International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

Fruit Detection and Classification using YOLO Models

Chitra Bhole
Computer Engineering Dept
Sir Padampat Singhania University
Udaipur, India
cbhole@somaiya.edu

Chandani Joshi
Computer Engineering Dept
Sir Padampat Singhania University
Udaipur, India
chandani.joshi@spsu.ac.in

Prasun Chakrabarti
Computer Engineering Dept
Sir Padampat Singhania University
Udaipur, India
prasun.chakrabarti@spsu.ac.in

Arvind Sharma
Computer Engineering Dept
Sir Padampat Singhania University
Udaipur, India
Sharma.arvind@spsu.ac.in

Abstract— Computer Vision and Deep Learning techniques have become an advent in multiple domains like healthcare, Technology, as well as Agriculture . Computer vision techniques like object detection are being widely used in agriculture to reduce to efforts required and make agriculture a little more efficient for the farmers. The applications of deep learning in agriculture include leaf disease detection and weather forecasting, and the most advent applications include object detection to detect fruits, and vegetables which can be ensembled with robotics for automated yield production and harvesting. The proposed article describes one such application of fruit detection using various YOLO (You Only Look Once) models. The study encompasses four fruit classes namely Chiku, Mango, Mosambi, and Tomato. Models of Yolo V3, Yolo V4, and Yolo V8 were trained on a customized dataset collected from Indian farms and fruit gardens. The real time images images were collected, pre-processed, and annotated using online labeling tools. A total of 1200 images were used as a part of the complete training process. Basic preprocessing was performed on these images and possible inbuilt augmentation techniques supported by the above-mentioned models were used.Training is applied on custom dataset for all classes. In this experiment we have received the F1 score for YOLOv3(Chiku-82%.Mamgo-91%,Mosambi-87%,,Tomato-77%),YOLOv4(Chiku-89%.Mamgo-98%,Mosambi-95%,,Tomato-91%) and YOLOV8 (Chiku-90%.Mamgo-75%,Mosambi-82%,,Tomato-84%)models. In these models YOLOv4 with two layers gives the highest accuracy for all the classes.

Keywords :-YOLO, Fruit Detection, Deep Learning ,Object Detection

I. INTRODUCTION

India is often referred to as the land of agriculture since most of the exports for various fruits and grains are done from India itself. Recently various Deep Learning applications have been adopted in agriculture to ease the overall process for farmers. Processes such as Harvesting, Ploughing, etc can be automated for faster and more efficient results with lesser manpower involved.

Today, most of the work involved in the agriculture domain is still done manually for example picking fruits, cutting crops sorting the grains etc. Object detection techniques can be implemented to automate such processes and the applications can also be used to supervise, moderate, and keep track of the growth of fruits in fields[1, 2, 3]. Our research is conducted under a similar sector of detecting fruits from Indian fruit trees and providing the count of objects detected in the image.

Object detection works on the concept of initially feeding the model with the annotation boxes so that the model can learn and extract the features of objects so that the model can predict boxes for the objects present in an image. YOLO works on a grid-based approach in which it divides the image into an NxN grid and tries to detect the parts of objects in

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

those images. It uses a single feed-forward network and processes the image only once to detect the object. It tries to detect objects in the respective grids and finally tries to compare and determine the object's center by using the predictions in surrounding grids.

The application of counting the objects detected can be used to get the idea of yield produced every day or the number of fruits exported, track the growth of yield, and monitor the growth cycle of fruits. Recently, there have been many advancements in the field of object detection and fast learning, lightweight and complex models like YOLO (You Only Look Once), RCNN (Region-Based Convolutional Networks) [4] and SSDs (Single Shot Detector) have been developed. The latest versions of these algorithms aim to train in the least possible time and provide reliable accuracies even on smaller datasets. The proposed literature studies and compares the performance of various models on different fruits namely:

1. Chiku
2. Mango
3. Mosambi
4. Tomato

The dataset was collected manually from the surrounding farms, gardens, and the Indian species of plants were taken into consideration for the proposed research thesis. Around 300 images belonging to each of the above-mentioned classes were collected, preprocessed, and labelled using an online image annotator tool named Make Sense AI. The tool provides the option to export the annotations in multiple formats like text files (For YOLO), XML files (Faster RCNN), and CSV Files which contain a path, class name, and bounding box coordinates as columns.

Initially, the images clicked using various devices were resized to the same resolution of 1280x1280 pixels. The standard size of training for YOLO is 416x416 px. However, the model accepts variable image sizes when defining the respective image size in config or YAML files. The image size and also be passed as a parameter in the training command.

YOLO V3 and V4 make use of config files to form the network structure and tune their respective parameters (for example, filter size, strides, channels, etc) for every layer (for example, Net Layer, Conv Layer, and YOLO Layers).The parameters must be tuned with the preceding and successive layer by carefully choosing the mathematical calculation so that the model while processing the images, does not encounter any errors. The Standard YOLO V3 model proposed by AlexyAB repository was used to train the YOLO V3 model. For YOLO V4, two variations of V4 were used. One of them is standard YOLO V4 architecture with 3 YOLO Layers. Second was the YOLO V4 custom architecture with only 2 YOLO Layers, carefully excluding the third YOLO Layer from the traditional architecture. YOLO V8 is the latest version of You Only Look Once object detection model and is assumed to have the highest MAP score among all the models. It is assumed to be the fastest learning object detection model to provide good and reliable accuracy in less than 100 epochs.

To incorporate the counting feature, initially, all the models were trained on the custom dataset and the results were observed for 5000 epochs. Upon carefully examining the results and performing the hyper-parameter tuning wherever required, the model weights were saved separately. Customized scripts were developed to provide the count of fruits detected which involved loading the weights, feeding the images (after resizing, reshaping to required dimensions), and finally outputting the boxes and count by accessing the predictions list. Image processing packages of Python live OpenCV, TensorFlow, PyTorch, and PIL were used to load the weights. Upon feeding the image to the model weights, the model returns a nested list. The nested list contains the labels, predicted bounding box coordinates, and the confidence to predict respective bounding boxes.

II. LITERATURE SURVEY

A. Existing Systems:

The field of Agriculture has seen many advancements in the past few years in research. Studies conducted involve applying deep learning concepts to real-world use cases for efficient processes to assist farmers in simplifying complex and time-consuming tasks. Technology and deep learning techniques have been revolutionizing every domain and agriculture is no exception. The past 3 years have been very effective when it comes to considering the research conducted. Various case studies conducted involve object detection.

The use cases involve detecting potatoes for detecting any possible sprouts for determining the quality of potatoes. [1, 5, 6] involve a case study for classification into three classes i.e. Potato, Germ-Potato, and Sprout-Potato and another one for Mangoes. The researchers proposed YOLO-V5-based techniques and variations[7]. The mAP scores achieved ranged from 67.8 to 88.1%. The study also involves comparative analysis of object detection models like Faster RCNN, SSD, and variations of YOLO.

Few other studies involve using segmentation techniques and essential feature extraction techniques to detect and count the fruits of various classes like Apples, Bananas [8]. Computer vision techniques are combined with noise removal, threshold segmentation, and Maximally Stable Color Region (MSCR) for object recognition, and finally, the Random Forest model to achieve the counting functionality.

The proposed study in [9, 10] uses YOLO v5 to detect and determine the ripeness level of mango fruit. The setup proposed involves a webcam for capturing the feed, a Raspberry Pi 4, and an LCD screen display for output. Techniques like Mango detection using YOLO, hue angle analysis, and CIELAB color segmentation for ripeness level classification were used for enhanced precision. An overall accuracy of 83.09% was obtained.

While [11] uses YOLO-V5 for pineapple fruit detection and determining the ripeness level of the fruits. The study divides the pineapple fruits into three classes depending on the stages of growth namely stage 1 for completely raw, stage 2 for partially ripened, and stage 3 for completely ripened fruits. YOLO V5-x was trained by tuning the parameters to

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

their optimal values for the best results. The accuracy obtained was over 95%.

Another research conducted for cucumber detection in [12] uses instance segmentation. The researchers propose a customized and improved Mask RCNN for precise segmentation of cucumber fruits. Pre-processing techniques like EXG (Excess Green) and LG (Logical Green) were used on binary mask images to eliminate the unnecessary parts of images like surroundings etc. The study also proposes a comparative analysis of models like YOLO V2, YOLO V3, Mask RCNN, Fast RCNN, and improved Mask RCNN with a maximum precision of 90.68 for the improved Mask RCNN.

Some hardware elements associated with IoT can also be taken into consideration for more reliable results like the one proposed in [13, 14, 15]. Initially, color map, edge map, texture map, and convolutional feature extraction were used for the segmentation and detection of fruits. On top of that, the Genetic algorithm for optimization helped deliver better results in terms of Global Average Pooling (GAP). The model also uses classifiers like KNN, Naive Bayes, and SVM [16]resulting in an average accuracy of 92%

The research proposed in [17] is the most similar to the one in our scenario aimed at detecting the fruits from the fruit trees and outputting the count of fruits detected. The researchers tried and tested various object detection models in the YOLO series. All the models from YOLO V2 to YOLO V5[18] along with variations in architectures such as tiny, V5-s, V5-x etc were also tested. YOLO-v3 tiny gave the best accuracy of 98.05% followed by the YOLO-v3 model with an accuracy of 97.40%. Contrast enhancement techniques were implemented while preprocessing the images to separate the features of targetted fruits from the surrounding camouflaging environment of trees and leaves resulting in easier detection and segmentation of fruits in such environments. The models were trained on a customized dataset of images collected using drone imagery and the average density of fruits per image was in the range of 10-15 fruits per image. Inbuilt augmentation techniques provided by the above models were utilized while training and the dataset collected was in video format (mp4) from which frames at the speed of 100fps were collected later on for further processing.

B. Advantages of Existing Systems:

i) Reliable accuracy

ii) Good performance

iii) Use of hybrid, customized, and effective techniques [19]

C. Limitations of Existing System:

i) Do not take into consideration agricultural requirements

ii) Datasets collected and processed were not in an agricultural environment. i.e., fruits were processed after harvesting not on fruits [20]

D. Research Gap:

The research conducted in most of the case studies taken into consideration deals with the detection and classification of fruits based on their ripeness[21] level or estimating the quality of the fruit based on its stages of growth. However, there is a requirement for systems that can perform the same in the pre-harvesting stages i.e. even before the fruits are harvested from the trees. So, further research is required in the area of detecting the fruits on trees and counting them which again introduces various challenges of camouflaging, complex feature extraction techniques, and specialized approaches for reliable results.

III. METHODOLOGY

The proposed methodology for the study conducted includes analyzing and comparing the performance of various versions of the YOLO (You Only Look Once) model namely:

a. YOLO V3
b. YOLO V4
c. YOLO V8

Along with various versions, variations of model architecture also exist in every version of the model. YOLO V3[22, 23] and V4 use the darknet framework for working. The darknet layers are stacked on top of each other for object detection, the last one is known as a detection head. The network architecture can be set by using the config files from the GitHub repository of AlexyAB's darknet repository. The configurations and parameter values need to be set carefully for smooth and errorless functioning of the models and accurate results. YOLO V8 is the latest version of the YOLO family. The various phases of the research go right from data collection to model development and parameter tuning. The various phases of the research study are discussed in detail as follows:

A. Data Collection and Processing:

A customized dataset was used for the case study to be implemented. The dataset was collected from the various fields, farms, and fruit gardens in the periphery. The dataset was preprocessed, resized, and cropped to remove unnecessary parts in an image and all the images were represented in a common format of extensions. At least 300 images belonging to every category were collected from the fields and around 1250 images in total (approximately 300 for each of Chiku, Mango, Mosambi, and Tomato) were collected, and preprocessed.

B. Data Annotation:

The basic requirements for an object detection dataset are images and annotations in the required formats. Models like RCNN, Fast RCNN, and Faster RCNN use Voc-XML files with customized tags for class names, bounding box coordinates, etc. While models of the YOLO family require a text file format of representation for the models to be trained. The text files contain annotation details of every bounding box, one in each line, which consist of class encoding (0 for class1, 1 for class2, and so on..), bounding box coordinates of top left X and Y coordinates, and finally the width and height of the bounding box. A sample representation of the same is shown below:

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

```
0 0.171813 0.147535 0.023655 0.037351
0 0.193601 0.138820 0.019920 0.023655
0 0.256474 0.119522 0.024900 0.022410
0 0.269236 0.139131 0.023033 0.024278
```

Fig 1. Data Annotation Text File Sample

Various tools are available for annotating the dataset for object detection such as LabelImg, VggAnnotator, MakeSenseAI, etc. Our case study involved using Make Sense AI to annotate the images for all the classes i.e. Chiku, Mango, Mosami, and Tomato[24] as shown in the image below:

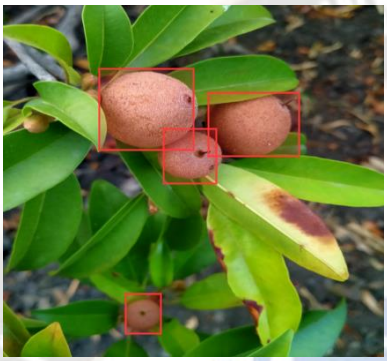

Fig 2. Data Annotation using Make Sense AI (Chiku)

The annotations can be exported in any of the required formats as mentioned above.

C. Model Development:

Once the dataset is ready with the annotations, the next step is to develop the model with the proper setup. Initially, it was decided to test the models V3, V4, and V8. To train the models of V3 and V4, the official GitHub repository of AlexyAB's darknet framework was cloned, and required changes were made like placing the dataset in the data/obj folder, editing the config files, customizing the obj.data and obj.names files and placing them in the data folder, defining the train.txt and test.txt files with names of images to be considered in the training and testing set. We used a custom Python script to divide the dataset into train and test sets and output required train.txt and test.txt files. For YOLO V8, specific folders need to be created for training, testing, and validation sets and the path is to be provided in the YAML file with appropriate parameters and obj.data for YOLO v3 and V4. The development for various models and their variations are discussed below in detail:

a. YOLO V3:

As mentioned above, the YOLO V3 model was trained using the official darknet repository. The config file was edited to set all the parameter values and the layer structure was adjusted to the standard YOLO V3. The standard YOLO V3 architecture is shown in the image below:

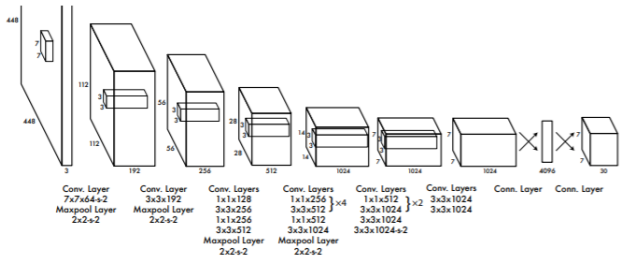

Fig 3. Standard YOLO V3 Architecture (source)

The standard YOLO V3 comes with a total of 106 convoluted neural network layers out of which 3 are standard YOLO Layers for feature extraction, 75 are convolutional fully connected layers, 23 shortcut layers, 4 route layers, and 2 upsample layers.

Fig 3 shows the standard YOLO V3 architecture with filter sizes and sizes after max-pooling of every layer. The necessary changes that everyone should consider while training a custom YOLO V3 model are setting up the filter size, and changing values of parameters like batch, subdivision, max_batches, steps, filters, etc. Also, it is essential to edit the files obj.names with class names, one in each new line, and obj.data with the paths of train, and test txt files and number of classes. The default image input size for V3 is 416 x 416 pixels.

b. YOLO V4:

Another advantage of using AlexyAB's official Darknet repository is that one can use the same setup of V3 to train the V4 model as well. The only changes required are in the config files. No need to perform any additional changes. Now, while training for YOLO V4, two variations of the model were taken into consideration: standard YOLO V4 with 3 YOLO Layers, and YOLO V4-tiny [25] i.e. V4 with 2 YOLO layers for feature extraction. YOLO V4 is advancement over V3 which uses CSPDarknet53 as its backbone. PAN net for feature aggregation and much more. It uses more convolutional networks for faster processing and better accuracy.

The YOLO V4 tiny model i.e. 2 YOLO Layer model is specialized to detect extremely tiny objects i.e. objects with a size lesser than 15 pixels. The customized network[26] is capable of extracting essential features from complex datasets (like the config for Chiku detection in our scenario). The two-layered architecture is fast in processing the dataset. There are not many architectural differences in YOLO V3[27] and V4 but only the backbone differences in feature aggregation, layer stacking, etc.

The customized YOLO V4 tiny was designed by making required changes in the config file such as twisting the network structure a little. The customized model only has 39 layers in total out of which, 21 are convolutional layers, 2 YOLO layers, 1 net layer, 1 upsample layer, 11 route layers, and 3 max pool layers are present. The modified network can be customized by setting all the required parameters to their appropriate values by following the previous layers such as strides, filters, etc. The activation layers of the convolutional

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

networks just before YOLO are set to linear and all other layer activations are set to leaky. The same parameters and network trends are used for the standard YOLO V4 model. The complexity of the standard model networks uncovers the hidden and high-level features [28] but in our scenario, low-level features are effective and essential for object detection. Hence, the customized YOLO V4 model for tiny objects performs better than the standard V4 network.

c. YOLO V8:

YOLO V8 is the latest object detection model of the YOLO family. It has the highest mAP score ever achieved and is currently the fastest training model. V8 has various advancements as compared to the previous versions such as the use of Feature Pyramid Networks, stronger backbone models of the darknet, multi-scale prediction, advanced training techniques, and object localization, and the usage of inbuilt augmentation techniques like random cropping, angular rotation, hue-saturation-value manipulation make the model to detect the objects for far dissimilar scenarios [29]. YOLO V8 shows improved performance over image, video, and webcam analysis case studies. The model supports the classification, object detection, and segmentation of images as per the needs.

The dataset to be trained for YOLO V8 needs to be segregated into three folders namely: Train containing the images and labels for the images, Test folder containing the test samples and Validation directory containing the images and annotations for the images to be validated during training for model improvement and validation [30]. The directory structure is shown in the following figure:

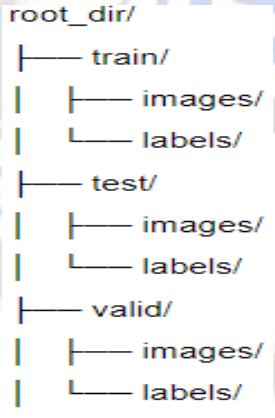

Fig 4. Dataset Directory Structure for YOLO V8

As mentioned earlier, all the necessary modules were loaded from the official Python package named Ultralytics[31, 32] for necessary YOLO V8 support. All the parameters that need to be altered can be defined with their optimal values in a YAML file and the file path can be provided while training the model. However, one can also set the model parameters using the command line interface as well simply by passing the value and name of the parameter to be tuned to.

There are 4 different types of YOLO V8 [33] models proposed by the ultralytics module based on the network size and their learning capabilities namely:

i) YOLO V8n - Nano Model

ii) YOLO V8s - Small Model

iii) YOLO V8m - Medium Model

iv) YOLO V8l - Large Model

v) YOLO V8x - Xtreme Model

These models vary in their architectural compositions and scenarios where they can be applied for. For example, if all the models are applied on the same dataset of an object detection case study, the results of the detections would vary in terms of precision of the bounding boxes, the ability of the model to detect in complex scenarios, etc. We used YOLO V8n for our case study since it is the most suitable for our scenario. The model has 225 layers in total with around 30 million parameters.

	from	n	params	module	arguments
0	-1	1	464	ultralytics.nn.modules.conv.Conv	[3, 16, 3, 2]
1	-1	1	4672	ultralytics.nn.modules.conv.Conv	[16, 32, 3, 2]
2	-1	1	7360	ultralytics.nn.modules.block.C2f	[32, 32, 1, True]
3	-1	1	18560	ultralytics.nn.modules.conv.Conv	[32, 64, 3, 2]
4	-1	2	49664	ultralytics.nn.modules.block.C2f	[64, 64, 2, True]
5	-1	1	73984	ultralytics.nn.modules.conv.Conv	[64, 128, 3, 2]
6	-1	2	197632	ultralytics.nn.modules.block.C2f	[128, 128, 2, True]
7	-1	1	295424	ultralytics.nn.modules.conv.Conv	[128, 256, 3, 2]
8	-1	1	460288	ultralytics.nn.modules.block.C2f	[256, 256, 1, True]
9	-1	1	164608	ultralytics.nn.modules.block.SPPF	[256, 256, 5]
10	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
11	[-1, 6]	1	0	ultralytics.nn.modules.conv.Concat	[1]
12	-1	1	148224	ultralytics.nn.modules.block.C2f	[384, 128, 1]
13	-1	1	0	torch.nn.modules.upsampling.Upsample	[None, 2, 'nearest']
14	[-1, 4]	1	0	ultralytics.nn.modules.conv.Concat	[1]
15	-1	1	37248	ultralytics.nn.modules.block.C2f	[192, 64, 1]
16	-1	1	36992	ultralytics.nn.modules.conv.Conv	[64, 64, 3, 2]
17	[-1, 12]	1	0	ultralytics.nn.modules.conv.Concat	[1]
18	-1	1	123648	ultralytics.nn.modules.block.C2f	[192, 128, 1]
19	-1	1	147712	ultralytics.nn.modules.conv.Conv	[128, 128, 3, 2]
20	[-1, 9]	1	0	ultralytics.nn.modules.conv.Concat	[1]
21	-1	1	493056	ultralytics.nn.modules.block.C2f	[384, 256, 1]
22	[15, 18, 21]	1	751507	ultralytics.nn.modules.head.Detect	[1, [64, 128, 256]]

Fig 5. Model Summary for YOLO V8n

Above figure shows the model summary for the latest YOLO V8 model's nano version. The model also supports various built-in augmentation techniques as mentioned earlier[34, 35].

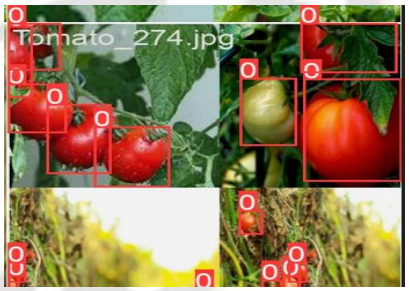

Fig 6. Random Cropping in YOLO V8(Tomato)

Figure 6 shows an example of a type of inbuilt augmentation supported by the YOLO V8 model in which random image sections are cropped randomly and combined to generate completely new instances in the dataset using the existing images and annotations. Such techniques help build models that are more robust and perform well in real-life use-case scenarios.

D. Hyperparameter Tuning:

Once the model training phase is ready and the results for initial iterations are observed, it is essential to tune the model parameters for a better learning curve and reliable performance. Hyperparameter tuning is very important to get your model working in good condition and fitting better for a generalized set of images. Some of the highly influential

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

parameters for an object detection model are learning rate, decay rate, anchor boxes, activation function, etc.

TABLE I. PARAMETER VALUES FOR BETTER RESULTS

Hyperparameter	Value
Training Batch Size	64
Subdivisions	16
Saturation	1.5
Exposure	1.5
Learning Rate (α)	0.0001
Hue	0.1
Filter Size	18
Activation	Leaky and Linear for the last Layer
Momentum	0.949

Table I shows the hyper-parameters to be tuned to their optimal values in order to achieve desired results as discussed in the following section.

E. Object Counting:

Once, all the models were fine-tuned and were performing up to the mark, a customized Python script was developed in order to iterate over the nested tuple returned by the model when fed with an image to predict the bounding boxes. The tuple returned is a 3-dimensional array-like structure that consists of [36]

i) coordinates of the bounding boxes

ii) labels for the prediction

iii) confidence of prediction for the respective bounding boxes.

The Python script simply iterates through the list and returns the length of the bounding box coordinates list in the tuple, which is identified to be the total number of bounding boxes predicted by the model. To achieve this, the model weights were first loaded using various modules and frameworks like Tensorflow, and PyTorch. OpenCV, Ultralytics, and Pillow were used [37]. Loading the models using these libraries helps us eliminate unnecessary steps like installing dependencies, setting up or cloning the repositories while training, providing paths to unnecessary files for mode configuration, etc

IV RESULTS AND DISCUSSIONS

In the last section, we discussed the implementation and the methodology used to implement the proposed system. Now, in this section, we will discuss the proposed models' results. Standard YOLO v3, standard YOLO v4, customized YOLO v4-tiny, and standard YOLO v8 models were trained for a customized dataset of Chiku, Mango, Mosambi, and Tomato fruits. The proposed system is a combination of different versions of the You Only Look Once (YOLO) algorithm trained and fine-tuned on a customized dataset. The models are developed in order to detect the fruits in agricultural environments, from the images taken directly from the fruit trees and provide a final count of the total number of fruits detected in the input image, as shown in the following figures.

The images shown below are the outputs of YOLO-V3, V4 customized and V8 models trained on our custom dataset for 5000 epochs

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

Fig 7. Detection outputs of YOLO V3, V4-customized and V8 for all four fruits

Three different models (excluding variations of the models of the same version) were trained on exact same dataset and the following results were obtained. The research conducted takes into account the performance of the models on four different categories of fruits namely Chiku, Mango, Mosambi, and Tomato. All the variations of models were trained for each of these 4 categories and the fruit-wise results obtained were as follows:

TABLE II. FRUIT-WISE RESULTS FOR VARIOUS MODELS

A. CHIKU

Model	**V3**	**V4**	**V8**
mAP Score	80.72	93.39	62.22
Precision	0.83	0.85	0.94
Recall	0.81	0.94	0.87
F1-Score	0.82	0.89	0.90

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 10
Article Received: *25 July 2023* ***Revised:*** *12 September 2023* ***Accepted:*** *30 September 2023*

Loss	0.3082	0.2146	0.3971

B. MANGO

Model	**V3**	**V4**	**V8**
mAP Score	92.47	99.87	65.5
Precision	0.94	0.99	0.74
Recall	0.87	1.00	0.60
F1-Score	0.91	0.98	0.75
Loss	0.4596	0.3265	0.4059

C. MOSAMBI

Model	**V3**	**V4**	**V8**
mAP Score	90.47	97.53	84.44
Precision	0.84	0.97	0.85
Recall	0.91	0.93	0.80
F1-Score	0.87	0.95	0.82
Loss	0.6547	0.4965	0.31482

Table II A, B, C, and D show the fruit-wise results for various models of YOLO. Note that, the results for the V4 model obtained are for the customized model with only 2 YOLO layers for feature extraction and customized activation functions so that the model works the best for our case study. As shown in Table II, the highest map scores are achieved for the customized YOLO V4 model with two YOLO layers. Various model architectures were experimented and the results were observed.

D. TOMATO

Model	**V3**	**V4**	**V8**
mAP Score	78.48	96.98	87.48
Precision	0.77	0.88	0.85
Recall	0.77	0.94	0.84
F1-Score	0.77	0.91	0.84
Loss	0.4204	0.2862	0.47209

The mAP score for Mango is the highest of all for our customized V4 model with two layers with a mean Average Precision of 99.87% followed by 97.53% for Mosambi, 96.98% for Tomato, 93.39% for Chiku. The mAP scores for V4 with two layers is the highest, V3 is the second highest, and the least for V8 since much of the customization was not performed for the latter models. The training results in terms of mAP scores for every 1000th iteration are shown in the figure below:

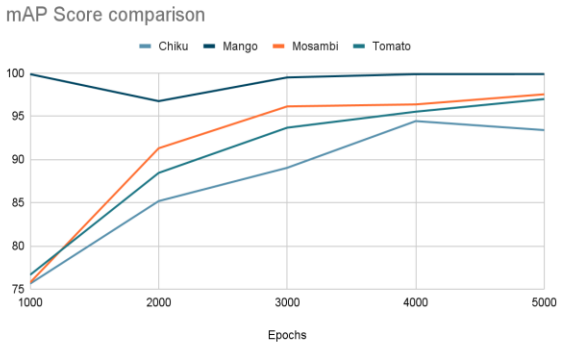

Fig 8. Epoch-wise comparison of mAP Scores for All Fruits

From Table II, we can see that the losses for all the models are quite low, and the lowest for the YOLO V8 model in most of the scenarios.

Figure 9 shows the confusion matrix for our customized YOLO V4 model for all the fruits in the order: Chiku, Mango, Mosambi and Tomato. As observed, the True Positive and True Negative predictions are very high resulting in higher values of Precision, Recall and F1-scores as shown in the tables above. Confusion matrix are a good estimation of the performance of models since it can help us avoid type 1 and type 2 errors in model performance.

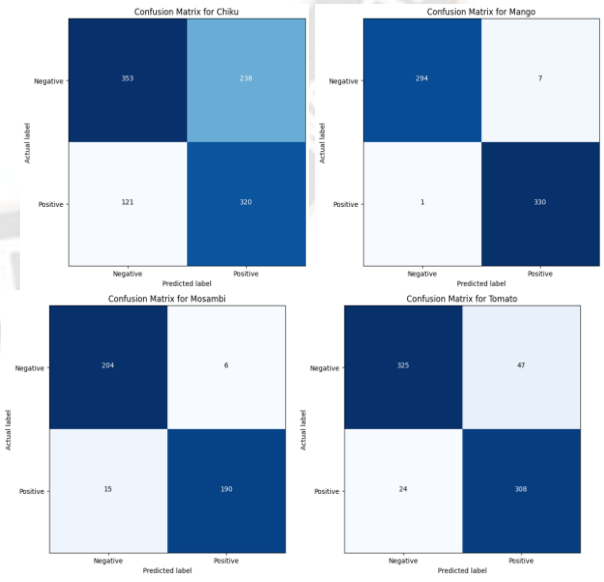

Fig 9. Confusion Matrices for YOLO-V4 Customized Model for all the Categories

V. CONCLUSION

As discussed in the above sections, it is clear that the performance of customized YOLO V4 is the best of all the other models because of its high mAP scores, lower loss values, and the optimal values of other parameters like precision, recall, and f1-scores. These results were achieved because of the architectural enhancements in the network structures. Further optimization can be possible for Chiku fruit since the scores of other fruits are higher.

VI. FUTURE SCOPE

While the results of the proposed study are good and reliable, further enhancements and improvements can be incorporated by using advanced techniques like using a hybrid model, segmentation techniques, and semantic edge detection for fruits, etc.

VII. CONFLICT OF INTEREST

The authors have no conflicts of interest to declare.
As per the research accuracy can be improved at different condition and different regions.

REFERENCES

[1] A. Ranjan and R. Machavaram, "Detection and Localisation of Farm Mangoes using YOLOv5 Deep Learning Technique," 2022 IEEE 7th International conference for Convergence in Technology (I2CT), Mumbai, India, 2022, pp. 1-5, doi: 10.1109/I2CT54291.2022.9825078.

[2] X. Li, Y. Qin, F. Wang, F. Guo and J. T. W. Yeow, "Pitaya detection in orchards using the MobileNet-YOLO model," 2020 39th Chinese Control Conference (CCC), Shenyang, China, 2020, pp. 6274-6278, doi: 10.23919/CCC50068.2020.9189186.

[3] Q. An, K. Wang, Z. Li, C. Song, X. Tang and J. Song, "Real-Time Monitoring Method of Strawberry Fruit Growth State Based on YOLO Improved Model," in IEEE Access, vol. 10, pp. 124363-124372, 2022, doi: 10.1109/ACCESS.2022.3220234.

[4] M. O. Shahzad, A. B. Aqeel and W. S. Qureshi, "Detection of Grape Clusters in Images Using Convolutional Neural Network," 2023 International Conference on Robotics and Automation in Industry (ICRAI), Peshawar, Pakistan, 2023, pp. 1-6, doi: 10.1109/ICRAI57502.2023.10089582.

[5] G. Dai, L. Hu, J. Fan, S. Yan and R. Li, "A Deep Learning-Based Object Detection Scheme by Improving YOLOv5 for Sprouted Potatoes Datasets," in IEEE Access, vol. 10, pp. 85416-85428, 2022, doi: 10.1109/ACCESS.2022.3192406.

[6] R. Pavithra and P. S. Priyadharshini, "A Random Forest Based Smart Fruit Counting," 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2022, pp. 729-733, doi: 10.1109/ICAIS53314.2022.9742725.

[7] Y. Li, Z. Gong, Y. Zhou, Y. He and R. Huang, "Production Evaluation of Citrus Fruits based on the YOLOv5 compressed by Knowledge Distillation," 2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Rio de Janeiro, Brazil, 2023, pp. 1938-1943, doi: 10.1109/CSCWD57460.2023.10152740.

[8] J. S. Ignacio, K. N. A. Eisma and M. V. C. Caya, "A YOLOv5-based Deep Learning Model for In-Situ Detection and Maturity Grading of Mango," 2022 6th International Conference on Communication and Information Systems (ICCIS), Chongqing, China, 2022, pp. 141-147, doi: 10.1109/ICCIS56375.2022.9998163.

[9] X. Liu, D. Zhao, W. Jia, W. Ji, C. Ruan and Y. Sun, "Cucumber Fruits Detection in Greenhouses Based on Instance Segmentation," in IEEE Access, vol. 7, pp. 139635-139642, 2019, doi: 10.1109/ACCESS.2019.2942144.

[10] R. B. A. Alde, K. D. L. De Castro and M. V. C. Caya, "Identification of Musaceae Species using YOLO Algorithm," 2022 5th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 2022, pp. 666-671, doi: 10.1109/ISRITI56927.2022.10052913.

[11] P. Nirale and M. Madankar, "Design of an IoT Based Ensemble Machine Learning Model for Fruit Classification and Quality Detection," 2022 10th International Conference on Emerging Trends in Engineering and Technology - Signal and Information Processing (ICETET-SIP-22), Nagpur, India, 2022, pp. 1-6, doi: 10.1109/ICETET-SIP-2254415.2022.9791718.

[12] G. Xia, J. Dan, H. Jinyu, H. Jiming and S. Xiaoyong, "Research on Fruit Counting of Xanthoceras Sorbifolium Bunge Based on Deep Learning," 2022 7th International Conference on Image, Vision and Computing (ICIVC), Xi'an, China, 2022, pp. 790-798, doi: 10.1109/ICIVC55077.2022.9886298.

[13] H. C. Pichhika and P. Subudhi, "Detection of Multi-varieties of On-tree Mangoes using MangoYOLO5," 2023 11th International Symposium on Electronic Systems Devices and Computing (ESDC), Sri City, India, 2023, pp. 1-6, doi: 10.1109/ESDC56251.2023.10149849.

[14] A. R. Jimenez, R. Ceres and J. L. Pons, "A machine vision system using a laser radar applied to robotic fruit harvesting," Proceedings IEEE Workshop on Computer Vision Beyond the Visible Spectrum: Methods and Applications (CVBVS'99), Fort Collins, CO, USA, 1999, pp. 110-119, doi: 10.1109/CVBVS.1999.781100.

[15] A. Stepanova et al., "Harvesting tomatoes with a Robot: an evaluation of Computer-Vision capabilities," 2023 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Tomar, Portugal, 2023, pp. 63-68, doi: 10.1109/ICARSC58346.2023.10129601.

[16] Z. Chen, S. Elsaid, D. Y. Y. Sim, T. Maul and I. Y. Liao, "Detection of Oil Palm Fresh Fruit Bunches (FFBS) with Computer Vision Models," 2022 International Conference on Mechanical, Automation and Electrical Engineering (CMAEE), Chengdu, China, 2022, pp. 86-91, doi: 10.1109/CMAEE58250.2022.00023.

[17] H. Sato, M. Ishii, H. Ito and K. Dohsaka, "Determining Fruit Set Density and Spatial Arrangement of Fruit Trees for Fruit Picking Operations," 2022 Joint 12th International Conference on Soft Computing and Intelligent Systems and 23rd International Symposium on Advanced Intelligent Systems (SCIS&ISIS), Ise, Japan, 2022, pp. 1-2, doi: 10.1109/SCISISIS55246.2022.10002107.

[18] Y. Wang, L. Yang, H. Chen, A. Hussain, C. Ma and M. Al-gabri, "Mushroom-YOLO: A deep learning algorithm for mushroom growth recognition based on improved YOLOv5 in agriculture 4.0," 2022 IEEE 20th International Conference on Industrial Informatics (INDIN), Perth, Australia, 2022, pp. 239-244, doi: 10.1109/INDIN51773.2022.9976155.

[19] M. H. Tunio, L. Jianping, M. H. F. Butt, I. Memon and Y. Magsi, "Fruit Detection and Segmentation Using Customized Deep Learning Techniques," 2022 19th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP), Chengdu, China,

2022, pp. 1-5, doi: 10.1109/ICCWAMTIP56608.2022.10016600.
[20] W. Yijing, Y. Yi, W. Xue-fen, C. Jian and L. Xinyun, "Fig Fruit Recognition Method Based on YOLO v4 Deep Learning," 2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Chiang Mai, Thailand, 2021, pp. 303-306, doi: 10.1109/ECTI-CON51831.2021.9454904.
[21] A. N. Yumang, D. C. Rubia and K. P. G. Yu, "Determining the Ripeness of Edible Fruits using YOLO and the OVA Heuristic Model," 2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM), Boracay Island, Philippines, 2022, pp. 1-6, doi: 10.1109/HNICEM57413.2022.10109379.
[22] A. Marin and E. Radoi, "Image-based Fruit Recognition and Classification," 2022 21st RoEduNet Conference: Networking in Education and Research (RoEduNet), Sovata, Romania, 2022, pp. 1-4, doi: 10.1109/RoEduNet57163.2022.9921050.
[23] Y. Pratama, I. Iskandar and P. T. P. Giawa, "Implementation of Convolutional Neural Network on Farming Robots for Detecting Broccoli," 2022 International Conference on ICT for Smart Society (ICISS), Bandung, Indonesia, 2022, pp. 01-07, doi: 10.1109/ICISS55894.2022.9915044.
[24] M. Boralkar and S. G. S. Yadav, "Smart Farming Techniques For Precision Agriculture," 2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), Mandya, India, 2022, pp. 1-7, doi: 10.1109/ICERECT56837.2022.10060521.
[25] R. S. Latha et al., "Fruits and Vegetables Recognition using YOLO," 2022 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2022, pp. 1-6, doi: 10.1109/ICCCI54379.2022.9740820.
[26] C. Yi, W. Wu, L. Yang and R. Jia, "Research on Fruit Recognition Method Based on Improved YOLOv4 Algorithm," 2023 IEEE 2nd International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA), Changchun, China, 2023, pp. 1892-1901, doi: 10.1109/EEBDA56825.2023.10090849.
[27] S. C, N. Manasa, V. Sharma and N. K. A. A., "Vegetable Classification Using You Only Look Once Algorithm," 2019 International Conference on Cutting-edge Technologies in Engineering (ICon-CuTE), Uttar Pradesh, India, 2019, pp. 101-107, doi: 10.1109/ICon-CuTE47290.2019.8991457.
[28] A. A. Menon, A. K. Nair, A. Vasudevan, K. D. K. S and A. T, "RipId App: Fruit Ripeness Estimator and Object Recognition Application," 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2022, pp. 1686-1691, doi: 10.1109/ICOEI53556.2022.9777106.
[29] https://medium.com/mlearning-ai/YOLO-v8-the-real-state-of-the-art-eda6c86a1b90
[30] https://medium.com/augmented-startups/train-YOLOv8-on-custom-data-6d28cd348262
[31] https://docs.ultralytics.com/
[32] https://github.com/ultralytics/ultralytics/tree/main/ultralytics
[33] https://medium.com/@hichengkang/custom-dataset-in-YOLO-v8-bf7468313026
[34] https://medium.com/red-buffer/apply-data-augmentation-on-YOLOv5-YOLOv8-dataset-958e89d4bc5d
[35] https://github.com/ultralytics/YOLOv5/issues/1929
[36] https://www.kaggle.com/code/paulojunqueira/YOLO-v8-vehicles-detecting-counting
[37] https://medium.com/@tenyks_blogger/this-weeks-top-5-tips-to-solve-performance-bottlenecks-in-fruit-counting-73407916d44d