COCONUT MATURITY LEVEL DETECTION SYSTEM

Sinchan Kunder [1], Prof. Sowmya M S [2]

[1] Student, Department of MCA, Bangalore Institute of Technology, Karnataka, India

[2] Professor, Department of MCA, Bangalore Institute of Technology, Karnataka, India

---***---

Abstract:

Coconuts are very nutritious but very difficult to carve. We, therefore, present an approach to defining a deep source of SLOV2 learning that can segment coconuts automatically. Overview SOLOv2 is the state-of-the-art one-stage detector that performs the task of precise real-time instance segmentation. Materials and Methods We had 344 images for training and 98 for validation, while the test was on 53 novel classes using Python 3 on Google Colab using a GPU, T4. The tests showed that SOLOv2 delivers very good performance for the recognition of ripeness in coconut. This is evidenced by the Average Precision (AP) of 63.80%, the Average Recall of 82.10%, and the overall mAP of 63.80%, yet without symptoms of overfitting. This further enhances matters of efficiency and costs in processing coconuts and benefits farmers and processors by reducing expenses incurred on labor.

Keywords - Coconut, Effieciency, Instance segmentation, SOLOv2, Performance metrics (Average Precision, Mean Recall, mAP)

1. INTRODUCTION

The coconut sector in India is intricately linked with the agricultural scenario of India, providing support for the livelihood of millions of farmers in addition to its being an important contributor towards the economy. The problem lies in the correct assessment of coconut maturity stages. Traditional methods used to measure coconut maturity are not only very time-consuming but also highly subjective and lead to inconsistent and inaccurate results. Imagine the poor farmer spending many hours under the scorching sun, manually inspecting each coconut for its readiness to harvest, and feel confused. Most of these inconsistencies result in poor quality and reduced yield of coconut products, affecting farmers' income and supply chain in large.

Overriding these challenges, we now introduce the Coconut Maturity Level Detection System, which Innovative technology poised to revolutionize the way coconut maturity assessment has been done so far. The system is engineered with state-of-the-art sensors that, alongside machine learning algorithms, turn the detection of coconuts' maturity level into a much faster and more effective process. By this innovation, in just a fraction of the time, farmers will be able to get the results with an enviable level of accuracy, thereby ascertaining the harvest of coconuts at their optimal maturity. State-of-the-art technology-embedded Coconut Maturity Level Detection System for increased efficiency and effectiveness of coconut harvesting to enhance the quality of the final products. The paradigm shift that the system has brought with it offers farmers reliable tools to lessen guesswork and labor associated with traditional methods. It is one sure step toward modernization of agricultural practices for benefiting the entire coconut industry in India.

Coconut ripeness can be categorized into three stages. These stages are essential for determining optimal harvest time. The first stage green ripeness features green husk indicating immaturity. It requires weeks to months to mature fully. The second stage, yellow ripeness shows initial signs of germination and husk rotting. This stage suggests start of maturation but still unsuitable for harvest.

The final stage is brown ripeness. Here the coconut's outer shell turns brown. This signifies full maturity and readiness for harvest. Understanding these stages is vital for farmers. It helps them harvest coconuts at peak quality. Thus enhancing yield and profitability. Recognizing these ripeness stages allows better crop management and improved harvesting practices. It leads to maximized coconut utilization, contributing to sustainable farming and increased earnings.

Contributions:

Our project leverages the SOLOv2 model to revolutionize coconut ripeness detection by overcoming limitations of traditional methods. These methods rely on subjective color analysis. Unlike these methods SOLOv2 employs advanced AI to accurately identify and segment individual coconuts. This ensures precise maturity recognition for optimal harvesting. By automating ripeness inspection, our system minimizes estimation errors and uncertainty. This yields more reliable results. It also informs better harvesting decisions. The system's focus on each coconut within a grove allows for enhanced production of high-quality crops. This increases yield and reduces costs. Efficient and scalable SOLOv2 model significantly cuts time and labor. It improves quality assurance practices across various project scales. We began by collecting and manually tagging dataset of 500 RGB images. Annotated using Roboflow. To train SOLOv2 model. Post-training we validated model's accuracy. Efficiency was also checked, refining it for better performance in diverse scenarios Our goal is to create non-intrusive automated system. One that accurately identifies coconut ripeness stages enhancing reliability and intelligence of ripeness detection in the coconut industry.

2. RELATED WORK

Increasing interest in automation for coconut fruit detection on trees has resulted in investigation into state-of-the-art computer vision and deep learning enablers of increased accuracy, efficiency, and scalability. There have been numerous seminal articles in the area. Novelero et al. [1] contributed a state-of-the-art UAV-based system that utilized convolutional neural networks to identify ripe coconuts with high accuracy on trees and thus support harvesting operations. Another well-known work by [2] used an end-to-end object detector called the Faster R-CNN model, which successfully classified coconut ripeness amidst different backgrounds, demonstrating good performance in the most difficult environmental conditions. Zheng and colleagues focused on the use of detailed satellite imagery to detect coconut trees on Tenarunga Island, showcasing applicability in deep learning for remote sensing applications. Addressing another challenge in a different area, the problem of non-destructive ripeness assessment is tackled by, who proposed a new technique using neural networks trained with sound vibrations from coconuts at levels progressing in ripeness that could provide a non-invasive alternative to traditional approaches. In robotic assistance, PSO has been used [5] to ensure that robots can identify coconuts from trees from their images, thus implying efficient self-navigation and localization. Xinlong et al. [6] proposed the SOLO framework, where for each instance of the object in the image, a mask is generated to increase detection accuracy, hence increasing segmentation accuracy. Advances in instance segmentation include the creation of YOLACT and Mask Scoring R-CNN, methods that implement Faster R-CNN 's architecture, which attains accurate object instance segmentation together with scoring on the quality of the masks. These innovations are essential in automating fruit harvesting processes and quality inspection tasks. Other innovations include that by Javel et al. [11], who used fuzzy logic to classify the several maturity stages of coconut fruit into color, weight, size, and texture attributes and therefore gave a robust framework for agricultural assessment. Gad et al. [14] contributed another by introducing mean average precision as an evaluation metric for object detection models, hence able to perform comprehensive performance analysis across different frameworks of detection. Lin et al. [15] proposed Feature Pyramid Networks, which augmented the object detection across a wide range of scales by modeling semantic information through a pyramid structure of feature maps. Hariharan \et al. [16] introduced a unified framework for both detection and semantic segmentation that empowers high-quality segmentation without requiring a bounding box alone. Instance-Sensitive FCN (ISFCN) [17] extended the segmentation accuracy by incorporating instance-level information for the more comprehensive representation of object categories and segmentation masks. Double-head R-CNN [18], M2Det [19], and Mask R-CNN [20] further generalized the functionality related to object detection and instance segmentation with enhanced robust performance. Hamid et al. [21] introduced the metric and loss function of the Generalized Intersection over Union, which refines the accuracy of bounding box regression by comprehensive measures of overlap between predicted and ground truth boxes

All these developments represent one single vision on how to automatically detect and evaluate coconut fruit from all angles, using state-of-the-art deep learning techniques on different imaging modalities and scenarios of applications.

3. PROBLEM STATEMENT

Manual assessment of coconut maturity is labour-intensive and thus induces errors; therefore, it suboptimizes harvesting practice, which reduces crop quality. Current practices are less efficient and reliable for extensive coconut plantations, needing an automatic classification of coconuts by maturity level. On this note, there is a need to develop a Coconut Maturity Level Detection System to off-tail all these challenges for an optimal harvesting operation that ensures maximum productivity and profitability.

4. PROPOSED SYSTEM

Figure 1: Sequential steps in the training and testing of a model, more so SOLOv2, for shell detection. The first step involved data collection, followed by preprocessing to have the data ready for training. After data preparation or annotation, "RoboFlow" or any other tool is used to set areas of interest manually and label them. These are maintained in JSON format, which stipulates object classes, their sizes, shapes, and locations in the image.

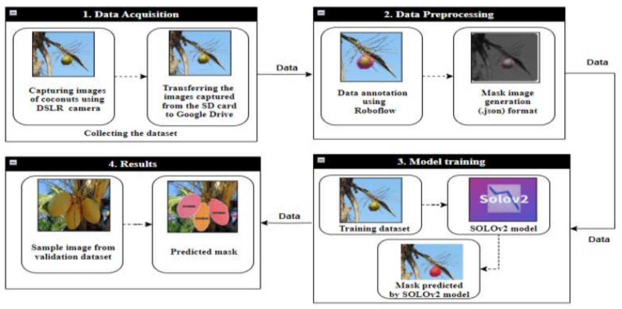
1. Data Acquisition
Capturing images of coconuts using DSLR camera
Transferring the images captured from the SD card to Google Drive
Collecting the dataset
Data
2. Data Preprocessing
Data annotation using Roboflow
Mask image generation (.json) format
Data
3. Model training
Training dataset
Solov2
SOLOv2 model
Mask predicted by SOLOv2 model
Data
Data
4. Results
Sample image from validation dataset
Predicted mask

Figure-1: Workflow

The instance mask images are annotated, and, further, mask images are generated where each group of objects in that particular image is assigned a different mask indicating the pixels pertaining to that class of objects. Using this model, the learning would be proper in respect of classifying and segmenting the objects.

SOLOv2 is used here. During training, the model learns to predict accurate masks on unseen images. Another validation dataset is used during training for the evaluation of model performance so that necessary adjustments can be done.

5. METHODOLOGY

A. Data Acquisition

1. Capturing the images.
2. Collecting the images.
3. Preparing the dataset.

1. Capturing the Images:

The images of the coconuts on the tree have been captured by visiting different geographical locations such as Kundapur, K.R Circle near Cubbon Park, Udupi and Mangalore by using the DSLR camera. The images captured were stored in the SD card. A total of 500 images were captured.

2. Collecting the Images:

A few images of the coconuts on the tree were also collected from the various websites (Like Pinterest) on the internet. A total of 40 images were collected.

3. Preparing the Dataset:

The images captured and stored in the SD card and the images collected were merged into a single folder forming a “coconut dataset” and then loaded into the Google Drive platform.

B. Data Preprocessing

1. Loading Dataset

2. Data Annotation

3. Splitting the Dataset

1. Loading the Dataset:

The dataset utilized for this project is specifically focused on coconuts on trees, containing a total of 540 images. The dataset located in the Google drive is loaded into the “RoboFlow” tool.

2. Data Annotation:

Data annotation is a process where a human data annotator goes through a raw data set to add categories, labels, and others to images so that machines are able to read and learn from images. Image annotation is focused on labelling an image with metadata, keywords, and other descriptors detailing of an image in relation to other image descriptors.

RoboFlow is an open-source annotation tool. It provides an annotation tool and This enables users to annotate objects manually within images or videos with bounding box annotation, polygon annotation, semantic segmentation, and smart polygons. This project made use of polygon annotation. In our project 3 classes namely tender coconut, mature coconut and dry coconut.

JSON, or JavaScript Object Notation, is a data structuring format widely used by web applications for communication. It serves as a replacement for the XML data exchange format and offers a simpler way to structure data. JSON supports data structures such as arrays and objects, and JSON documents are executed quickly on the server.

A dataset version is assigned, and the exported dataset can either be downloaded as a .zip file or downloaded by using a curl link from the command-line interface. The contents of the .zip file are the train, test, and val folders with images and JSON files.

3. Split the Dataset into Training and Testing:

The entire dataset is divided into a training dataset and a testing dataset with a ratio of 70:30. It uses the training data to train, and learn the hidden features of the model for improvement The pictures. At each point in time, the same training data is fed to the Fully Convolutional Network (FCN). The architecture is repeated and the model continues to learn the features of the image dataset.

The coconut of the training data consists of a variety of images, in order to train the model on all of them situations and any unseen data samples that may occur in the future can be predicted. Once trained, the set of test data is used. It provides unbiased final model performance metric in precision, IoU, precision, recall, mAP.

C. Model Training

1. Data Preparation:

First, one needs to prepare the data for the training of SOLOv2. Specifically, this means splitting the dataset into training, validation, and test sets. The training set is used to train the model, the one validation set is used for tuning the hyperparameters and assessment of a model's performance during the training dataset is used during model training, and the test dataset contains what is used to evaluate the final performance of the model.

2. SOLOv2 Architecture:

Basically, the SOLOv2 model is an instance segmentation model relying on a fully convolutional network with one pass. Its multi-level FPN acts as its backbone for capturing hierarchical features with different scales. These features will be fed to a series of SOLOV2Heads, each predicting object instances of a certain scale, and the SOLOV2Head cascades and densely generates object predictions from feature maps with semantic segmentation masks and objectness score predictions.

They use a combination of semantic and mask branches that enables high-quality object mask instance segmentation in an end-to-end manner.

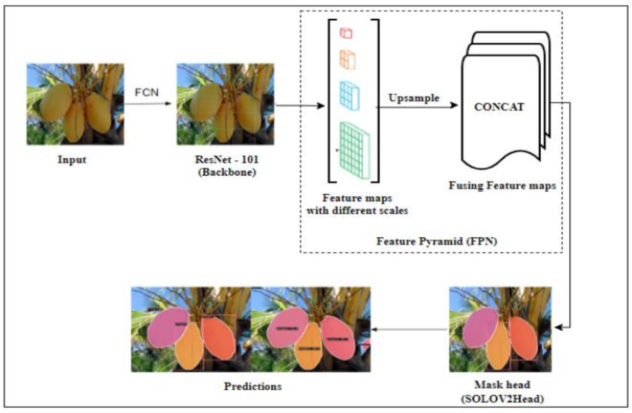

Figure -2: Solov2 Architecture

First of all, this model starts with input layer, which takes an image of size (256,256,3). The image it is processed by the pre-trained ResNet 101, which is an FCN. The backbone is composed of four stages, each of which produces feature maps at different scales. All four stages (0, 1, 2, 3) will be used to obtain the output feature maps.

It is the standard basic module applied in almost all state-of-the-art object detectors: the feature pyramid network. The FPN combines the feature maps of different scales—256, 512, 1024, and 2048—in a feature pyramid to capture both high-level semantic information and fine-grained details. Specifically, this is achieved via a top-down pathway that refines feature maps at each scale using the immediately higher-resolution feature map higher to lower. The lateral connections establish communication between different levels of the network. Up sample the higher resolution feature maps and fuse them with corresponding lower resolution feature maps using element-wise addition producing the 5 output feature maps.

Output from feature pyramid is used as input for mask head which will be responsible for generating instance segmentation masks. The mask head module is of type SOLOV2Head, it Thus, it allows generating a mask for each object that is detected. Moreover, it takes intermediate feature maps as input.

Part	**Type**	**Filters**	**Filter Size**	**Output Shape**	**Activation Function**
Input	-	-	-	(256,256, 3)	-
ResNet101	Pre-trained FCN	-	-	(256,256, 256)	-
FPN	Module	-	-	(256,256, 256)	-
FPN-output 0	Convolutional Layer	256	1x1	(256,256, 256)	Sigmoid
FPN-output 1	Convolutional Layer	256	3x3	(128,128, 256)	Sigmoid
FPN-output 2	Convolutional	256	3x3	(64,64,25	Sigmoid

Table 1. Solov2 Model Architecture

In this case, there are 4 convolutional layers, which are stacked in the SOLOV2Head module. Further, masks are to be generated on the basis of the number of classes, which in this case is 3, and features extracted from intermediate feature maps. The Mask Head generates masks with the same spatial resolution as the input feature maps. It further uses DiceLoss for mask segmentation and FocalLoss for classification with respective loss functions.

Finally, the output of the Mask Head is the final set of instance segmentation predictions for coconut images. These include the instance masks, class labels, and bounding boxes for the detected coconuts in the input images.

D. Results

During training, this model processes images in batches; that is, it predicts segmentation maps in its forward pass and calculates gradients in the backward pass for updating its parameters with the help of some optimizer, optimizing to reduce loss. Results of the trained model with respect to one sample image from the dataset validate segmentation performance. These methods measure segmentation metrics quantifying the accuracy with regard to Intersection over Union or IOU. It scored an IOU of 0.811 over both the training and validation data sets, hence very strong in segmentation accuracy and reliable in object classification.

6. RESULTS AND EVALUATION

To make the process of evaluation more enriched, the experiment used various levels of controlled parameter variables. The primary evaluation indicators included mean Average Precision (mAP), AP—Average Precision AR—Average Recall, IoU—Intersection over Union thresholds, and Precision-Recall curves.

1. mean Average Precision (mAP): This metric is used to measure the accuracy for object detection and localization. It is the average of the sum of Average precision by the total number of classes.

Mathematically mAP is computed as:

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

Where:

- AP is the Average Precision values. i
- n is the total number of classes.

2. Average Precision (at different intersection-over-union (IoU) thresholds): It evaluates the object detection precision at different IoU thresholds, which measure the overlap between the predicted bounding boxes and the ground truth bounding boxes. The AP at different IoU thresholds is obtained by calculation of AP for each threshold of IoU separately.

The formula for calculating AP at a specific IoU threshold is as follows:

$$AP(IoU) = \int_0^1 p(r)\, dr$$

Where:

- p(r) represents the interpolated precision at a specific recall value r.
- The integral is taken over the range of recall values from 0 to 1.

3. Average Precision (at varying IoU thresholds): It measures the model's ability to detect objects accurately at various overlap levels. This is with the ground-truth bounding boxes.

The formula for calculating AR at different IoU thresholds is as follows:

$$AR(IoU) = \frac{1}{n}\sum(Recall(IoU))$$

Where:

- n is the total number of IoU thresholds.
- Recall(IoU) represents the recall at a specific IoU threshold.

4. Precision-Recall Curves: These are graphical representations that this will demonstrate the precision-recall trade-off for a given classifier or model.

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

Where:

- TP is True Positives.
- FP is False Positives.

The Recall is calculated as follows:

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

Where:

- TP is True Positives.
- FN is False Negatives.

The various IoU threshold average precision values of the SOLOv2 model. For the IOU threshold range from 0.50 to 0.95 and for all areas, it resulted in an average precision of 0.557. This means that the model is capable of detecting and classifying different objects irrespective of their sizes and shapes. For assessing the IOU threshold value of 0.50, the average precision for all areas increased to 0.717 and hence overall better at object detection than at IOU thresholds, which are relatively lax. At an IOU threshold of 0.75, all areas' average precision was 0.614, therefore relatively comparable to the latter, but a bit less accurate since the latter is the higher The performance can be broken down to different areas. And for the same IOU at the threshold, it did an average precision of 0.142, 0.392, and 0.728 for small, medium, and large objects, respectively, of all areas. It is able to detect items within the IOU threshold from 0.5 through 0.95. These findings suggest this model has performance varying with perspective, due to scene and object sizes, yet works fairly well on the detection and classification of larger objects.

The average recall values of the SOLOv2 model at different IoU thresholds. For IoU threshold range from 0.5 up to 0.95, and considering areas of all instances, this returned an average recall of 0.821, indicating that this model can hold most of the true positive instances with high accuracy regarding varying size and shapes. Another very interesting thing to point out here is that for all IoUs across all areas considered remains to be at thresholds within the sprawl. However, considering different areas separately, the model's average recall in this case was 0.223 for small objects, 0.731 for medium-sized, and large objects it performed strongly at 0.911. This result could indicate that it is extremely good at capturing and identifying large objects but maintains a good level of recall for smaller and medium-sized objects.

Different IoU thresholds and their corresponding mAP values of the SOLOv2 model. For the IoU threshold range from 0.50 to 0.95, considering all areas, it turned in an mAP value of 0.5570. It gives an idea about the overall performance of the model in detecting and classifying the objects accurately, irrespective of sizes and shapes. If the model's performance is checked on an IoU threshold of 0.50, it increased to 0.7170 at an IoU threshold of 0.5, which means better precision and recall in the detection of objects. At a threshold value of 0.75 for IoU, the mAP dropped a little; the value was 0.6140, thus showing this model is less accurate for a higher threshold of IoU. Moving into details of different areas separately, small objects had an mAP value of 0.1420, the medium-sized objects had an mAP value of 0.3920, and large objects had a very impressive mAP value of 0.6280. Results are within the threshold range of the IoU from 0.50 to 0.95. These results show quite varying performance across different areas and sizes of objects, leaning toward better detection and classification of large objects.

The varying average precision, average recall and mAP values in the table is attributed to the differences in Area (small, medium, and large) and the chosen IoU thresholds for evaluation.

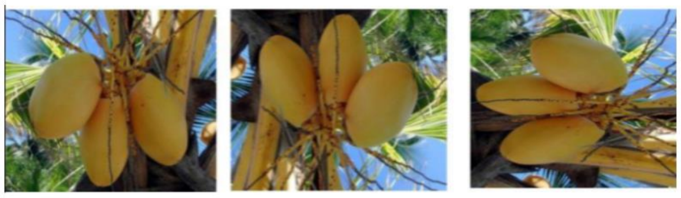
Figure 3: Input images at different angles

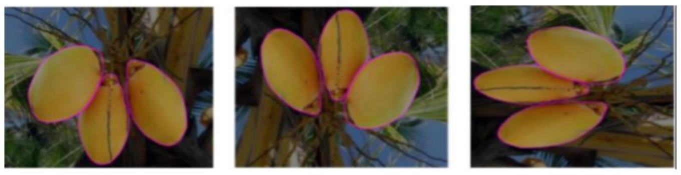
Figure 4: Annotated images at different angles

Figure. 3 are the raw images (not a model of it) taken from different angles by DSLR camera, In the Fig. Figure 3: The image at the left is that of angle =0 in Fig. The next middle image was taken at 180 degrees. 90 degrees, right (third image from R).

Figure 4: Image with polygon annotation, the annotation are done using online tool RoboFlow.

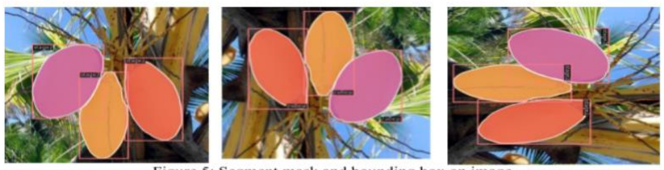
Figure 5: Segment mask and bounding box on image

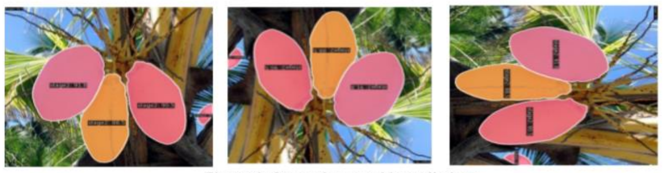
Figure 6: Output image with predictions

Figure 5. The images are shown in fig, 5 using a segment mask to highlight each stage by color (class label located near object indicates the associated phase and bounding box is drawn around objects)

Figure 6. Output images; Prediction of the ripeness stage for each object is along with confidence level, set by model

7. CONCLUSION

The main focus of the Coconut Fruit Ripeness Stage Detection Project is to build a system which can be used for detecting at which stage (ripening) coconut fruits are. The project was able to identify three ripeness stages using the autoencoder, namely Stage 1 (unripe), Stage2(partially ripe) and STAGE3(fully ripe). The project reached these values because it had based on ResNet101 as backbone, feature pyramid (FPN) and maskhead of the SOLOv2 model which is said to be able to analyze images in great detail by accurately labeling each object. The project started by gathering an extensive and various dataset of coconut fruit images with ambiguous kinds. Using RoboFlow each image was annotated and labeled as to what stage of ripeness the coconut fruit is. This annotated dataset was then used to train the SOLOv2 model.Other evaluation metrics used to study the performance of the proposed model include precision, recall, and mAP. The proposed approach achieved an average precision about 63.80%, an average recall about 81.10%, and an mAP value about 63.80% for the IoU threshold 0.50:0.95 and area Large. The developed Coconut Fruit Ripeness Stage Detection System makes the process of ripeness assessment fully automated and provides an important tool for farmers and industry players. This system shall help them in make informed decisions on harvesting and post-harvest practices that can improve the coconut industry's productivity.

8. REFERENCES

1. Novelero, Jilbert M and Cruz, Jennifer C Dela "On-tree Mature Coconut Fruit Detection based on Deep Learning using UAV images", pp. 494-499, 2022.
2. Parvathi, Subramanian and Selvi, Sankar Tamil “Detection of maturity stages of coconuts in complex background using Faster R-CNN model”,journal on Biosystems Engineering, Volume 202, pp.119-132, 2021.
3. Zheng, Juepeng and Wu, Wenzhao and Yu, Le and Fu, Haohuan "Coconut Trees Detection on the Tenarunga Using High - Resolution Satellite Images and Deep Learning.", pp. 6512-6515, 2021.
4. Fadchar, Nemilyn A and Cruz, Jennifer C Dela "A Non-Destructive Approach of Young Coconut Maturity Detection using Acoustic Vibration and Neural Network.”, pp.136 -137, 2020.
5. Junaedy, Alfin and Sulistijono, Indra Adji and Hanafi, Nofria "Particle swarm optimization for coconut detection in a coconut tree plucking robot." , pp.182-187, 2017.
6. Wang, Xinlong and Kong, Tao and Shen, Chunhua and Jiang, Yuning and Li, Lei “Solo: Segmenting objects by locations”, pp.649-665, 2020.
7. Wang, Xinlong and Zhang, Rufeng and Kong, Tao and Li, Lei and Shen, Chunhua “Solov2: Dynamic and fast instance segmentation” ,journal on Advances in Neural information processing systems, vol. 33, pp. 17721-17732, 2020.
8. Bolya, Daniel and Zhou, Chong and Xiao, Fanyi and Lee ,Yong Jae “Yolact: Real-time instance segmentation” , pp.9157-9166, 2019.
9. Cheng, T., Wang, X., Huang, L., Liu, W. ”Boundary-Preserving Mask RCNN”, 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16, pp. 660-676, 2020.
10. Wan, Shaohua and Goudos, Sotirios “ Faster R-CNN for multi-class fruit detection using a robotic vision

system " , journal on Computer Networks ,vol.168 , pp.107036, 2020.

11. Javel, Irister M and Bandala, Argel A and Salvador, Rodolfo C and Bedruz, Rhen Anjerome R and Dadios, Elmer P and Vicerra, Ryan Rhay P "Coconut fruit maturity classification using fuzzy logic", pp.1-6, 2018.
12. Nepal, Upesh and Eslamiat, Hossein "Comparing YOLOv3, YOLOv4 and SOLOV2 for autonomous landing spot detection in faulty UAVs.", journal on Sensors, vol.32, no.2, pp.464, 2022.
13. Thuan, Do "Evolution of Yolo algorithm and SOLOV2: The State-of-the-Art object detection algorithm.", 2021.
14. Gad, Ahmed Fawzy "Evaluating object detection models using mean average precision (mAP).", journal on PaperspaceBlog, 2020.
15. Lin, Tsung-Yi and Doll{\'a}r, Piotr and Girshick, Ross and He, Kaiming and Hariharan, Bharath and Belongie, Serge "Feature pyramid networks for object detection", pp.2117--2125, 2017.
16. Hariharan, Bharath and Arbel ez, Pablo and Girshick, Ross and Malik, Jitendra "Simultaneous detection and segmentation", pp.297-312, 2014.
17. Wu, Yue and Chen, Yinpeng and Yuan, Lu and Liu, Zicheng and Wang, Lijuan and Li, Hongzhi and Fu, Yun "Rethinking classification and localization for object detection", pp.10186-10195, 2020.
18. Zhao, Qijie and Sheng, Tao and Wang, Yongtao and Tang, Zhi and Chen, Ying and Cai, Ling and Ling, Haibin "M2det: A single-shot object detector based on multi-level feature pyramid network",vol. 33, no.01, pp.9259-9266, 2019.
19. He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian "Deep residual learning for image recognition", pp.770-778, 2016.
20. Rezatofighi, Hamid and Tsoi, Nathan and Gwak, JunYoung and Sadeghian, Amir and Reid, Ian and Savarese, Silvio "Generalized intersection over union: A metric and a loss for bounding box regression", pp.658--666, 2019.
21. Mari, Andrea and Bromley, Thomas R and Izaac, Josh and Schuld, Maria and Killoran, Nathan "Transfer learning in hybrid classical-quantum neural networks", journal on Quantum, vol. 4, pp.340, 2020.