testing is done using an octa-core, 16 threads PC of an Intel(R) Core(TM) i5-10300H CPU with 2.50 GHz, 16GB RAM and an NVIDIA GeForce RTX 2060 GPU (6 GB memory).

$$Pr = \frac{T_r P}{T_r P + F_l P} \tag{1}$$

$$Re = \frac{T_r P}{T_r P + F_l N} \tag{2}$$

$$F1 = 2 \bullet \frac{Pr \bullet Re}{Pr + Re} \tag{3}$$

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

Where:

- T_rP - true positive
- F_lN - false negative
- F_lP - false positive

Table 2. Segmentation performance calculated over validation set

Author	Method	IoU	Pr	Re	F1
Dhanesha *et al.* [17]	YCgCr				53.54%
Dhanesha *et al.* [18]	YCgCr	72.77%			83.62%
	HSV	66.58%			79.0%
	U-Net Ripe	54.61%	61.53%	87.07%	68.26%
Anitha *et al.* [19]	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%
This paper	U2-Net Ripe	71.24%	93.07%	69.23%	83.21%
	U2-Net Unripe	65.74%	89.42%	72.73%	79.32%

*MRCNN - mask region-based convolutional neural networks

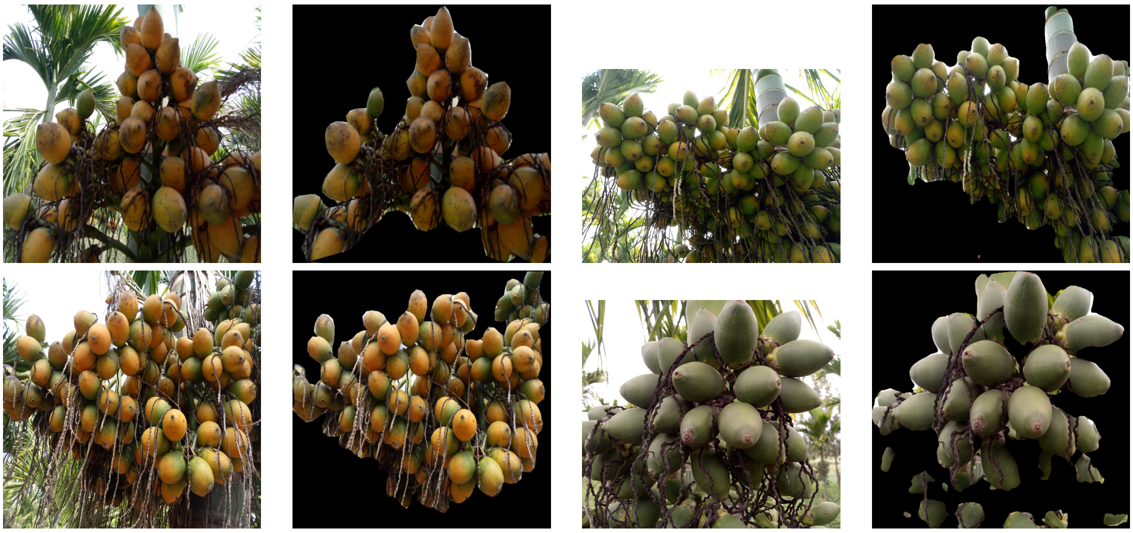


Figure 4. Representative results of segmentation

4. YIELD COUNT OF ARECANUT

The objective is to build an object detection model that can count the number of areca nuts in a given image. Object detection methods act as a fusion of image categorization and object finding. It generates one or more bounding boxes and labels each bounding box. These methods can deal with multi-class categorization, localization, and objects with many incidents. Different kinds of object detection include Retina-Net, single-shot multiBox detector (SSD) and Fast RCNN. These methods can address the challenges, such as limitation of data and object identification modelling, but need to be able to identify the objects in a single algorithm pass. You only look once (YOLO) has gained popularity for its higher performance over other object identification methods. Yolo [29]–[31] merges the classification phase and region proposal network (RPN) into one network, resulting in a more compact object identification model with more excellent computational order, making them suitable for instantaneous applications. Compared to earlier region proposal-based detectors [32] and [33] detect objects in two stages, Yolo forecasts the bounding boxes and the corresponding class label in one run using a single feed-forward network. Yolov2 [30], the second kind of Yolo [29], was presented aiming at the considerable enhancement of performance and speed. Faster R-CNN spurred the introduction of anchors for detection in Yolov2. The anchors increase detection performance, reduce challenges, and simplify the network training process. Batch normalization [34] was introduced to the convolution layers in the meantime, pushing mean Average Precision (mAP) to 95.14% and skipping connection [35]. The recall and localization performance of Yolov2 was enhanced when compared to Yolo. Yolov3 [31] builds on Yolo, and Yolov2 became one of the modern techniques for object identification. Yolov3 uses multi-class classification and binary cross-entropy to calculate the classification loss instead of mean square error. As shown in Figure 5, Yolov3 [36] uses logistic regression to foresee objects in three distinct scales (similar to FPN24) and the result for each bounding box. DarkNet-53 is used instead of DarkNet-19 as a new attribute extractor.

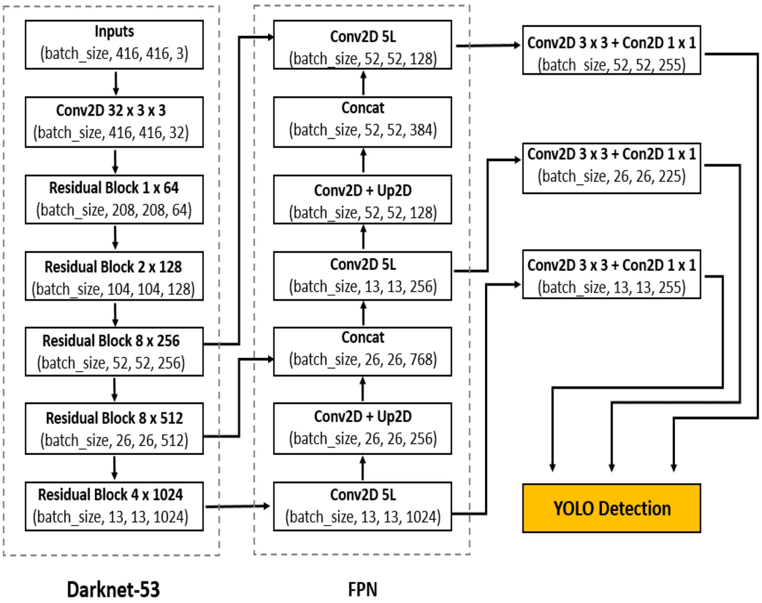

Inputs
(batch_size, 416, 416, 3)
Conv2D 32 x 3 x 3
(batch_size, 416, 416, 32)
Residual Block 1 x 64
(batch_size, 208, 208, 64)
Residual Block 2 x 128
(batch_size, 104, 104, 128)
Residual Block 8 x 256
(batch_size, 52, 52, 256)
Residual Block 8 x 512
(batch_size, 26, 26, 512)
Residual Block 4 x 1024
(batch_size, 13, 13, 1024)
Conv2D 5L
(batch_size, 52, 52, 128)
Concat
(batch_size, 52, 52, 384)
Conv2D + Up2D
(batch_size, 52, 52, 128)
Conv2D 5L
(batch_size, 13, 13, 256)
Concat
(batch_size, 26, 26, 768)
Conv2D + Up2D
(batch_size, 26, 26, 256)
Conv2D 5L
(batch_size, 13, 13, 1024)
Conv2D 3 x 3 + Con2D 1 x 1
(batch_size, 52, 52, 255)
Conv2D 3 x 3 + Con2D 1 x 1
(batch_size, 26, 26, 225)
Conv2D 3 x 3 + Con2D 1 x 1
(batch_size, 13, 13, 255)
YOLO Detection
Darknet-53
FPN

Figure 5. Architecture of Yolov3

The DarkNet-53 is a chain of 53 convolutional layers with a dimension of 1×1 followed by filters of size 3×3 with skip connections. Compared to ResNet-152, DarkNet-53 has low billion floating point operations (BFLOP), yet it is two times faster with classification performance than that ResNet-152. Yolov3 improves significantly on small object detection performance. Alexey *et al.* [37] recently introduced Yolov4, the next version of Yolov3. With comparable performance, it runs twice as fast as EfficientDet. In Yolov4, the average precision and frames per second were enhanced by 10% and 12%, respectively. The CSPDarkNet53 backbone, spatial pyramid pooling (SPP) extra block [38], path aggregation network (PANet) neck [39], and Yolov3 head make up the Yolov4 framework. With Mish [40], CSPDarkNet53 improves CNN's learning capacity. The SPP is used with the CSPDarkNet53 to considerably extend the receptive field, segregate the most relevant context features and reduce network operating speed to nearly nothing. Instead of the FPN in Yolov3, PANet is used in YOLOv4 to collect feature maps from various stages. Yolov4 allows for broader use of traditional GPUs while enhancing the performance of the classifier and detector. This study uses the label what you see (LWYS) procedure to recognize arecas in complicated environments using a modified Yolov3 model named Yolo-areca model. The addition of dense architecture [41] into Yolov3 to assist the reuse of attributes for more generalized areca identification and SPP application to lower the error and increase the accuracy are among the ideas put forward to reduce the disadvantages of deep learning and to make detectors intelligent as humans.

4.1. Yolo-areca model

The Yolo-areca model replaced the blocks 8*256 and 8*512 in Yolov3 shown in Figure 5 with a dense architecture for enhanced feature reuse and characterization. One convolutional layer and a 1*1 bottleneck layer were put together for every thick layer to enable accurate detection of tiny areca in various settings. The concatenated features of 26*26*768 and 13*13*384 in the FPN of the Yolo-areca model increase to 26*26*2816 and 13*13*1408 features. The Yolo-areca model was trained separately with segmented and unsegmented images to find a more accurate and faster Yolo-areca real-time detection model. Leaky rectified linear unit (ReLU) [42] with FDL*3 was used to activate the yield count of segmented images. All the layers of Yolov3 were pruned as follows: Yield count in unsegmented images is triggered using Mish [40] with FDL*1; Yield count in segmented images is activated using Mish with FDL*3, and SPP [38]. Mish, which is interpreted as $f(x) = x.tanh(c(x))$, where $c(x) = ln(1 + e^x)$, the softplus triggering function was found to beat ReLU. Introducing this activation function significantly improves every deep neural network's performance.

SPP [38] was launched following the last residual block to optimise the network topology. The feature extraction capability is strengthened as the convolutional layers deepen and the receptive field of a neuron increases. However, if the shape feature map of the arecas is obscured, the location details of the small areca become erroneous or even forgotten in some cases [43]. Because of more arecas in the image, there will be missed detections and lower accuracy. As a result, the SPP module can resolve the issue. The model was trained and tested on architecture with the following configuration: Intel Xeon(R) 64-bit 2.3 GHz CPU, 16 GB RAM, NVIDIA Tesla T4 GPU, CUDA v11.2, cuDNN v7.6.5. Images with a resolution of 416×416 pixels are fed into the model. Training loss is reduced when the learning rate is adjusted [30]. The rate of learning was adjusted to 0.001 for 4000 iterations with a maximum batch size of 6000 ripe and unripe areca. Batch and subdivision were set to 32 and 16 correspondingly to lower memory usage. Momentum and decay rates were set to 0.949 and 0.0005, respectively. Yolov4 has been trained using pre-trained weights.

4.2. Training

The model has been trained separately using 1017 segmented and 1017 unsegmented images; 80% were used for training, and the remaining 20% were used for validation. All the images are resized to 1920×1080 resolution. The label for each class and the coordinates of all the bounding boxes of ground truth images are required for training [29]–[31]. All ground truth bounding boxes were labelled using the graphical image annotation tool. Labelling each areca in an image is done by a bounding box using the LWYS approach shown in Figure 6. In particular, the bounding boxes for the heavily occluded arecas were drawn by a presumed shape using human intelligence. Three people verified the labelled images to confirm we have done annotations correctly. Four standard measures, namely: Precision (Pr), Recall (Re), F1-score (F1) and IoU given by (1) to (4), have been used to judge the accomplishment of the model. Table 3 summarizes the training performance. The higher values of the above measures indicate a more fantastic version of the model. The model performs better for segmented images because most unwanted background information has been eliminated during segmentation.

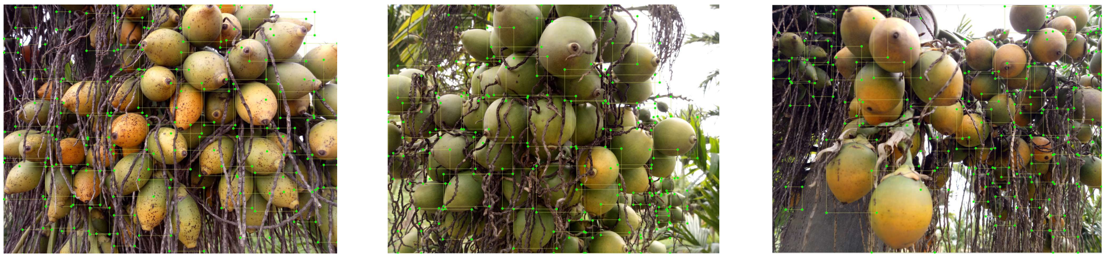


Figure 6. Sample annotated images

Table 3. Training performance

Method	Precision	Recall	F1-Score	IoU
Yolo-seg	93.0%	85.0%	89.0%	70.64%
Yolo-unseg	76.0%	96.0%	85.0%	58.10%

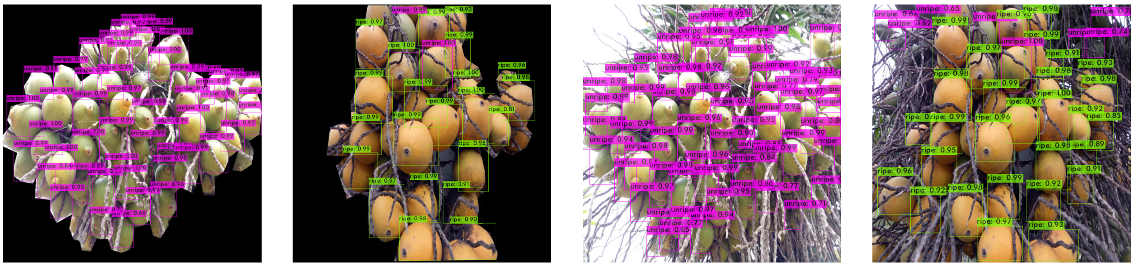
4.3. Results and comparision

The goal is to present an efficient and accurate technique for counting arecanuts in a bunch from an image acquired in field conditions. The trained model has been tested, and the mean absolute percentage error

(MAPE) on counting of areacnuts for 20 images has been determined to be 6.7% for unsegmented images and 5.3% for segmented images as shown in Table 4. Since there were no benchmarks for the comparison, methods that applied yield estimation on other crops were used as a measure for analogy. The model performance is better for segmented images because of the elimination of unwanted background information. The model performance is better than apple crop-load estimation [12], which shows 21.1% and [44], which shows 15% of MAPE. The Yolo-areca model achieves good results in comparison with other models. The appositeness of the approach is evidenced by its high performance across the entire data set that differs in terms of color, surrounding and resolution. The sample yield outputs of the model are shown in Figure 7. The model takes an average of 32 ms to count arecas in an image. Incorporating DenseNet and SPP increases the accuracy and lowers detection speed indicating the tradeoff between accuracy and speed.

Table 4. Error of identifying arecanuts

Unsegmented images						Segmented images					
Image	Actual	Detected	FP	FN	Error	Image	Actual	Detected	FP	FN	Error (%)
1	45	48	4	4	6.6	1	45	48	0	3	6.6
2	62	61	2	2	1.6	2	62	68	2	2	9.6
3	42	46	4	2	9.5	3	42	43	1	0	2.4
4	40	37	2	0	7.5	4	40	38	2	1	5.0
5	64	61	1	6	4.7	5	64	58	2	1	9.4
6	55	47	1	3	14.5	6	55	56	0	1	1.8
7	46	49	3	6	6.5	7	46	47	0	3	2.2
8	55	57	2	4	3.6	8	55	56	2	4	1.8
9	35	35	0	2	0.0	9	35	33	1	3	5.7
10	49	46	3	4	6.1	10	49	44	2	2	10.2
11	29	33	4	1	13.8	11	29	30	1	1	3.4
12	47	46	0	4	2.1	12	47	42	2	3	10.6
13	60	56	1	7	6.6	13	60	59	0	2	1.6
14	34	37	3	8	8.8	14	34	34	0	3	0
15	23	22	1	0	4.3	15	23	20	1	2	13.0
16	40	40	0	2	0.0	16	40	36	1	2	10.0
17	56	50	5	4	10.7	17	56	54	0	1	3.5
18	57	46	3	12	19.3	18	57	54	0	1	5.2
19	55	53	5	3	3.6	19	55	56	4	1	1.8
20	43	45	3	4	4.6	20	43	44	1	2	2.3
				MAPE	6.7					MAPE	5.3



Seg

Unseg

Figure 7. Representative results of Yield count

5. SUMMARY

This paper presents deep learning-based techniques for segmentation of arecanut bunch and counting of nuts in an arecanut bunch. U-Net squared model demonstrated greater segmentation performance across both ripe and unripe data sets that differ in terms of illumination, surrounding, density, shape and color. The model is trained with ripe arecanut and can generalize for unripe arecanut, which vary significantly in color. The outcome shows that the model outperforms compared to the benchmark. Yolo-areca model based on modified Yolov3 for counting nuts shows better accuracy of 94.7% when trained using segmented images compared to other methods applied to other commodities.

REFERENCES

[1] M. F. McCabe, R. Houborg, and A. Lucieer, “High-resolution sensing for precision agriculture: from Earth-observing satellites to unmanned aerial vehicles,” in *Remote Sensing for Agriculture, Ecosystems and Hydrology XVIII*, 2016, vol. 9998, pp. 346-355, doi: 10.1117/12.2241289.

[2] ”‘Vision 2050,” *Central Plantation Crops Research Institute*, Jul. 2015, [Online]. Available: https://cpcri.icar.gov.in/filemgr/webfs/publication/vision2050.pdf.

[3] B. Peng, L. Zhang, and D. Zhang, ”A survey of graph theoretical approaches to image segmentation,” *Pattern Recognition*, vol. 46, pp. 1020-1038, Mar. 2013, doi: 10.1016/j.patcog.2012.09.015.

[4] X. D. Bai, Z. G. Cao, Y. Wang, Z. H. Yu, X. F. Zhang, and C. N. li, ”Crop segmentation from images by morphology modeling in the CIE L*a*b* color space,” *Computers and Electronics in Agriculture*, vol. 99, pp. 21-34, Nov. 2013, doi: 10.1016/j.compag.2013.08.022.

[5] E. Hamuda, M. Glavin, and E. Jones, ”A survey of image processing techniques for plant extraction and segmentation in the field,” *Computers and Electronics in Agriculture*, vol. 125, pp. 184-199, Jul. 2016, doi: 10.1016/j.compag.2016.04.024.

[6] A. B. Payne, K. B. Walsh, P. P. Subedi, and D. Jarvis, ”Estimation of mango crop yield using image analysis-segmentation method,” *Computer and Electronics in Agriculture*, vol. 91, pp. 57-64, Feb. 2013, doi: 10.1016/j.compag.2012.11.009.

[7] Q. Wang, S. Nuske, M. Bergerman, and S. Singh, ”Automated crop yield estimation for apple orchards,” in *Experimental Robotics*, Heidelberg, Germany: Springer, 2013, vol.88, pp. 745-758, doi: 10.1007/978-3-319-00065-7_50.

[8] D. Font, M. Tresanchez, D. Martinez, J. Moreno, E. Clotet, and J. Palacin, ”Vineyard yield estimation based on the analysis of high resolution images obtained with artificial illumination at night,” *Sensors*, vol. 15, no. 4 pp. 8284-8301, Apr. 2015, doi: 10.3390/s150408284.

[9] D. Wang, D. He, H. Song, C. Liu, and H. Xiong, ”Combining SUN-based visual attention model and saliency contour algorithm for apple image segmentation,” *Multimedia Tools and Applications*, vol. 78, pp. 17391-17411, 2019, doi: 10.1007/s11042-018-7106-y.

[10] Y. Li, Z. Cao, H. Lu, Y. Xiao, Y. Zhu, and A. B. Cremers, ”In-field cotton detection via region-based semantic image segmentation,” *Computers and Electronics in Agriculture*, vol. 127, pp. 475-486, Sep. 2016, doi: 10.1016/j.compag.2016.07.006.

[11] D. Wang, Z. Cao, Y. Xiao, Y. Li, and Y. Zhu, ”Region-based colour modelling for joint crop and maize tassel segmentation,” *Biosystems Engineering*, vol. 147, pp. 139-150, Jul. 2016, doi: 10.1016/j.biosystemseng.2016.04.007.

[12] A. Gongal, A. Silwal, S. Amatya, M. Karkee, Q. Zhang, and K. Lewis, ”Apple crop-load estimation with over-the-row machine vision system,” *Computers and Electronics in Agriculture*, vol. 120, pp. 26-35, Jan. 2016, doi: 10.1016/j.compag.2015.10.022.

[13] U.-O. Dorj, M. Lee, and S. Yun, ”An yield estimation in citrus orchards via fruit detection and counting using image processing,” *Computers and Electronics in Agriculture*, vol. 140, pp. 103-112, Aug. 2017, doi: 10.1016/j.compag.2017.05.019.

[14] M. N. Reza, I. S. Na, S. W. Baek, and K.-H. Lee, ”Rice yield estimation based on K-means clustering with graph-cut segmetation using low-altitude UAV images,” *Biosystems Engineering*, vol. 177, pp. 109-121, Jan. 2019, doi: 10.1016/j.biosystemseng.2018.09.014.

[15] S. Siddesha, S. K. Niranjan, and V. N. M. Aradhya, ”A study of different color segmentation techniques for crop bunch in arecanut,” in *Handbook of Research on Advanced Hybrid Intelligent Techniques and Applications*, Hershey, Pennsylvania, USA: IGI Global, 2016, pp. 1-28, doi: 10.4018/978-1-4666-9474-3.ch001.

[16] R. Dhanesha and C. L. S. Naika, ”A novel approach for segmentation of arecanut bunches using active contouring,” in *Integrated Intelligent Computing, Communication and Security, Studies in Computational Intelligence*, Singapore: Springer, 2019, vol. 771, pp. 677-682, doi: 10.1007/978-981-10-8797-4_69.

[17] R. Dhanesha, C. L. S. Naika, and Y. Kantharaj, ”Segmentation of arecanut bunches using YCgCr color model,” in *2019 1st International Conference on Advances in Information Technology (ICAIT)*, 2019, pp. 50-53, doi: 10.1109/ICAIT47043.2019.8987431.

[18] R. Dhanesha, D. K. Umesha, C. L. S. Naika, and G. N. Girish, ”Segmentation of arecanut bunches: a comparative study of different color models,” in *2021 IEEE Mysore Sub Section International Conference (MysuruCon)*, 2021, pp. 752-758, doi: 10.1109/MysuruCon52639.2021.9641680.

[19] A. C. Anitha, R. Dhanesha, and C. L. S. Naika, ”Arecanut bunch segmentation using deep learning techniques,” *International Journal of Circuits, Systems and Signal Processing*, vol. 16, pp. 1064-1073, 2022, doi: 10.46300/9106.2022.16.129.

[20] T. P. Saarela and M. S. Landy, ”Combination of texture and color cues in visual segmentation,” *Vision Research*, vol. 58, pp. 56-67, Apr. 2012, doi: 10.1016/j.visres.2012.01.019.

[21] D. S. Guru and H. G. Shivamurthy, ”Segmentation of mango region from mango tree image,” in *Mining Intelligence and Knowledge Exploration*, Cham, Switzerland: Springer, 2013, vol. 8284, pp. 201-211, doi: 10.1007/978-3-319-03844-5_21.

[22] R. Perez-Zavala, M. Torres-Torriti, F. A. Cheein, and G. Troni, ”A pattern recognition strategy for visual grape bunch detection in vineyards,” *Computers and Electronics in Agriculture*, vol. 151, pp. 136-149, Aug. 2018, doi: 10.1016/j.compag.2018.05.019.

[23] U. Verma, F. Rossant, and I. Bloch, ”Segmentation and size estimation of tomatoes from sequences of paired images,” *EURASIP Journal on Image and Video Processing*, vol. 33, pp. 1-23, 2015, doi: 10.1186/s13640-015-0087-0.

[24] A. Rahman and A. Hellicar, ”Identification of mature grape bunches using image processing and computational intelligence methods,” in *Proceedings of IEEE Symposium on Computational Intelligence for Multimedia, Signal and Vision Processing (CIMSIVP)*, 2014, pp. 1-6, doi: 10.1109/CIMSIVP.2014.7013272.

[25] S. Bargoti and J. P. Underwood, ”Image segmentation for fruit detection and yield estimation in apple orchards,” *Journal of Field Robotics*, Feb. 2017, doi: 10.1002/rob.21699.

[26] C. Hung, J. Nieto, Z. Taylor, J. Underwood, and S. Sukkarieh, ”Orchard fruit segmentation using multi-spectral feature learning,” in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, vol. 77, pp. 5314-5320, doi: 10.1109/IROS.2013.6697125.

[27] R. Kestur, A. Meduri, and O. Narasipura, ”MangoNet: A deep semantic segmentation architecture for a method to detect and count mangoes in an open field,” *Engineering Applications of Artificial Intelligence*, vol. 77, pp. 59-69, Jan. 2019, doi: 10.1016/j.engappai.2018.09.011.

[28] S. Xie and Z. Tu, ”Holistically-Nested edge detection,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1395-1403, doi: 10.1109/ICCV.2015.164.

[29] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, ”You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[30] J. Redmon and A. Farhadi, ”YOLO9000: Better, Faster, Stronger,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517-6525, doi: 10.1109/CVPR.2017.690.

[31] J. Redmon and A. Farhadi, ”Yolov3: An incremental improvement,” *arXiv*, vol. 1804.02767, 2018, doi: 10.48550/arXiv.1804.02767.

[32] R. Girshick, ”Fast R-CNN,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.

[33] S. Ren, K. He, R. Girshick, and J. Sun, ”Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis Machine Intelligence*, vol. 39, no. 6, pp. 1137-1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[34] S. Ioffe and C. Szegedy, ”Batch normalization: accelerating deep network training by reducing internal covariate shift,” in *Proceedings of the 32nd International Conference on Machine Learning*, Jul. 2015, vol. 37, pp. 448-456, doi: 10.1145/3527155.

[35] K. He, X. Zhang, S. Ren, and J. Sun, ”Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[36] M. O. Lawal, ”Tomato detection based on modified Yolov3 framework,” *Scientific Reports* vol. 11, no. 1447, pp. 1-11, 2021, doi: 10.1038/s41598-021-81216-5.

[37] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, ”YOLOv4: Optimal speed and accuracy of object detection,” *arXiv*, 2020, doi: 10.48550/arXiv.2004.10934.

[38] K. He, X. Zhang, S. Ren, and J. Sun, ”Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904-1916, Sep. 2015, doi: 10.1109/TPAMI.2015.2389824.

[39] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, ”Path aggregation network for instance segmentation,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8759-8768, doi: 10.1109/CVPR.2018.00913.

[40] M. D. Mish, ”A self-regularized nonmonotonic neural activation function,” *arXiv*, Aug. 2020, doi: 10.48550/arXiv.1908.08681.

[41] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, ”Densely connected convolutional networks,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261-2269, doi: 10.1109/CVPR.2017.243.

[42] V. Nair and G. E. Hinton, ”Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, Jun. 2010 pp. 807-814.

[43] O. Russakovsky *et al.*, ”ImageNet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, pp. 211-252, 2015, doi: 10.1007/s11263-015-0816-y.

[44] O. Cohen, R. Linker, and A. Naor, ”Estimation of the number of apples in color images recorded in orchards,” *Computer and Computing Technologies in Agriculture, IFIPAICT*, Heidelberg, Germany: Springer, 2010, vol. 344, pp. 630-642, doi: 10.1007/978-3-642-18333-1_77.

BIOGRAPHIES OF AUTHORS

Anitha Arekattedoddi Chikkalingaiah holds a Master of Technology degree from Visvesvaraya Technological University, Karnataka, India in 2008. She also received her B.E. (Computer Science and Engineering) from Visvesvaraya Technological University, Karnataka, India in 2005. She is currently an assistant professor at Computer Science and Engineering Department in Government S.K.S.J. Technology Institute, Bengaluru, India. She is a research fellow in Department of Studies in Computer Science and Engineering, University B.D.T College of Engineering, Davanagere, Karnataka, India. Her research includes pattern recognition, image processing, machine learning, and graph theory. She can be contacted at email: anithakrishna2008@gmail.com.

RudraNaik Dhanesha received Graduation (B.E.) in 2007 from Computer Science and Engineering at Visvesvaraya Technological University, Belgavi and Post Graduation (M.Tech.) in 2010 from Computer Science and Engineering at UBDTCE, Davangere, affiliated to Kuvempu University, Karnataka. He also pursuing Ph.D. in Computer Science and Engineering, UBDTCE research center affiliated to VTU, Belagavi, Karnataka, India. He is currently working as Assistant Professor in DOS in Computer Science, Davangere University, Davangere, Karnataka, India. His research interests include image processing and computer vision, deep learning, IoT, and pattern recognition. He can be contacted at email: dhanesh025@gmail.com.

Shrinivasa Naika Chikkathore Palya Laxmana obtained his Graduation (B.E.), Post Graduation (M.E.) from University Visvesvaraya College of Engineering, Bangalore University. Received Ph.D. from Indian Institute of Technology -Guwahati, Assam, India. He is currently working as a Professor, DOS in Computer Science and Engineering, UBDT College of Engineering, Davangere, Karnataka, India. He published many papers in National and International journals, conferences. His research interests include image processing and computer vision, deep learning, data science, and pattern recognition. He can be contacted at email: naika2k6@gmail.com.

Krishna Alabujanahalli Neelegowda obtained his Graduation(B.E.), Post Graduation(M.E.) from University Visvesvaraya College of Engineering, Bangalore University. Received Ph.D. from Visvesvaraya Technological University, Karnataka, India. He is currently working as a Professor and Head of the Department, Department of Computer Science and Engineering, SJB Institute of Technology, Bengaluru, India. He published many papers in National and International journals, conferences. His research interests include image processing and computer vision, deep learning, data science, and pattern recognition. He can be contacted at email: drkrishnaan@gmail.com.

Anirudh Mangala Puttaswamy obtained his bachelor's degree in computer science and engineering from SJB Institute of Technology, Bengaluru, India in 2022. He is currently pursuing his master's in data science at Royal Melbourne Institute of Technology, Australia. He can be contacted at email: anirudhmp41@gmail.com or s3993305@student.rmit.edu.au.

Pushkar Ayengar obtained his bachelor's degree in computer science and engineering from SJB Institute of Technology Bengaluru, India in 2022. He currently works as a Data Security Engineer at BETSOL and his research interest includes are artificial intelligence, system security, and networks. He can be contacted at email: ayengarpushkar@gmail.com.