CHAPTER - 1

INTRODUCTION

In most developing countries such as India, agriculture constitutes the major part of the economy and it is basic occupation of Indian people. Agronomy, Horticulture, Agricultural Engineering, Agricultural Economics, Animal Science are the five main branches of agriculture. Horticulture is one of the most cost-demanding factors in agriculture industry. Horticulture deals with the study of cultivation of fruits, vegetables, and ornamental crops. Horticulturists apply their knowledge skills, and technology intensively produce fruits for personal or social needs. Their main aim is to improve fruit growth, yield, nutritional value, quality and resistance to insects and diseases. Now-a-days, robotic harvesting has done to help farmers. Reducing input cost, minimizing working time, improving fruits quality and increasing yield of fruits is the main goal of horticulturist. Image processing, computer vision, machine-learning techniques, robotic harvesting playing important role in horticulture sector. Production and consumption of agricultural products, especially fruit and vegetables, in India has shown a marked upward trend over the past few years.

Post-harvest processing of fruits comprises a series of unit operations, including cleaning, waxing, sorting, grading, packing, transport, and storage, while counting without damage to fruit has being considered the most important post-harvest step. Farmers continue to examine and sort/grade harvested fruits by visual appearance. Manually classification and counting of fruits based on geometrical or shape features is laborious, stressful and costly. Thus, there is a critical need to be able to quick, accurate and efficiently evaluate agricultural products without the use of human labor. Therefore, automated classification and sorting systems that use image processing, computer vision, machine-learning techniques to determine geometrical and shape parameters, such as size, shape, color, ripeness, mass, bruising, disease, and rot has being developed in many countries. In addition, an automated detection and counting of on tree fruit system would free up labor, which is already in short supply and expensive.

1.1 NEED OF PROJECT

1.1.1 Help the farmers in horticulture industry

Generally, in India the quality inspection of fruits has performed by human experts. This manual sorting by visual inspection is labor intensive, time consuming and suffers from the problem of inconsistency and inaccuracy in judgment by different human. Machine learning, computer vision and image processing techniques have found increasingly useful in the fruit industry, especially for applications in quality inspection and defect sorting applications.

1.1.2 Robotic Harvesting of on tree fruits

Autonomous robotic harvesting is a rising trend in agricultural applications, like the automated harvesting of fruit. Farmers continuously look for solutions to upgrade their production, at reduced running costs and achieve maximum profit within less amount of time. This is where harvesting robots come into play. One of the challenges of developing a fruit-harvesting robot is fruit detection. To detect the fruit, an image processing approach will be used and has the ability to classify and detect on tree fruits in cluster and partially occluded fruit.

1.1.3 Reduce labor cost and labor work

Cut fruit from tree, sort them on basis of ripe condition, count number of fruits requires a lot of labor and it is time consuming. There is need of some automatic machine that cut, sort and count fruits in minimum time without degrading the quality of fruit. Image processing and robotic technology plays important role to solve this problem.

1.1.4 Avoid damaging of fruits

During manual counting, fruits are count as well as separate according to their growth conditions. Ripe, unripe, over-ripe fruits has distributed in separate basket. Fruits may get damage by moving fruits manually from one basket to other basket. Image processing, video processing and conveyor belt system will help to avoid damaging of fruits during counting and sorting of fruits.

CHAPTER - 2

LITERATURE SURVEY

This section consist the description of existing methodologies such as computer vision, image processing, deep convolutional neural network, machine learning, computer vision.

2.1 Deep Fruits: A Fruit Detection System Using Deep Neural Networks

Inkyu Sa, Zongyuan Ge, Feras Dayoub, Ben Upcroft, Tristan Perez and Chris McCool [1] suggested a new approach for detection of fruit using deep neural network. They used transfer learning for the task of fruit detection using images obtained from two modalities namely color (RGB) and Near-Infrared (NIR). Detection operation performed by fine-tuning of the VGG16 network based on the pre-trained ImageNet model. Fine tune consists of updating the model parameter using the new data. It involves a new classification layer, updating all of the layers and obtained a new model. They performed rapid training about 2 hours on a K40 GPU to obtained trained model on seven different categories of fruit. They proved through their results that use of RGB and NIR multi-modal information within early and late fusion networks provides improvement over a single deep convolution neural network (DCNN).Their model was able to accurate detect apple, avocado, capsicum, mango, orange, rockmelon, strawberry fruits

2.2 Deep Count: Fruit Counting Based on Deep Simulated Learning

Maryam Rahnemoonfar and Clay Sheppard [2] proposed a simulated learning for counting fruits. They used novel deep learning architecture for counting fruits based on convolutional neural networks (CNN) and a modified version of Inception-ResNet. Inception-ResNet combines the ideas of Inception, which captures features at multiple sizes by concatenating the results of convolutional layers with different kernel sizes, and residual network, which use skip connections to create a simple path for information to flow throughout a neural network. The network has implemented using TensorFlow running on an NVidia 980Ti GPU. In this paper

operation performed only on tomato fruit. It can count accurately tomatoes under shadow, occluded by foliage, branches, or if there is some degree of overlap amongst tomatoes.

2.3 AFDGA: Defect Detection and Classification of Apple Fruit Images using the Modified Watershed Segmentation Method

A.Raihana and R. Sudha [3] proposed a new approach to detect the defected fruits using AFDGA-Apple Fruit Detection, Grading and Analyzation. They used Modified Watershed Segmentation to segment the fruits from image. For analysis of defected fruit they used grey level co-occurrence matrix (GLCM) based feature extraction method. Finally, for fruit classification images they used support vector machine (SVM) in terms of its features. An input image consists of apple fruits in various conditions like healthy, injured, and red injured, red healthy, green injured and green healthy. Operations like noise removal, image enhancement, image segmentation, feature extraction, feature selection, SVM classification and apple grading done sequentially on images. Whole simulation of experiment has done on MATLAB . In this paper, they able to conclude accurate detection and classification result in both experiment and simulation based results obtained from MATLAB.

2.4 A Machine Vision-Based Maturity Prediction System for Sorting of Harvested Mangoes

Chandra Sekhar Nandi, Bipan Tudu, and Chiranjib Koley [4] performed the prediction of maturity level by help of video signal collected by the Charge Coupled Device (CCD) camera placed on the top of the conveyer belt carrying mangoes. Extracted image frames from the video signal have been corrected and processed to extract color features, which was found to be more relevant for the prediction of maturity level. Further author performed recursive feature elimination technique in combination with support vector machine (SVM). SVM based classifier has employed to identify the most relevant features among the initially chosen 27 features. Finally, the optimum set of reduced number of features have obtained and used for classification of the mangoes into four different classes according to the maturity level.

2.5 A New Method for Fruits Recognition System

Woo Chaw Seng [5] presented a new Fruit recognition system has been proposed, which combines three features analysis methods: color-based, shape based and size-based in order to increase accuracy of recognition. For Fruits Recognition System, the KNN algorithm performs fruit classification by using the distance measure system shows the fruit name and a short description to user. Proposed fruit recognition system analyzes, classifies and identifies fruits successfully up to 90% accuracy.

2.6 Classification of Ripe or Unripe Orange Fruits Using the Color Coding Technique

B.Kanimozhi and R.Malliga [6] introduces a new approach to recognize ripe and unripe orange author discussed Otsu segmentation method. This had used for automatically perform clustering-based image thresholding or the reduction of a gray-level image to a binary image. The algorithm assumes that the image contains two classes of pixels following bi-modal histogram foreground pixels and background pixels, it then calculates the optimum threshold separating into two classes ripe and unripe The applied procedure codes were described and run in MATLAB. Authors was able to implement this technique for robotic application in fruit harvesting industry.

2.7 A Novel Red Apple Detection Algorithm Based on AdaBoost Learning

Donggi Kim, Hongchul Choi, Jaehoon Choi, Seong Joon Yoo and Dongil Han [7] has proposes an AdaBoost learning algorithm used for detecting apples to measure the number of apples on the trees. The proposed algorithm explores whether there are apple trees or not based on the number of image block-unit edges, and then it detects apple areas. In order to extract colors appropriate for apple areas, the CIE L*a*b* color space has been used. In order to extract apple characteristics strong against illumination changes, modified census transform (MCT) has used. Second step, by using the AdaBoost learning algorithm characteristics data on the apples are learn and generated. With the generated data, the detection of apple areas has been made. Further

authors proposed AdaBoost algorithm has a higher detection rate than existing pixel-based image processing algorithms and minimizes false detection.

2.8 Classification of Fruits Using Computer Vision and a Multiclass Support Vector Machine

Yudong Zhang and Lenan Wu [8] performed experiments in three stages. In first stage background of each image removed by a split-and-merge algorithm. In second stage the color histogram, texture and shape features of each fruit image has extracted to compose a feature space. Third, principal component analysis (PCA) was used to reduce the dimensions of feature space; Finally, three kinds of multi-class SVMs were constructed, i.e.,Winner-Takes-All SVM, Max-Wins-Voting SVM, and Directed Acyclic Graph SVM. Meanwhile, three kinds of kernels were chosen, i.e., linear kernel, Homogeneous Polynomial kernel, and Gaussian Radial Basis kernel; finally, the SVMs were trained using 5-fold stratified cross validation with the reduced feature vectors as input. Thus, authors were able to classify 18 categories fruit successfully.

2.9 A Segmentation Algorithm for the Automatic Recognition of Fuji Apples at Harvest

D.M. Bulanon, T. Kataoka, Y. Ota; T. Hiroma [9] proposed a machine vision system which includes color charge coupled device camera to capture apple images, and a personal computer to process images for recognizing and locating the fruit. First input image has converted into gray scale image. Second histogram of image of the luminance created. The histogram of the red color difference grey-level image shows that the higher peak is associated with the leaves, branches and other parts of the background. The lower peak represents the red fruit. To separate the fruit from the other two portions using the red color difference, a contour drawn on the fruit or red color boundary and thus classified the fruit from the background.

2.10 On Plant Detection of Intact Tomato Fruits Using Image Analysis and Machine Learning Methods

Kyosuke Yamamoto, Wei Guo, Yosuke Yoshioka and Seishi Ninomiya [10] discussed image-processing method for accurately detect individual intact tomato fruits, including mature, immature and young fruits, on plant using a conventional RGB digital camera in conjunction with machine learning approaches. The proposed method consists of three steps. At the first step pixel-based segmentation was conducted to roughly segment the pixels of the images into classes composed of fruits, leaves, stems and backgrounds. Further, Blob-based segmentation conducted to eliminate misclassifications generated at the first step. At the third step, *X*-means clustering applied to detect individual fruits in a fruit cluster. The developed method did not require an adjustment of the threshold values of each image for fruit detection because the image segmentations has conducted based on classification models generated by machine learning approach.

CHAPTER – 3

OBJECTIVE OF THE WORK

3.1 AIM

The aim of the project is to use computer vision, machine-learning and deep learning techniques to classify, detect and count on tree fruits from images and videos.

3.2 OBJECTIVES OF PROPOSED WORK

The main objectives of this project are:

- To classify fruit use deep convolution neural network model and machine learning algorithms.
- To detect fruit from image remove background such as leaves, branches, sky and soil from image and video.
- To count number of fruits by using computer vision techniques.

3.3 PROBLEM STATEMENT AND SCOPE

3.3.1 PROBLEM STATEMENT

Application of machine learning, deep neural network, computer vision and image-processing techniques for fruit classification, detection and counting of fruits present on tree in an image and videos.

3.3.2 SCOPE OF PROPOSED SYSTEM

The scope of this project is limited to 12 different categories of fruits that are available in the fruit data sets which is used to train the system. Orange, lemon, pineapple, banana, strawberry, pomegranate, granny- smith apple, blueberry, cherry, peach, raspberry and walnut are the fruits used to train dataset.

3.4 DATA PREPARATION

3.4.1 DATA COLLECTION

Dataset used for this project work is an image dataset of size 224 X 224. It contains images of 12 different categories of fruits viz ; orange, lemon, pineapple, banana, strawberry, pomegranate, granny- smith apple, blueberry, cherry, peach, raspberry and walnut. It includes 4631 training images and 3317 testing images.

Fruit	**Number of Training Images**	**Number of Test Images**
Oranges	345	134
Lemon	246	100
Pineapple	377	113
Banana	320	170
Strawberry	305	100
Pomegranate	146	100
Granny Smith Apple	392	100
Blueberry	500	500
Cherry	500	500

Peach	500	500
Raspberry	500	500
Walnut	500	500
Total Images	4631	3317

Table 3.1: Details of Dataset

CHAPTER – 4

ARCHITECTURE OF SYSTEM

4.1 CLASSIFICATION OF FRUITS USING MACHINE LEARNING

Classification of fruit is one of the biggest challenge is not achieve by computer vision techniques. Generally, fruits are classified based on their own size, shape, color, height, diameter and weight but these parameter are not compatible with the image-processing techniques. Machine learning is a technique that learn from features provided to it and predict accurate results. In this project, fruits are classified based on the features such as height, diameter, weight and color code. A dataset with sufficient features and corresponding label is passed to the machine learning algorithms. Various machine learning algorithms are used for the purpose of comparison such as logistic regression, decision tree, k-nearest neighbor, linear discriminant analysis, Gaussian naive bayes and support vector machine. Further, the dataset is divided into train set and test set. Size of train set is more than test set because more the train sample then machine will learn more features from dataset and give accurate prediction. In this methodology, 70% dataset divided in train set and 30% dataset in test set to learn more features from train set and predict accurate results.

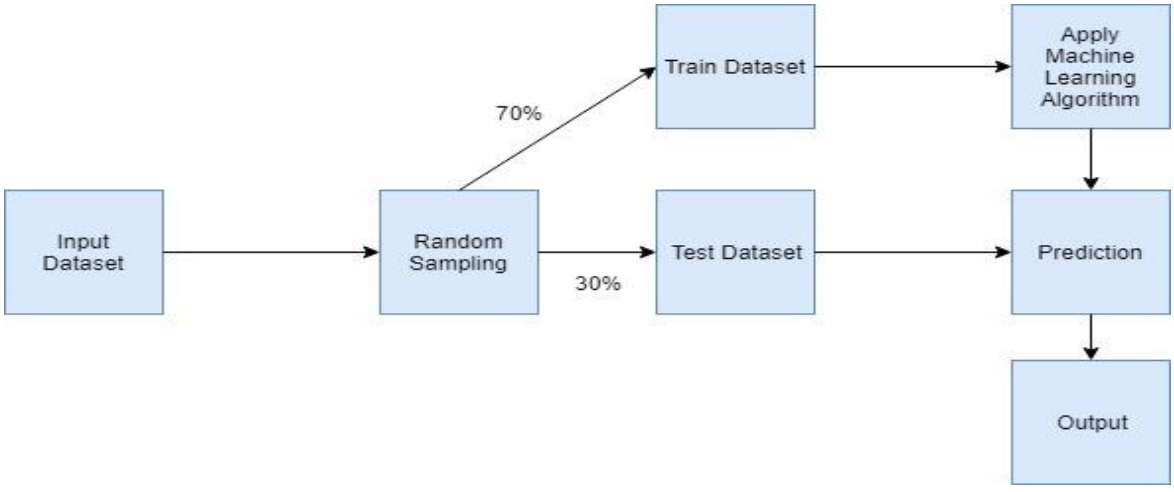


Fig 4.1: Flow chart for classification of Fruits using Machine Learning

4.1.1 MACHINE LEARNING (ML) ALGORITHMS

ML algorithms can learn from available data and improve accuracy from experiences, without human intervention. Learning tasks may include learning the function that maps the input to the output. In this project for classification of fruits dataset is available. Hence, supervised machine learning algorithms are used. Algorithms predict the outcome of a given test image, where the output is in the form of different categories of fruits.

4.1.1.1 Decision tree algorithm

A decision tree algorithm represents a procedure for classifying categorical dataset on basis of their own attributes.

Input: 'S' represents set of dataset, 'A' represents set of attributes, 'C' represents set of classes in 'S', 'T' represents subset created from splitting set by attribute 'A'

Output: Constructed decision tree

Algorithm: Decision Tree

1: Compute the entropy for dataset 'S'

$$H(S) = \sum_{c \in C} -P(C) log_2 P(C)$$

2: For every attribute:

2.1: Calculate entropy for all categorical values

2.2: Calculate information gain for the current attribute

$$IG(A,S) = H(S) - \sum_{t \in T} -P(t) log_2 P(t)$$

3: Choose the highest gain attribute as root node.

4: Repeat step 2 and 3 until get whole desired tree.

4.1.1.2 K-Nearest Neighbors algorithm

The K-Nearest Neighbors algorithm estimates how likely a test data point is to be a member of one group or another. It looks at the data points around a test data point to determine what group it is actually in. This algorithm is used to predict the outcome of a given test image where the output is in the form of different categories of fruits.

Input: A training sample S= (x_1, y_1), (x_2, y_2)……..(x_m, y_m), 'm' represents number of training sample.

Output: For every point x ∈ χ, return the majority label among { $y_{\pi(x)} : i \leq k$}

Algorithm: K-Nearest Neighbors

1. Initialize the value of K

2. for i = 1 to m,

   2.1: Calculate Euclidean distance between the points

   $$\rho(x, x') = \| x - x' \| = \sqrt[2]{\sum_{i=1}^{d} (x_i - x_i')^2}$$

   2.2 Arrange all Euclidean distance to x in increasing order.

   i.e. for all i < m, $\rho(x, x_{\pi i(x)}) \leq \rho(x, x_{\pi i+1(x)})$

   2.3 Get top k rows from the sorted array

   2.4 Get the most frequent class of these rows

   2.5 Return the majority label among $\{y_{\pi i(x)} : i \leq k]$

4.1.1.3 Gaussian Naive Bayes

A Naive Bayes Classifier is a supervised machine-learning algorithm that uses the Bayes' Theorem, which assumes that features are statistically independent. The theorem relies on the 'naive' assumption that input variables are independent of each other, i.e. there is no way

to know anything about other variables when given an additional variable. This is the probabilistic algorithm, which means that it calculates the probability of each tag for a given test sample, and then output the tag with the highest one. Probability is obtained by using Bayes Theorem, which describes the probability of a feature, based on prior knowledge of conditions that might be related to that feature.

Input: 1. 'x' represents features in the dataset

2. 'Ck' represents labels in the dataset

Output: Choose the label which has highest probability score

Algorithm: Gaussian Naive Bayes

1: Calculate the probability that the vector belongs to each class $P(Ck \mid x_1, x_2 \ldots x_n)$

where $P(Ck \mid x) = \frac{(P(x \mid Ck) * P(Ck)}{P(x)}$

2: Choose the class which has highest probability score

4.1.1.4 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is used for dimensionality reduction technique in the pre-processing step for classification and machine learning applications. The goal is to project a dataset into a lower-dimensional space with good class-separability in order avoid overfitting.

Input:

- 'm_i' represents mean vector of different classes
- 's_w' represents with-in class scatter matrix
- 's_i' represents scatter matrix of every class

- 'N_i' represents sample sizes of the respective classes,

Output:

Algorithm: Linear Discriminant Analysis

1: Compute the d- dimensional mean vector for the different classes from the dataset

2: Compute within-class and the between class scatter matrix ,

$S_w = \sum_{i=1}^{n} s_i$

$S_i = \sum_{x \in Di}^{n} (x - mi)(x - mi)^T$

$mi = \frac{1}{ni} \sum_{x \in Di}^{n} x_k$

$sb = \sum_{i=1}^{c} Ni\ (mi - m)(mi - m)^T$

where, m = overall mean

3: Compute eigen vector (e1, e2…...ed) and corresponding eigenvalues (λ1, λ2…λn)

4 : select linear discriminant for the new feature subspace

4.1: Sort the eigenvector by the new feature subspace

4.2: Choose 'k' eigen vectors with largest eigen values

5: Transform the sample into the new subspace

4.2 CLASSIFICATION OF FRUITS USING DEEP CONVOLUTION NEURAL NETWORK

Machine learning requires features and labels to learn from the dataset .Further, these algorithms used to train the model and predict results. Extracting features, applying algorithm, training model and then predicting results is a time consuming process. Machine learning algorithms are unable to classify fruits under shadow, occluded by foliage, branches and during overlap amongst fruits. To overcome these drawbacks. Deep learning is a new technique use to learn and extract features from images by the help the convolution neural layer (CNN) layer. The time require to classify fruit using CNN is very less as compare to machine learning approach.

CNN are complex feed forward neural networks. CNNs is used for image classification and object recognition. A ConvNet consists of an input and an output layer, as well as multiple hidden layers in between input layer and output layer. The hidden layers of a CNN typically consist of convolutional layers, pooling layers, fully connected layers and normalization layers. Convolutional layers apply a convolution operation to the input and pass result to the next layer. Pooling means that combines the outputs of neuron clusters at previous layer into a single neuron in the next layer. Generally, max pooling is used to extract maximum value or pixels from each of a cluster of neurons at the prior layer. Fully connected layers connect all neuron in one layer to all neuron in another layer.

To construct our own convolution neural network model for our own dataset is time consuming as well as utilize more CPU memory. To avoid this, in this proposed work Vgg-16, a pre-trained model trained on weights of 'ImageNet' is used. This model is used for prediction, feature extraction, and fine-tuning of own dataset which include images of fruits. Fine-tuning is a new approach that replace and retrain the trained convolution neural network for available dataset.

4.2.1 Vgg16 Model

VGG-16 is a deep convolutional network model for object recognition developed and trained on ImageNet dataset. The network characterized by the fact that it uses a simple 3x3 convolutional layer stack on top of each other in increasing depth. The number '16' represents the number of weight layers of the neural network.

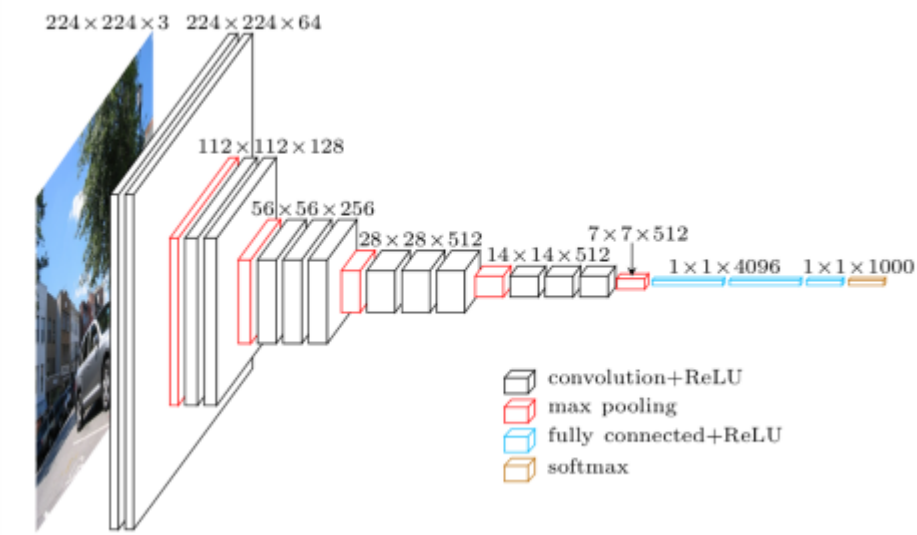


Fig 4.2: VGG- 16 architecture

During training, the input to the CNN is a fixed size 224 x 224 RGB image. The mean RGB value subtracted from each pixel of the images computed on the training set. The image is then pass through a collection of convolutional layers, where filters with a very small receptive field- typically 3 x 3- are used. In the VGG 16 Model, the convolution stride was fixed to 1 pixel, the spatial padding of convolutional layer input was such that the spatial resolution was preserved after convolution, i.e. the padding was 1 pixel for 3 × 3 conv. layers. Spatial pooling carried out by five max-pooling layers, which follow some of the convolutional layers, where not all the convolutional layers had followed by max pooling. Max pooling was performed over a 2 × 2 pixel window, with stride of 2. Three Fully-Connected (FC) layers followed a stack of convolutional layers. The final layer was the soft-max layer. The configuration of the fully

connected layers was the same in all networks. All hidden layers were equipped with the rectification non-linearity. The training carried out by optimizing the multinomial logistic regression objective using mini-batch gradient descent with momentum or the acceleration constant. The batch size is set to 256. The training is regularize by weight decay and dropout regularization for the first two fully connected layers (dropout ratio set to 0.5). The learning rate is initially set to 10^{-2} and then decreased by a factor of 10 when the validation set accuracy stopped improving. In total, the learning rate decreased by 3 times, and the learning was stopped after 74 epochs because highest accuracy is achieved at 74 epochs.

4.2.2 Fine-tuning of VGG-16 model

In this proposed work image classification for own dataset has done by fine-tuning the pre-trained Vgg-16 model which is trained on weight of 'ImageNet'. In the fine-tuning process, all weights are changed when trained on the new dataset except the weights of the last layers for the original dataset. First, define fully connected layer and load the ImageNet pre-trained weight to the model. Further, truncate the original softmax layer and replace it with own number of class labels for classification task. In this proposed work number of class labels are 12. Freeze the weight for the first few layers in network so that they remain intact throughout the fine-tuning process. Further, perform fine-tuning of layers by minimizing the cross-entropy loss function using stochastic gradient descent (sgd) algorithm. Load own dataset, split it into training and testing sets, decide number of batch size and epocs, start fine-tuning the model and save model in format of file with extension '.h5'. Last step is to write separate program in python for prediction of fruit in which the saved model is use for prediction of fruit from images.

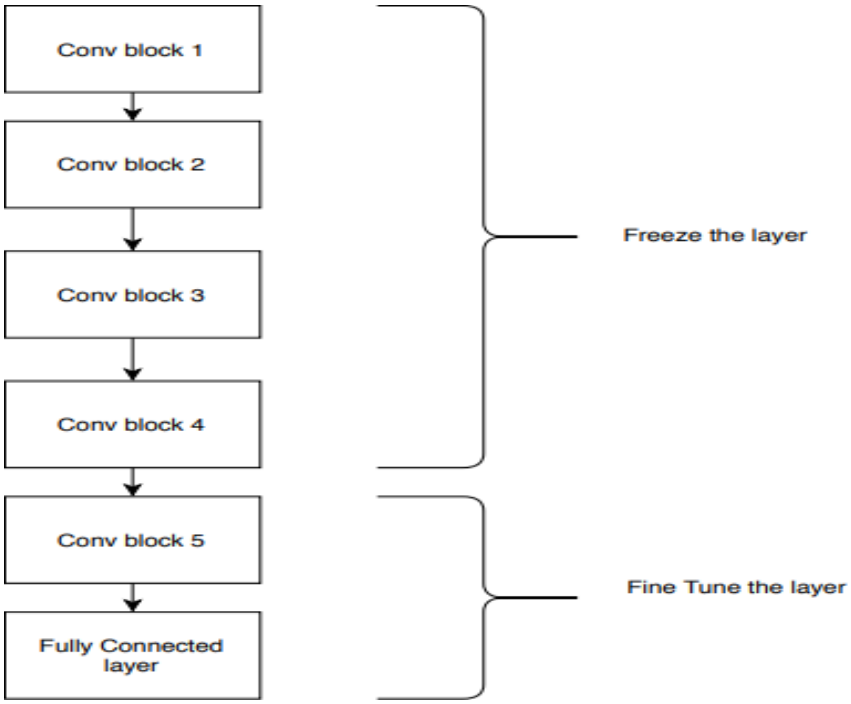


Fig 4.3: Architecture diagram for Fine-tuning of Vgg-16 network

4.2.2.1 Algorithm for fine-tuning of Vgg-16 model

Input : Vgg-16 model

Output: A new trained model which is trained on our own dataset

Algorithm: Fine-tuning of Vgg-16 model

1: Define the fully connected layer

2: Load the ImageNet pre-trained weight

3: Truncate the last layer i.e. softmax layer of the vgg-16 network and and replace it with new softmax layer of 12 categories.

4: Freeze the weights of first few layers of the pre-trained network.

5: Fine-tune the model by minimizing the cross entropy loss function using stochastic gradient descent (sgd) algorithm and set smaller learning rate to train the network.

6: Load our own dataset, split it into training and testing sets, and start fine-tuning the model by setting value of batch size and number of epoch.

7: Save model with '.h5' extension

8: Stop.

4.3: DETECTION AND COUNTING OF FRUITS

The accurate detection of fruits is achieved by using computer vision techniques. First images are taken from dataset is converted into Hue Saturated Value (HSV) image. Further, morphological operations are performed on HSV image to remove nosie such as branches, leaves, sky and soil from image. In this proposed work morphological operations such as mask-open and mask-close is performed. Mask-open operation removes small dots popping randomly on the masked image. Mask-close operation is used to remove the holes present on the output of the mask-open operation. Further, contouring i.e. curve that joins all the continuous point of the same color is done on previously obtained mask-close image. Last step of the process is to construct a rectangle around contour obtained in the previous step. Count of rectangles in the output image shows count of fruits in the input image. Sequential flow of process is shown in Figure 4.4

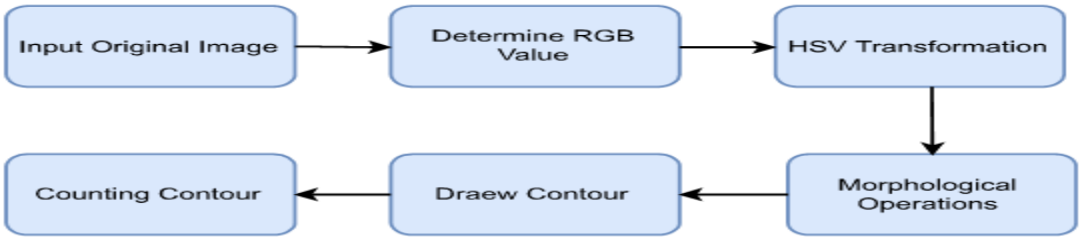


Fig 4.4: Flowchart for Detection and Counting of on tree Fruits.

4.3.1 Algorithm for detection and counting of fruit

Images and videos from the dataset are send to the algorithm to perform image-processing operations. Operations such as image pre-processing, noise removal, image segmentation, contouring is done sequentially and algorithm for the same is as follows:

Algorithm: Detection and count fruit

Input: Set of 'N' images from dataset having width 'W' and height 'H'

Output: Number of fruit count in the image.

1.Take input image from dataset

2. Resize image: (W * 0.5) X (H * 0.5)

3. Convert RGB image into HSV image.

The R,G,B values are divided by 255 to change the range from 0…..255 to 0…....1:
R' = R/255
G' = G/255
B' = B/255
Cmax = max(R', G', B')
Cmin = min(R', G', B')
Δ = Cmax - Cmin

Hue calculation:

$$H = \begin{cases} 0° & \Delta = 0 \\ 60° \; X \; \left(\frac{G'-B'}{\Delta} \; mod \; 6\right), & Cmax = R' \\ 60° \; X \; \left(\frac{B'-R'}{\Delta} + 2\right) & , Cmax = G' \\ 60° \; X \; \left(\frac{R'-G'}{\Delta} + 4\right) & , Cmax = B' \end{cases}$$

Saturation calculation:

$$S = \begin{cases} 0 \, , & Cmax = 0 \\ \frac{\Delta}{Cmax} \, , & Cmax \neq 0 \end{cases}$$

Value calculation:

V = Cmax

4. Set threshold value of HSV: [0,360]

5. Compare obtained threshold value of HSV image with set threshold value

Step 5.1. if obtained threshold value > set threshold value

Step 5.2. Detect object

Step 5.3. else reject object.

6. Apply contour to detected object

7. Count number of counter

8. Display output image

CHAPTER - 5

EXPERIMENTAL WORK

This chapter discuss the experimental setup and experiments performed during the project work.

5.1 EXPERIMENTAL SETUP

All the experiments are conducted on the following configuration:

Hardware	Software
System:- Intel core i7	Operating system:- Linux(Ubuntu)
Hard disk:- 250 GB	Language:- Python
RAM:- 8 GB	Tool:- Anaconda

Table 5.1: Hardware and Software Setup

5.2 EXPERIMENTS

5.2.1 Experiment 1: Classification of fruit using machine learning algorithm

Machine learning is a technique that learn from features provided to it and predict accurate results. In this experiment, fruits are classified based on the features such as height, diameter, weight and color code. A dataset with sufficient features and corresponding label is passed to the machine learning algorithm such as logistic regression, decision tree, k-nearest neighbor, linear discriminant analysis, Gaussian naive bayes and support vector machine. Further, the dataset is divided into train set and test set. Size of train set is more than test set because more the training sample then machine will learn more features from dataset and give accurate prediction. In this

methodology, 70% dataset divided in train set and 30% dataset in test set, to learn more features from train set and predict accurate results.

5.2.2 Experiment 2: Classification of fruit using deep neural network

A. Fine-tuning of convolution neural layers

Generally, fine-tuning operation can be performed on model such as Vgg-16,ResNet, Inception-v3, AlexNet, GoogleNet and Keras, Tensorflow, Theano is used as backends. In this experiment fine-tuning of convolution neural layers is performed on Vgg-16 model and Keras is used as backend. Fine-tuning is a process to take a network model that has been already trained for a some particular task, and make use of it to perform a second similar task. This is achieved by replacing the weights of some convolution layers except last output layer of the Vgg-16 architecture.

Experiment begins with defining fully connected layer and load the ImageNet pre-trained weight to the Vgg-16 model .Further, truncate the original softmax layer and replace it with own number of class labels for classification task. In this experiment number of class labels are 12 because, twelve different categories of fruits such as, orange, lemon, pineapple, banana, strawberry, pomegranate, granny smith apple, blueberry, cherry, peach, raspberry and walnut are used for training. Further, freeze the weight for the first few layers in network so that they remain intact throughout the fine-tuning process. Perform fine-tuning of layers by minimizing the cross-entropy loss function using stochastic gradient descent (sgd) algorithm. Next step is to load own dataset, split it into training and testing sets, decide number of batch size, epocs and further start fine-tuning the model. Fine-tuning of convolution layer is trial and error process. Train the model until highest accuracy is achieved. In this experiment model is trained for batch size equal to 110 and 20 epoch. Figure 5.1 shows screenshot of comparative analysis of obtained results such as training loss, training accuracy, validation loss, validation accuracy obtained during each training phase of experiment.

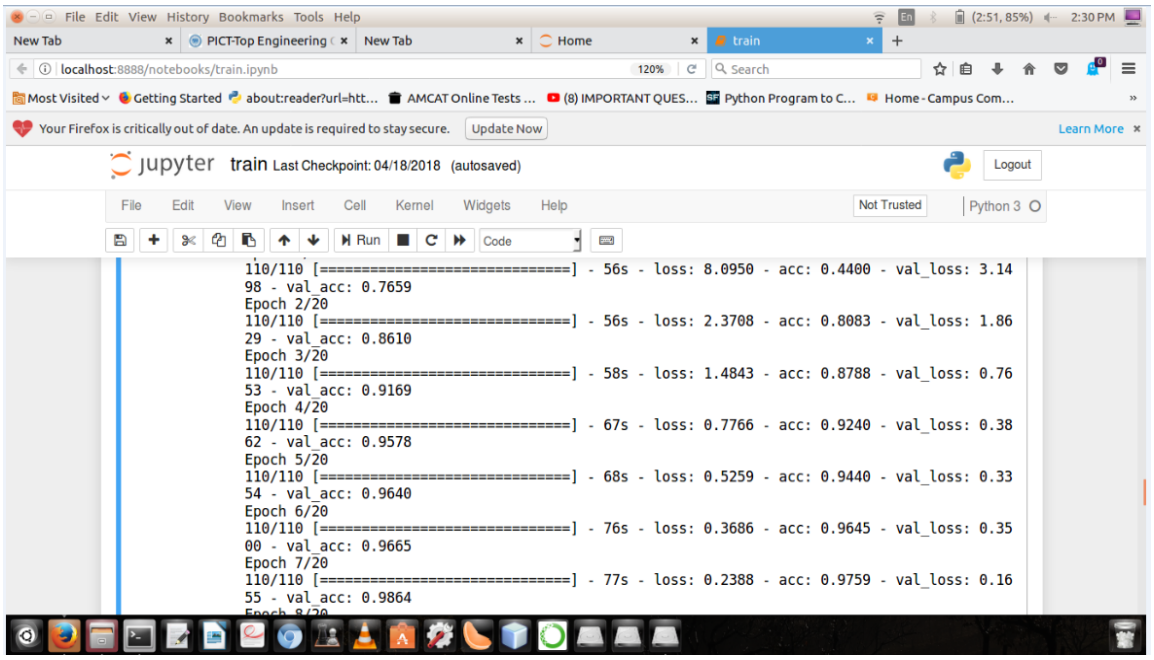


Fig 5.1.1: Experimental results obtained at each training phase.

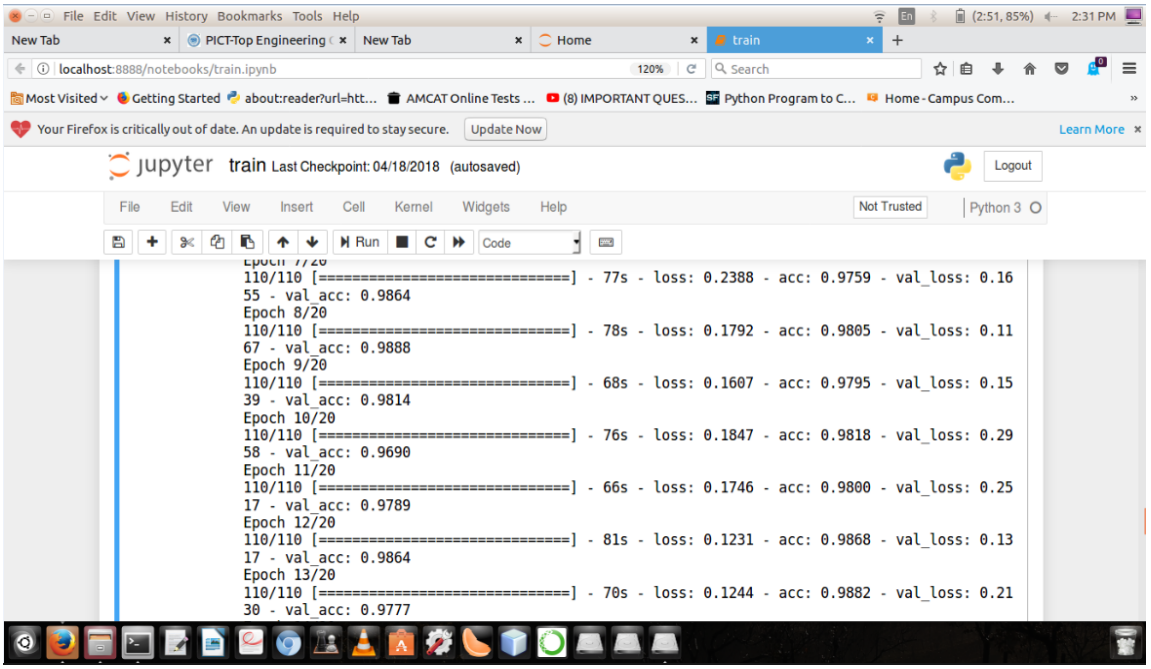


Fig 5.1.2 : Experimental results obtained at each training phase.

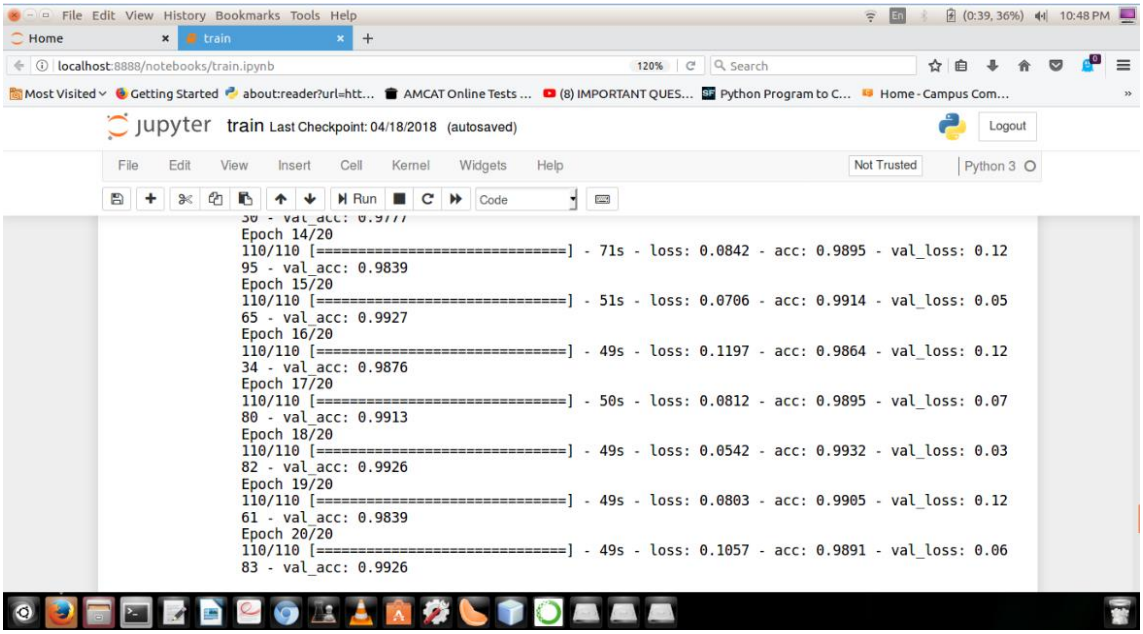


Fig 5.1.3: Experimental results obtained at each training phase

5.2.3 Experiment 3: Image pre-processing

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. In this experiment all input RGB image and videos is converted into HSV image and video. Color vision can be processed using RGB color space or HSV color space. RGB color space describes colors in terms of the amount of red, green, and blue present. HSV color space describes colors in terms of the Hue, Saturation, and Value. Color description plays an integral role, for separation of object from background of the image. The HSV model describes colors similarly, to how the human eye tends to perceive color. RGB defines color in terms of a combination of primary colors, where as, HSV describes color using more familiar comparisons such as color, vibrancy and brightness. Hence, the HSV color model is often preferred over the RGB model. Following figure 5.2 shows input image on which pre-processing operation are done to extract required features from the image.

Fig 5.2: Input image

For object and color based segmentation input RGB image is converted to HSV image. Figure 5.3 shows HSV image obtained after preprocessing operation on RGB image.

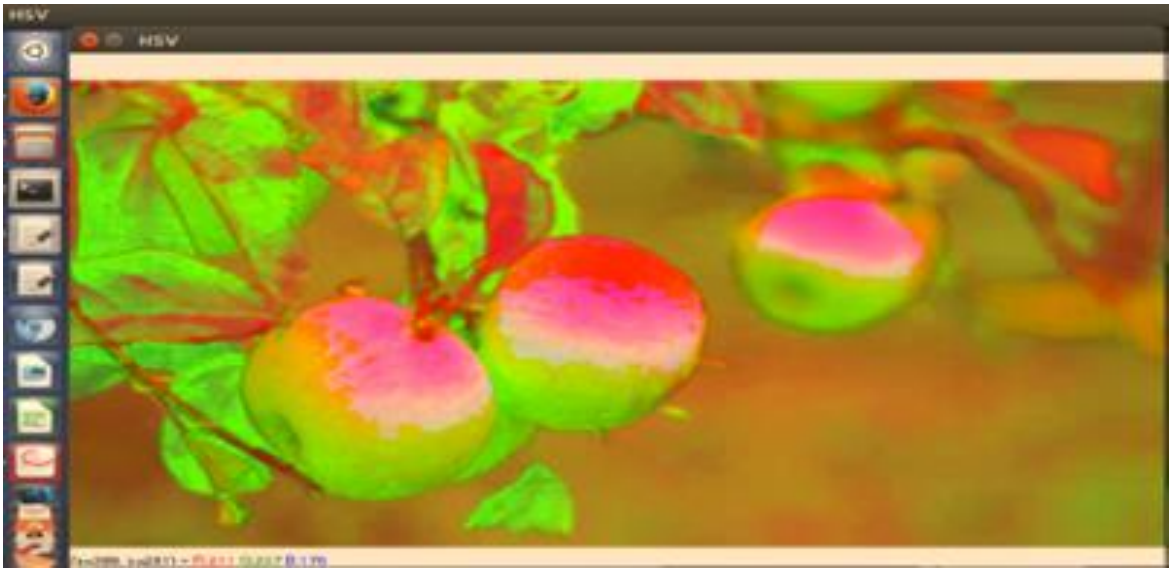


Fig 5.3: HSV image

5.2.4 Experiment 4: Morphological operation

Morphological transformations are some simple operations normally performed on binary images. It needs two inputs, one is our original image, second one is called ‘structuring element’ or ‘kernel’which decides the nature of operation. Two basic morphological operators are Erosion and Dilation. Other variant forms of morphological transforms are ‘opening’ and ‘closing’ which are also perform on image to extract shape of required object from the image. Structuring element used in this experiment is of size 5 X 5 which is smaller than regular input image. Erosion is done to remove background noises from the image. In this experiment a kernel of size 5 X 5 is move across whole image and detect require color from the whole image.

Erosion: Erosion operation erodes away the boundaries of foreground object. This operation always try to keep foreground in white. In this experiment the kernel slides through the image as in 2D convolution. A pixel in the original image either 1 or 0 will be considered 1 only if all the pixels under the kernel is 1, otherwise it is eroded and made to zero. All the pixels near boundary will be discarded depending upon the size of kernel. So the thickness or size of the foreground object decreases or simply white region decreases in the image. It is useful for removing small white noises so that will detach two connected objects. Following figure 5.4 shows result obtained by preforming erosion operation on HSV image.

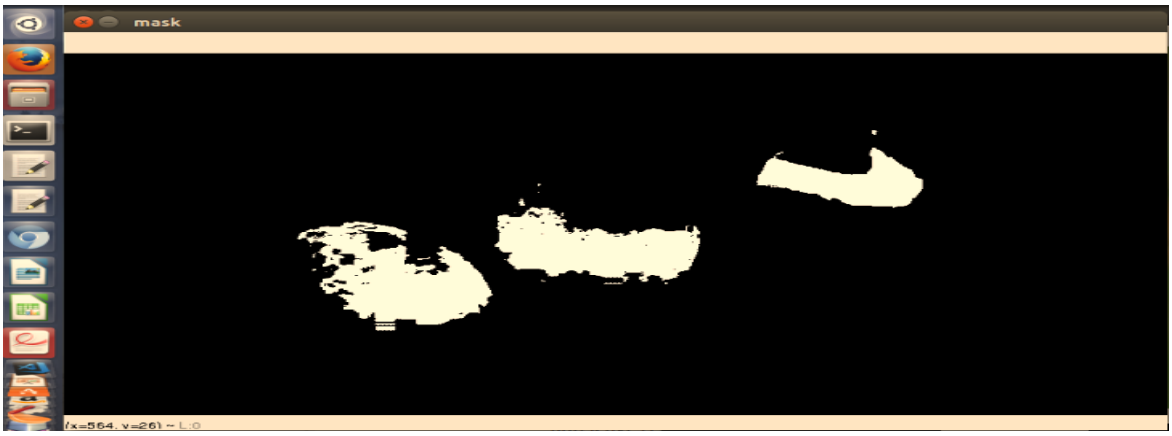


Fig 5.4: Mask image

5.2.5 Experiment 5: Opening Operations

Opening is just another name of erosion . It is useful in removing noise, In the previous experiment small dots are randomly popping in the image. In this experiment small dots are remove i.e background noise is remove by moving kernel of 5 X 5 across the whole image. Following figure 5.5 shows result obtained by preforming opening operation on mask image.

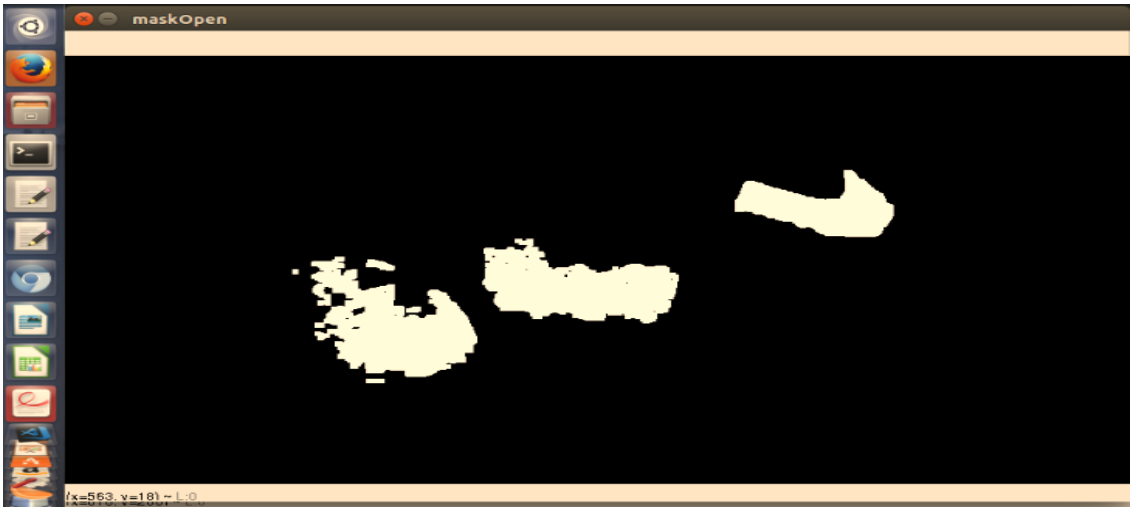


Fig 5.5: Mask open

5.2.6 Experiment 6: Closing Operation

Closing is reverse of opening. It is useful in closing small holes inside the foreground objects, or small black points on the object. In the previous experiment small holes are present in the actual object which is removed by moving kernel of size 20 X 20 across whole image. Following figure shows result obtained by preforming closing operation on mask open image.

Fig 5.6 : Mask close

5.2.7 Experiment 7: Contouring Operations

Curve that joins all the continuous point of the same color is called as contouring. Contour is used to denote object boundary. Contouring is done on the mask close image to detect accurate object. Following figure shows result obtained by preforming contouring operation on mask close image.

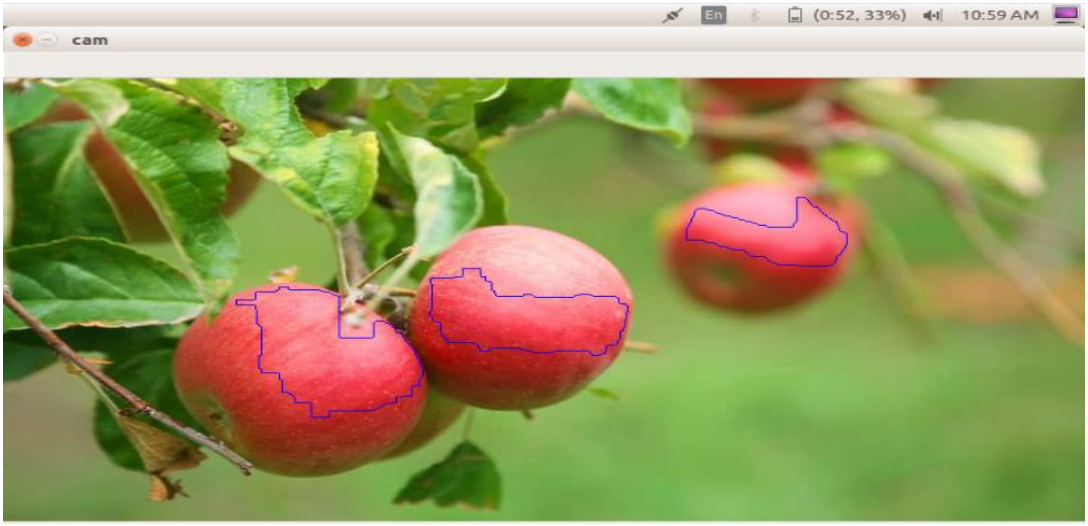


Fig 5.7: Contour operation

5.2.8 Experiment 8: Fruit Counting

Last step of the whole experiment is to count fruits present in the image and video. Construct a rectangle across boundary of contour obtained in the contouring experiment. Number of count of rectangles in the output image shows count of fruits in the input image.

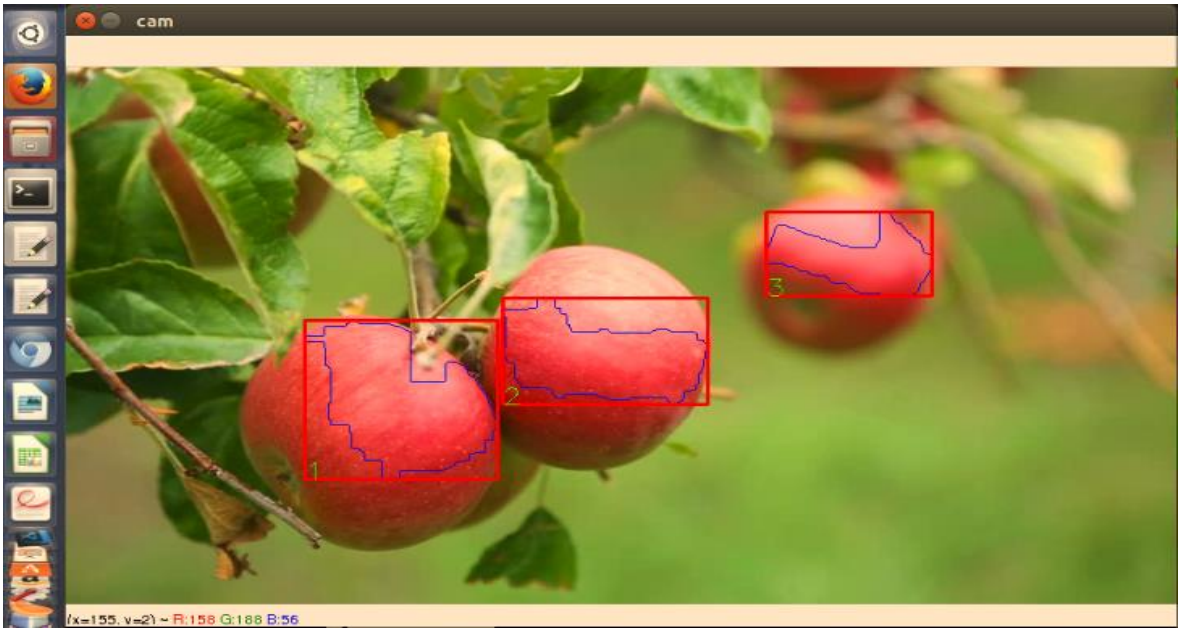


Fig 5.8: Count of fruits in an input image

All experiments concludes that all the operations such as image pre-processing, noise removal, image segmentation and contouring are performed sequentially with the help of code written in python. The various libraries that are used in python for the experiment are sklearn, opencv, numpy, scipy, matplotlib and pandas. The final output obtained is also an image or video which shows the detected fruit along with the fruit count. The execution time required to perform these experiment is around 3 second.

CHAPTER - 6

ANALYSIS PROCEDURE

Various experiments carried out and the setup required for the experiments is explained in detail in the previous chapter. The detailed explanation of analysis of proposed system is explained below in this chapter.

6.1 MACHINE LEARNING MODEL

Machine learning is a set of techniques, which learn from available dataset by developing algorithms or set of logical rules. Supervised machine learning algorithms are used because of the availability of the dataset as supervised algorithms need labeled dataset for classification. To train and test the machine learning model, supervised machine-learning algorithms like linear regression, decision tree, k- nearest neighbor (KNN), linear discriminant analysis (LDA), support vector machine (SVM), and gaussian navie bayes are used. The dataset is passed to all the algorithms mentioned above and then the train set accuracy and test set accuracy is calculated in percentage for each algorithm. Obtained accuracy of each algorithm is shown in the table given below.

Accuracy/ Algorithm	Linear Regression (LR)	Decision Tree (DT)	K-Nearest Neighbors(KNN)	Linear Discriminate analysis (LDA)	Support Vector Machine (SVM)	Gaussian Navie Bayes (GNB)
Train set	70%	100%	99.8%	86%	61%	86%
Test set	40%	73%	89%	67%	33%	67%

Table 6.1: Comparative analysis of supervised machine-learning algorithms

Comparative analysis of all machine learning algorithms is shown with the help of following graph:

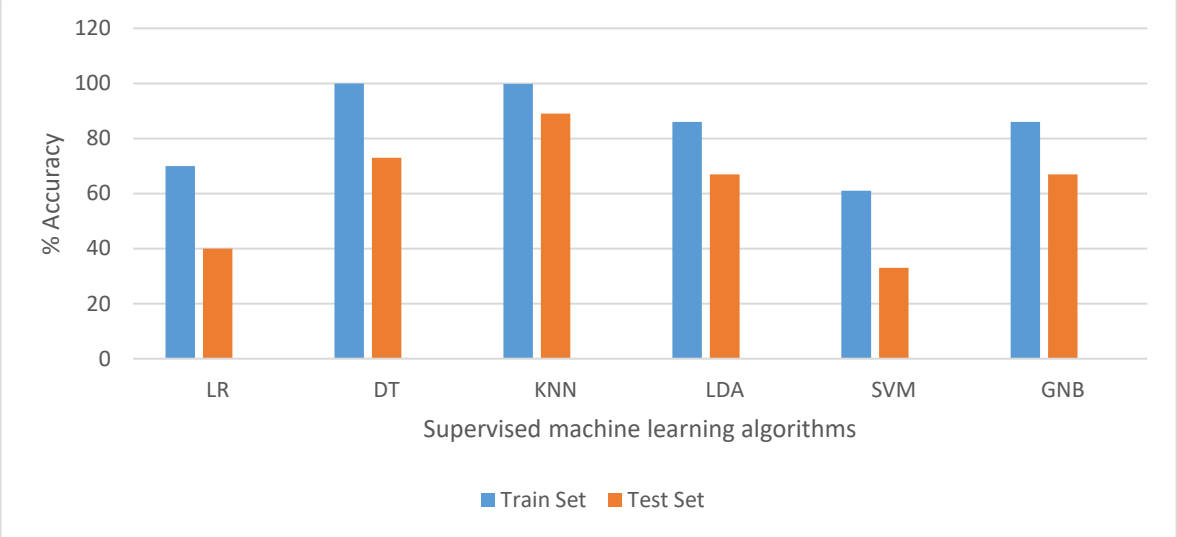


Fig 6.1: Comparative analysis of supervised machine-learning algorithms

In KNN algorithm, 'K' denotes number of neighbors present surrounding to the test sample. An object is classified by a majority vote of its neighbors. Further, train the algorithm by varying the value of K from 1 to 20 and determine accuracy of algorithm at these values of K. At K=5 algorithm chooses 5 nearest neighbors to predict the label or fruit category of the test data. By comparing the results obtained at various values of K, the following graph shows that the greatest accuracy is obtained at K=5.

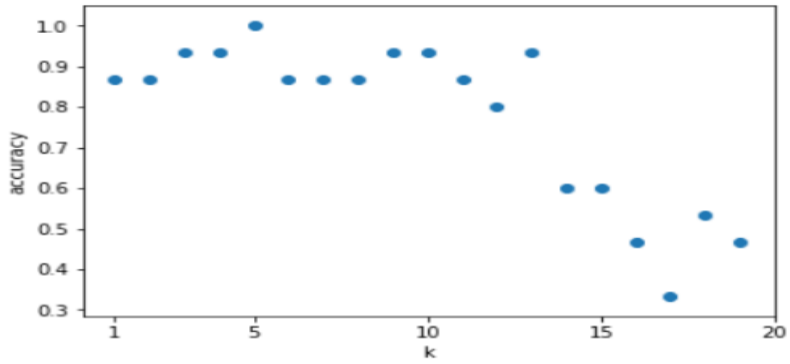


Fig 6.2: Accuracy obtained at various values of 'K'

6.2 DEEP NEURAL NETWORK MODEL

Machine learning algorithms give good results in the classification of fruits of different sizes. But its classification accuracy is affected when same shape and size of fruit is encounter. To overcome this drawback, deep learning technique is used which is good at classifying different fruits of the same shape and size. The convolution neural network (CNN) is trained and tested on our own dataset. Further, fine-tuning of convolution neural layers is done on Vgg-16 architecture. At the end of these operations, a new trained model is obtained. This trained model is used for classification of fruits even if fruits are under shadow, occluded by foliage, branches, or if there is some degree of overlap amongst fruits, which was not possible earlier with the supervised machine learning algorithms.

Training of model is done until get the highest accuracy for the model. During training phase, values of both batch size and number of epoch vary to achieve better accuracy. Number of epoch is equal to number of times neural layers is going to be trained. More features can be extracted by training neural layers more number of time. As the number of epoch increases, training accuracy increases because during each phase of training more number of features are extracted by convolution neural layers. Following graph shows training and validation accuracy occurred at various epocs ranging from 1 to 20. The CNN training has stopped after 20 epocs, because training accuracy is achieved near to 100%.

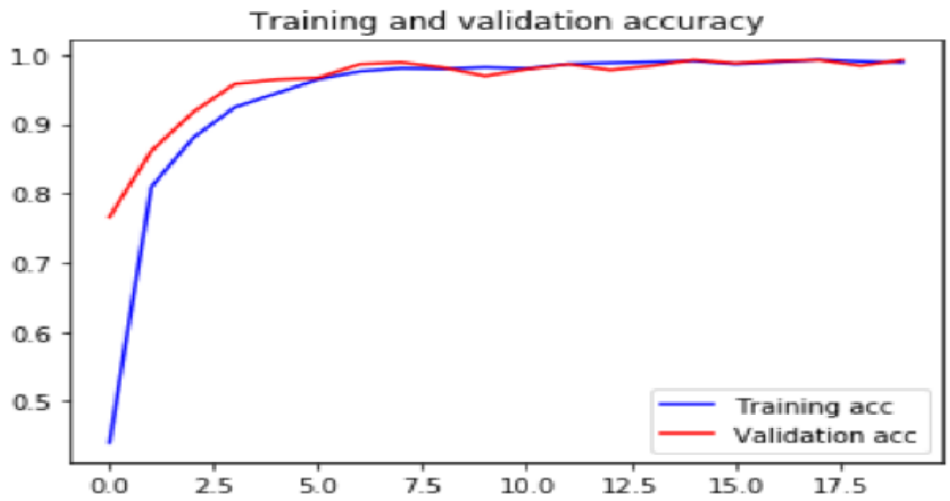


Fig 6.3: Graph of Training and Validation accuracy upto 20 epoch

During training phase dropout layer helps to reduce overfitting by preventing layer from seeing twice the exact same extracted feature. Hence, training and validation losses is going to be decreases as number of epoch increases. Following graph shows training and validation losses occurred at various epocs ranging from 1 to 20.The CNN training has stopped after 20 epoch, because training losses are achieved near to 0%.

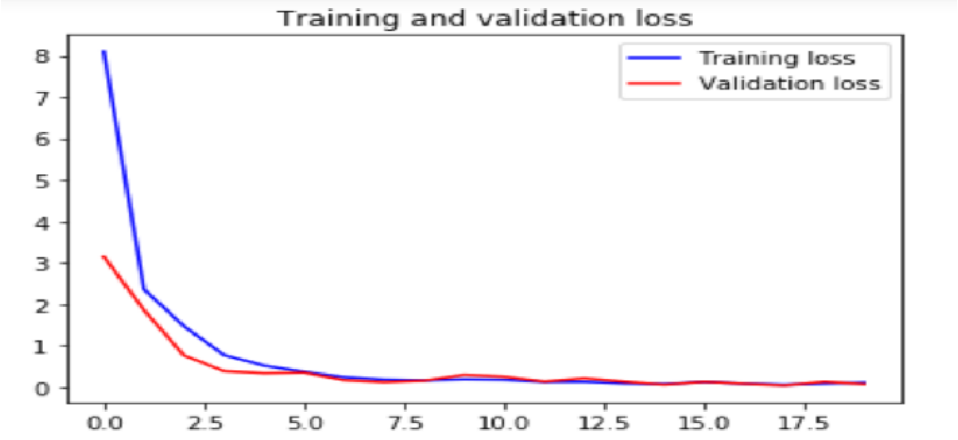


Fig 6.4: Graph for Training and Validation Loss upto 20 epoch

6.3 COMPARISION BETWEEN MORPHOLOGICAL OPERATIONS

Curve that joins all the continuous point of the same color is called as contouring. Contouring is done to detect accurate object from the binary image. If this operation performed on mask open image and construct contour then it will draw many contours on the image and give large number of count of fruits in the given input image and video. Following figure shows observation obtained by performing contour operation on mask open image.

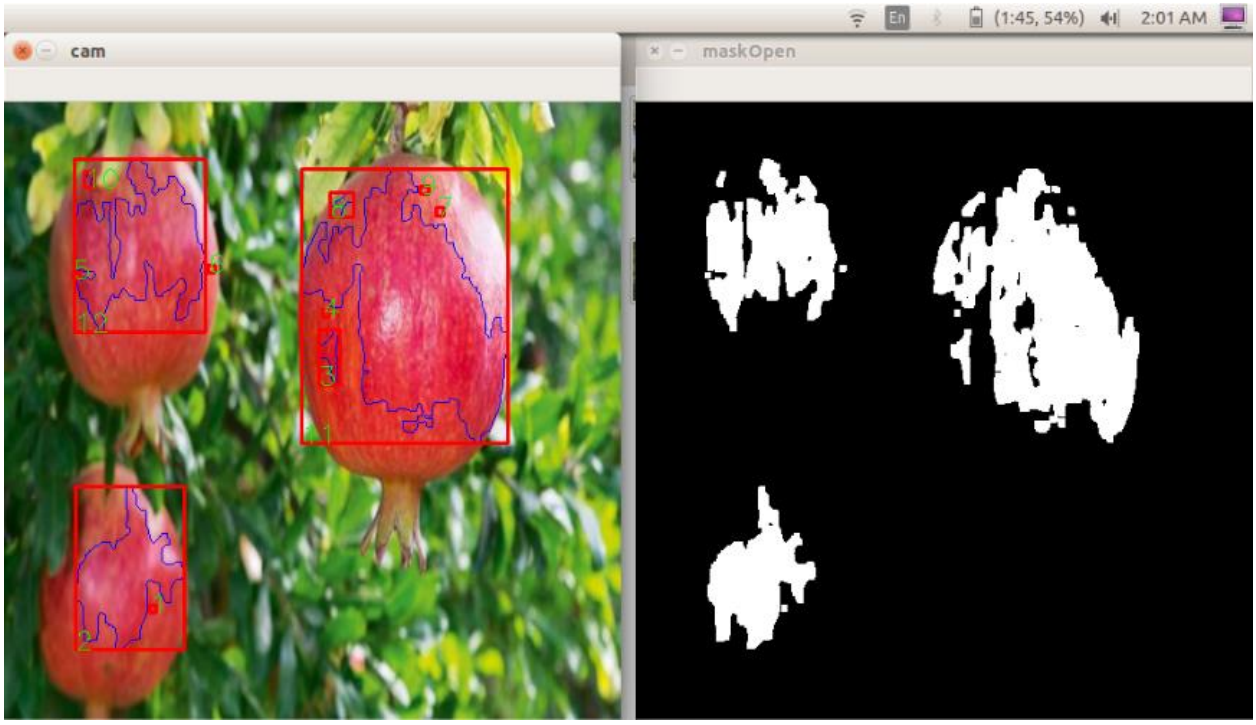


Fig 6.5: Wrong number of count of fruit in the image

Closing morphological operation closes small holes present in the foreground object. Thus, by applying contouring operation on mask close image gives accurate number of fruits present in the image and video. Following figure shows observation obtained by performing contour operation on mask close image.

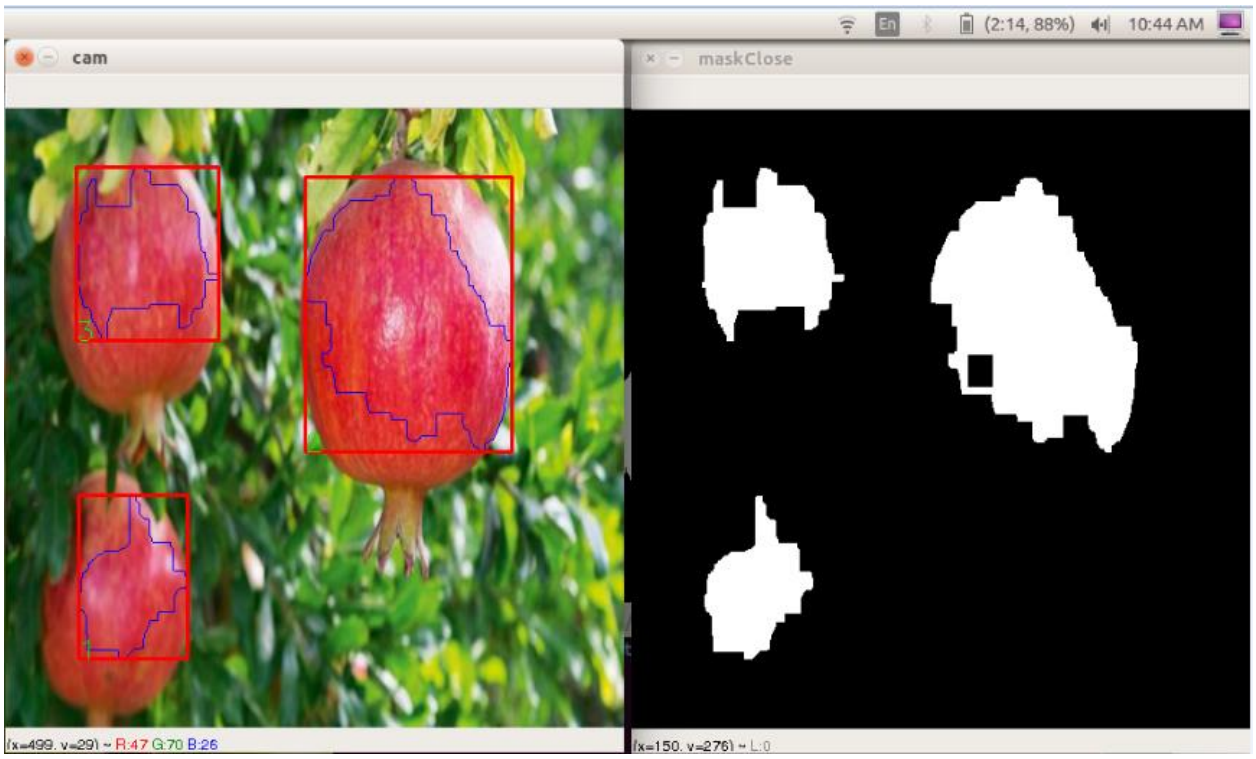


Fig 6.6: Accurate counting of fruit in the image

Thus, applying contouring operations on mask-close image gives accurate result as compare to contouring operations applied on mask open image.

CHAPTER – 7

RESULTS AND DISCUSSION

This chapter include screenshots of all the results obtained during experiments.

7.1 RESULTS FOR CLASSIFICATION OF FRUIT

Developed methodology consists of three steps. In the first step, dataset consists of 12 categories of fruit images. Dataset of each fruit images is divided into training images and testing images. In the second step, dataset i.e. group of images are passed to the layer of convolution neural network. It is impossible to build own convolution neural network, because it takes a lot of time and consume lot of CPU memory and graphics. Therefore, a pre-trained model called Vgg-16 is used. A new model is created by fine-tuning of Vgg-16 model. In the last step, classification of fruit is done based on the new model created during the second phase. The new trained model classify fruits even if fruits are under shadow, occluded by foliage, branches, or if there is some degree of overlap amongst fruits. Following are the results obtained by using new trained model which classify fruits from image.

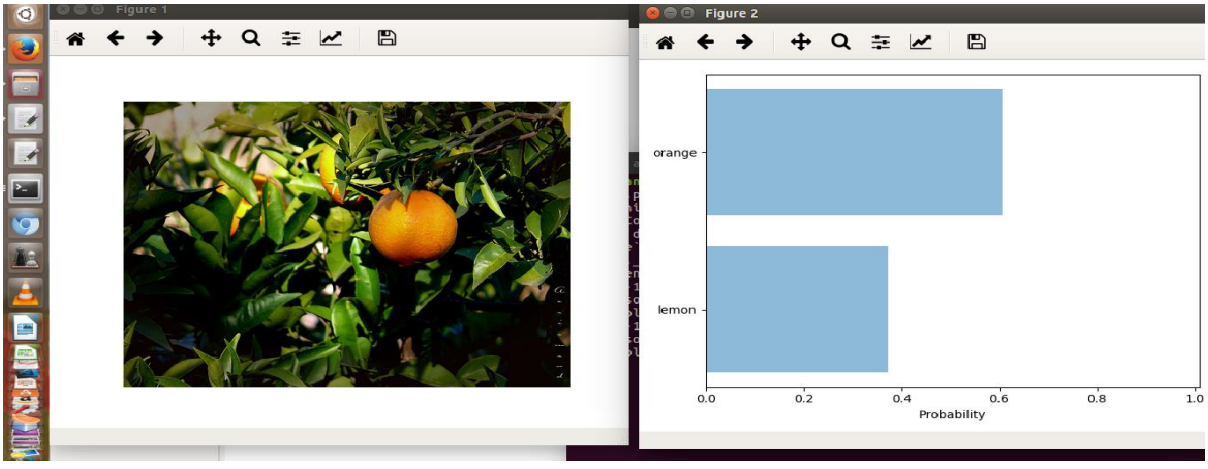


Fig 7.1: Prediction of Orange fruit

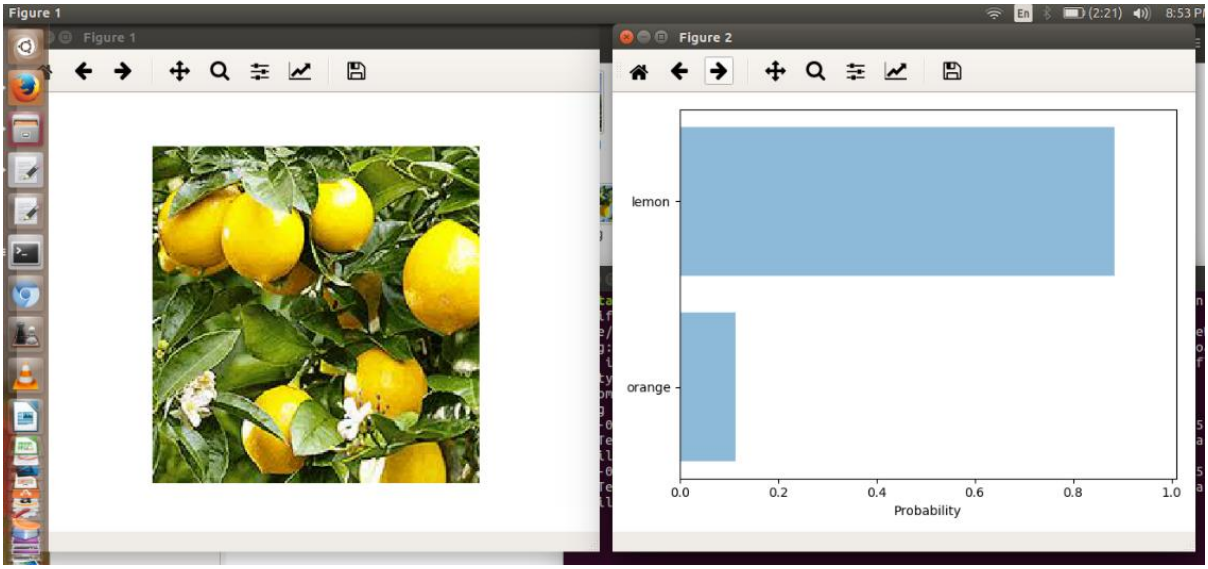


Fig 7.2 : Prediction of lemon fruit

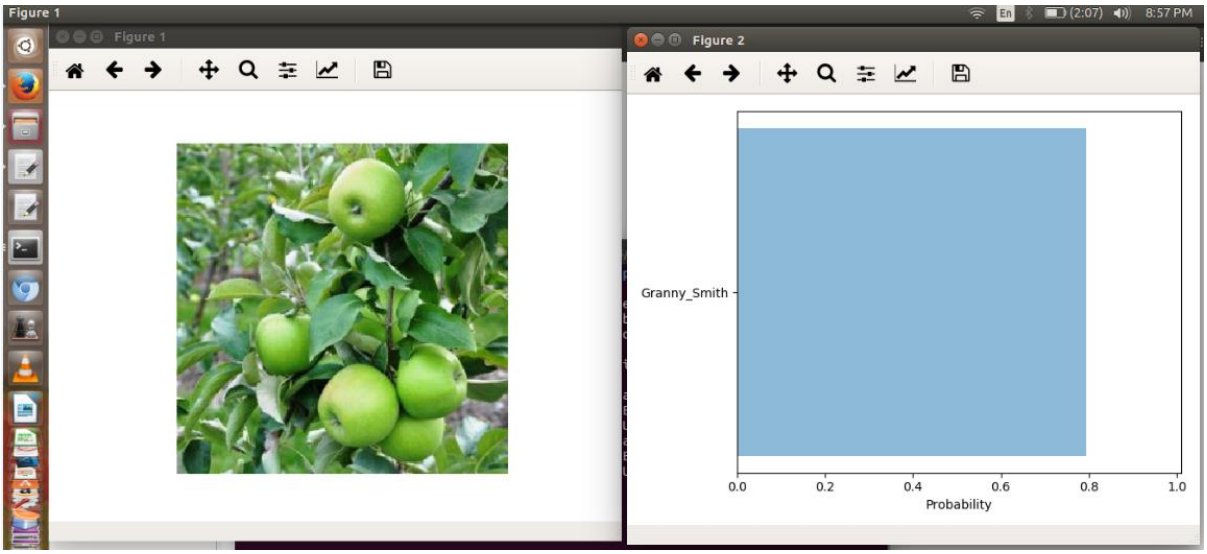


Fig 7.3 : Prediction of Granny-Smith Apple fruit

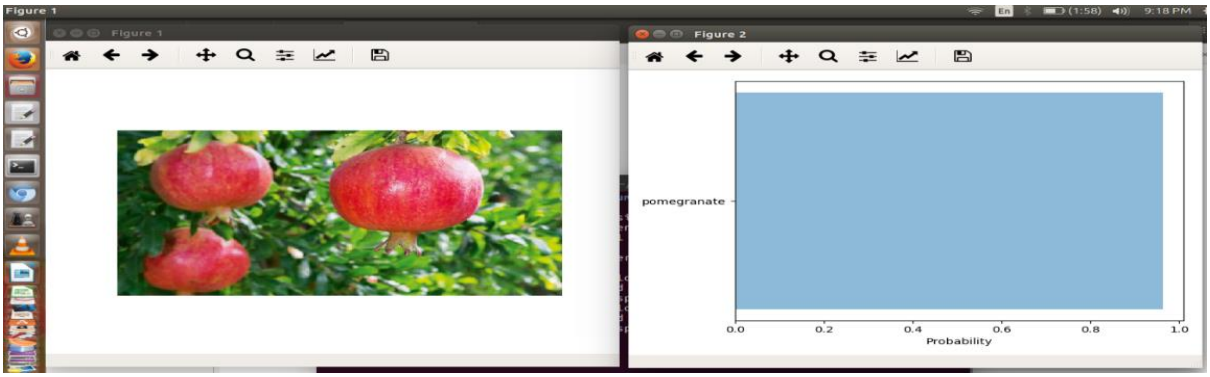


Fig 7.4: Prediction of Pomegranate fruit

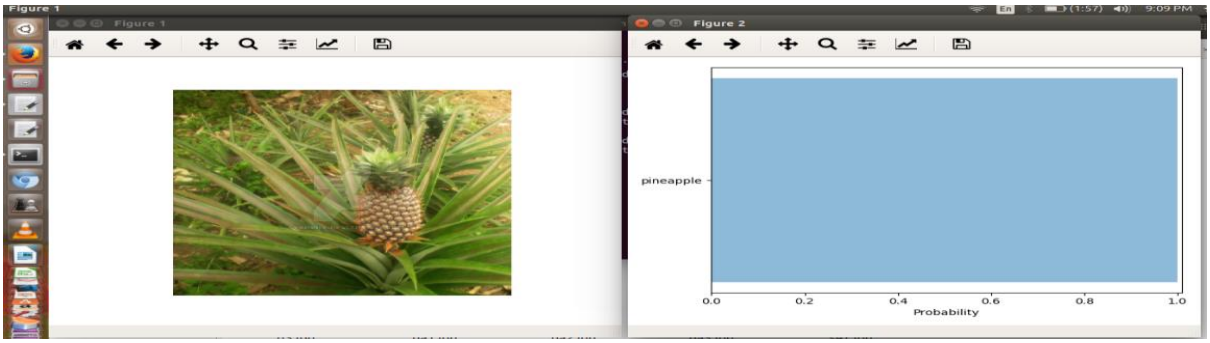


Fig 7.5: Prediction of Pineapple fruit

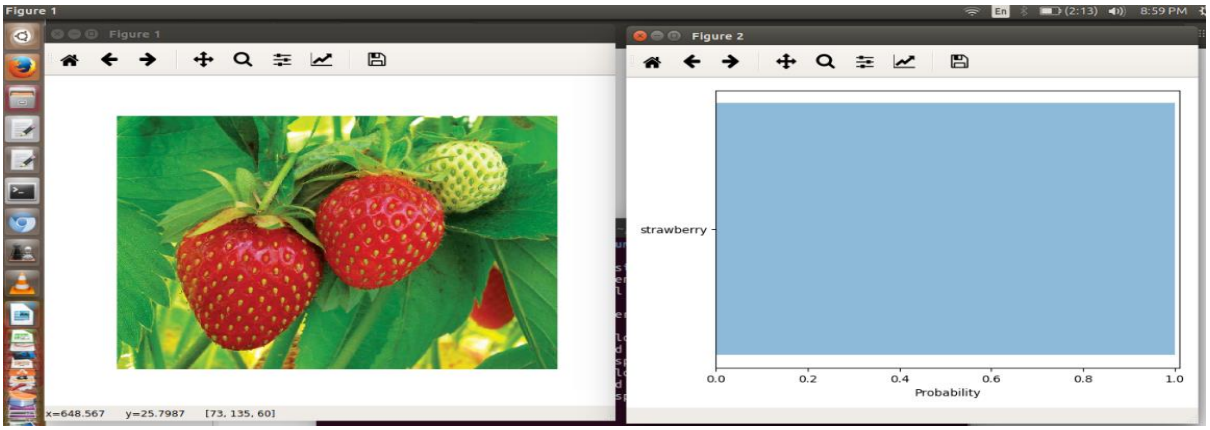


Fig 7.6: Prediction of Strawberry fruit

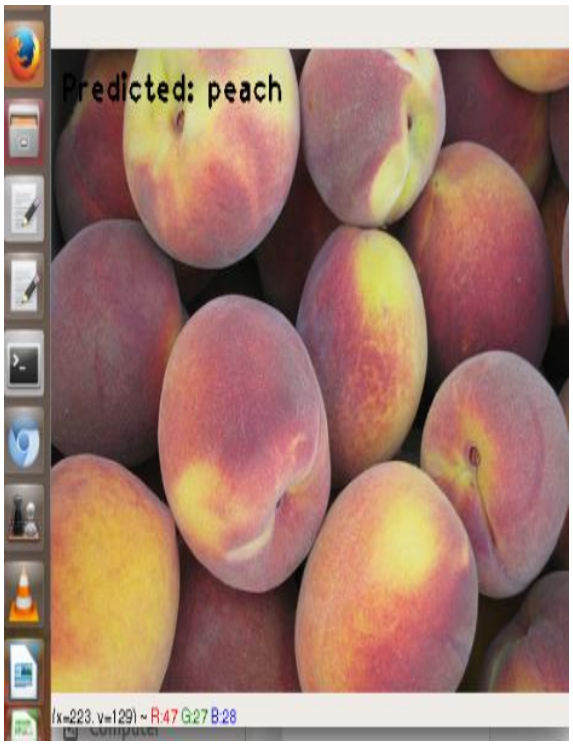


Fig 7.7 : Prediction of Peach fruit

Fig 7.8: Prediction of Walnut fruit

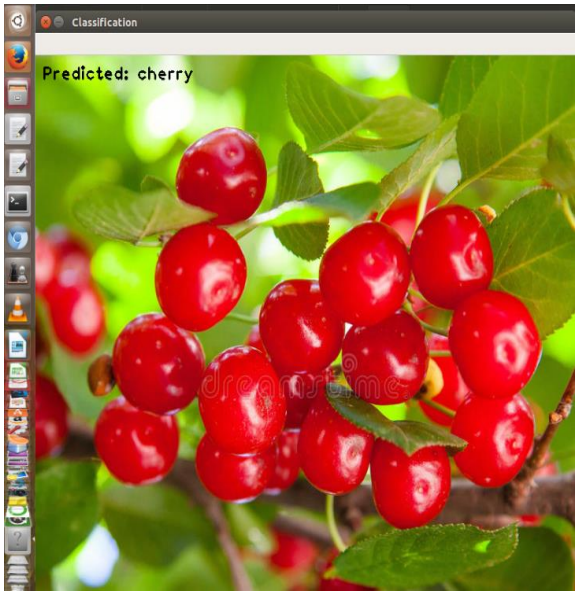


Fig 7.9: Prediction of Cherry fruit

Fig 7.10: Prediction of Blueberry fruit

Fig 7.11: Prediction of Raspberry fruit

7.2: RESULTS FOR DETECTION AND COUNTING OF FRUITS

Computer vision and image-processing techniques are together used to detect and count fruits in the image. Following images show the screenshots of the results obtained after performing experiments.

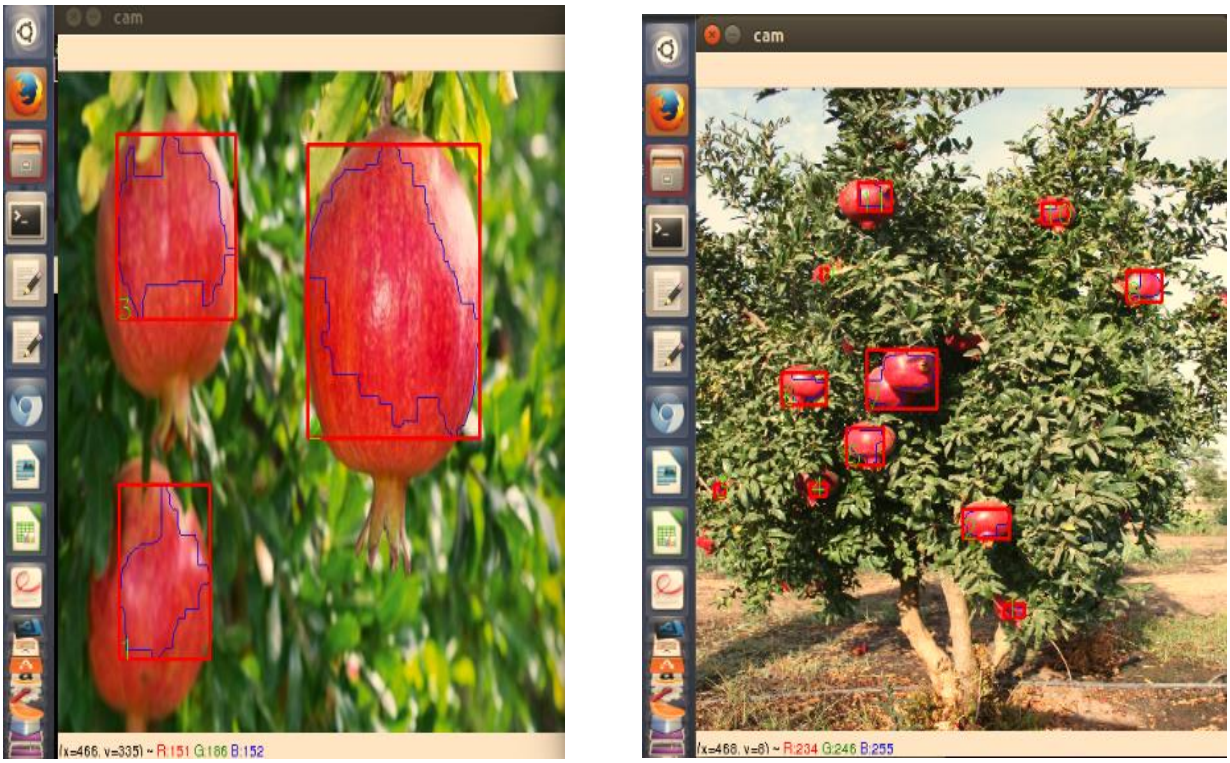


Fig 7.12: Pomegranate detection and Counting

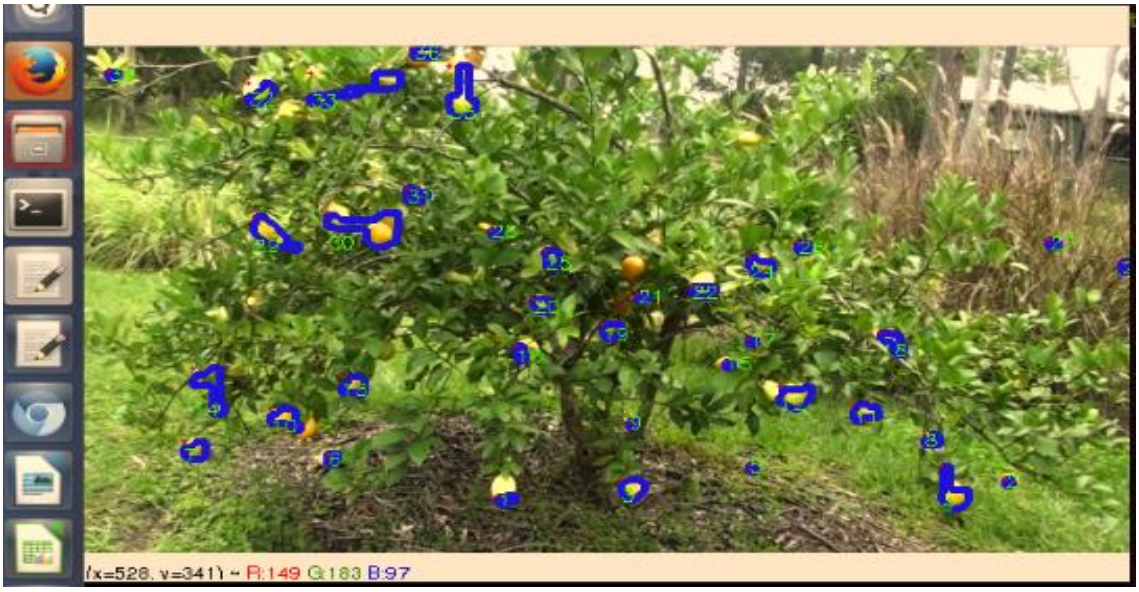


Fig 7.13: Lemon detection and counting

Fig 7.14: Orange detection and Counting

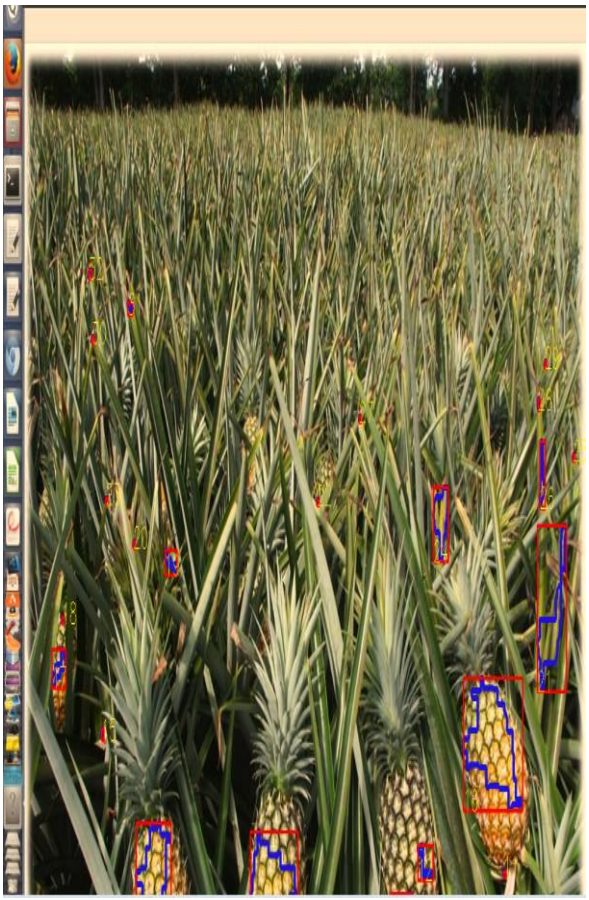


Fig 7.15 : Pineapple detection and Counting

CHAPTER – 8

CONCLUSIONS AND SCOPE FOR FUTURE WORK

. This proposed work classify fruit based on the features such as height, width, diameter and color code of fruit. All these features are passed to the supervised machine learning algorithms. Machine learning algorithms are unable to classify fruits under shadow, hidden by leaves, branches and during overlap amongst fruits. To overcome these drawbacks 'Deep Learning' a new technique has been used to learn and extract features from images by the help of the convolution neural layers. A new fruit classification model is created by the help of fine-tuning of 'Vgg-16' a pre-trained model. This proposed work classifies the 12 different categories of fruits viz; orange, lemon, pineapple, banana, strawberry, pomegranate, granny-smith apple, blueberry, cherry, peach, raspberry and walnut. 99.8% validation accuracy is obtained at 20 epochs during training phase of model. Image pre-processing, noise removal, image segmentation and contouring are used sequentially to detect and count the number of fruits present on the tree from the images as well as videos.

In future, this proposed system can be further extended to develop a mobile application that help farmers to classify, detect and count fruits. To extend this proposed work, different categories of fruit images should be used for training purpose. The adoption of this new methodology help the farmers in robotic harvesting.

REFERENCES

1. Inkyu Sa, Zongyuan Ge, Feras Dayoub, Ben Upcroft, Tristan Perez, and Chris McCool, "DeepFruits: A Fruit Detection System Using Deep Neural Networks," *Sensors*, vol. 16, no. 8, pp 1-23,Aug. 2016

2. Maryam Rahnemoonfar and Clay Sheppard, "Deep Count: Fruit Counting Based on Deep Simulated Learning," *Sensors*, vol. 19, pp 1-12, April-2017.

3. A. Raihana and R. Sudha , "AFDGA: Defect Detection and Classification of Apple Fruit Images using the Modified Watershed Segmentation Method," *IJSTE - International Journal of Science Technology & Engineering*, vol. 3, no. 6,pp. 75-85, Dec. 2016.

4. Chandra Sekhar Nandi, Bipan Tudu, and Chiranjib Koley, "A Machine Vision-Based Maturity Prediction System for Sorting of Harvested Mangoes," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 7, pp. 1722-1729, July 2014.

5. Woo Chaw Seng, " A New Method for Fruits Recognition System ," *International Conference on Electrical Engineering & Informatics*, vol.1, 130-134, 2009

6. B.Kanimozhi and R.Malliga, "Classification of Ripe or Unripe Orange Fruits Using the Color Coding Technique," *Asian Journal of Applied Science and Technology*, vol. 1, no. 3, pp. 43-47, April 2017.

7. Donggi Kim, Hongchul Choi, Jaehoon Choi, Seong Joon Yoo and Dongil Han, "A Novel Red Apple Detection Algorithm Based on AdaBoost Learning," *IEIE Transactions on Smart Processing and Computing*, vol. 4, no. 4, pp. 265-272, Aug. 2015.

8. Yudong Zhang and Lenan Wu, "Classification of Fruits Using Computer Vision and a Multiclass Support Vector Machine," *Sensors*, pp. 12489-12505, 2012

9. M. Bulanon, T. Kataoka, Y.Ota, and T.Hiroma, "A Segmentation Algorithm for the Automatic Recognition of Fuji Apples at Harvest," *Biosystems Engineering*, vol. 83, no. 4, pp. 405-412, Aug. 2002.

10. Kyosuke Yamamoto, Wei Guo, Yosuke Yoshioka, and Seishi Ninomiya, "On Plant Detection of Intact Tomato Fruits Using Image Analysis and Machine Learning Methods," *Sensors*, pp 12192-12206, July 2014.

11. Shiv Ram Dubey, Pushkar Dixit, Nishant Singh, and Jay Prakash Gupta, "Infected Fruit Part Detection using K-Means Clustering Segmentation Technique, *"International Journal of Artificial Intelligence and Interactive Multimedia*, vol. 2,no. 2, pp. 65-72, July 2013.

12. V.Leemans, H. Magein, and M.-F.Destain, "Defects segmentation on Golden Delicious apples by using color machine vision," *Computers and Electronics in Agriculture,* pp. 117-130, Jan. 1998.

13. Anisha Syal, Divya Garg, and Shanu Sharma, "Apple Fruit Detection and Counting Using Computer Vision Techniques," *IEEE International Conference on Computational Intelligence and Computing Research,* 2014

14. Estrella Funes, Yosra Allouche, Gabriel Beltrán, and Antonio Jiménez, "A Review: Artificial Neural Networks as Tool for Control Food Industry Process," *Journal of Sensor Technology,* vol 5, pp. 28-43, Feb 2015

15. Mandeep Kaur and Reecha Sharma, "Quality Detection of Fruits by Using ANN Technique," *IOSR Journal of Electronics and Communication Engineering,* vol. 10, no. 4, pp. 35-41, Aug. 2015.

16. Mahendran R, Jayashree GC, and Alagusundaram K, "Application of Computer Vision Technique on Sorting and Grading of Fruits and Vegetables," *Journal of Food Processing & Technology,* pp. 1-7, 2011.

17. Zeeshan Malik, Sheikh Ziauddin, Ahmad R. Shahid, and Asad Safi, "Detection and Counting of On-Tree Citrus Fruit for Crop Yield Estimation," *International Journal of Advanced Computer Science and Applications,*vol. 7, no. 5,pp 519-523, 2015.

18. Anisha Syal, Divya Garg, Shanu Sharma, "A Survey of Computer Vision Methods for Counting Fruits and Yield Prediction," *International Journal of Computer Science Engineering,* vol. 2, pp. 346-350, Nov. 2013.

APPENDIX A

PUBLICATIONS

1. Ankita G.Vaidya, Dr. A. M. Bagade, "Real Time Identification, Counting and Sorting of on tree Fruits," International Conference on Communication, Computing, Storage & Energy, Feb-2018

2. Ankita G.Vaidya, Dr. A. M. Bagade, "Automatic Fruit Detection, Counting and Sorting using Computer Vision and Machine Learning Algorithms,"9th Post Graduate Conference of Information Technology, iPGCon - 2018

SUBMITTED

1. Ankita G.Vaidya, Dr. A.M. Bagade, "Robotic On Tree Fruit Harvesting To Increase The Yield And Fruit Quality In Fruit Farming", Journal of The Institution of Engineers (India): Series B, July -2018

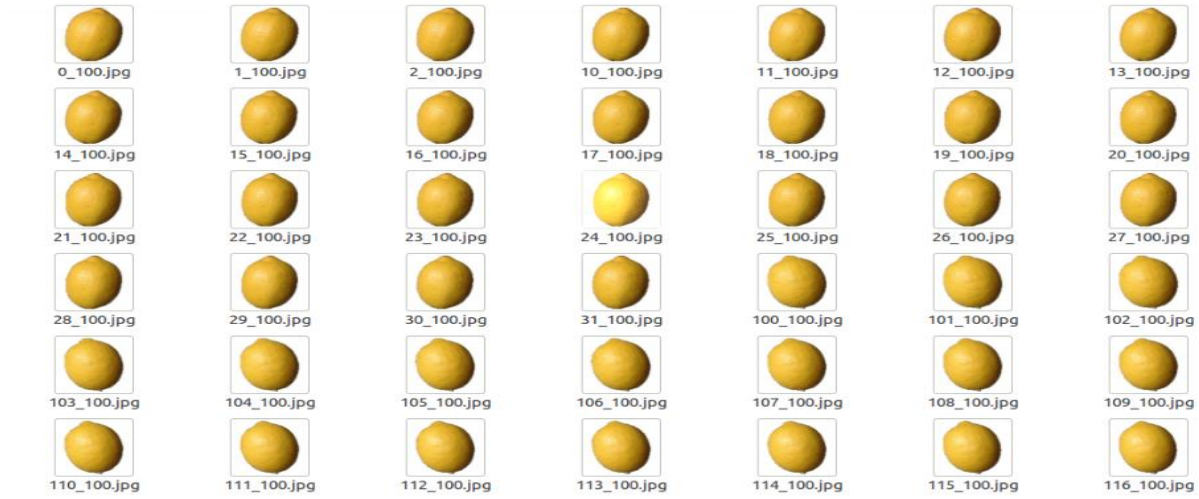
APPENDIX B

DATASET

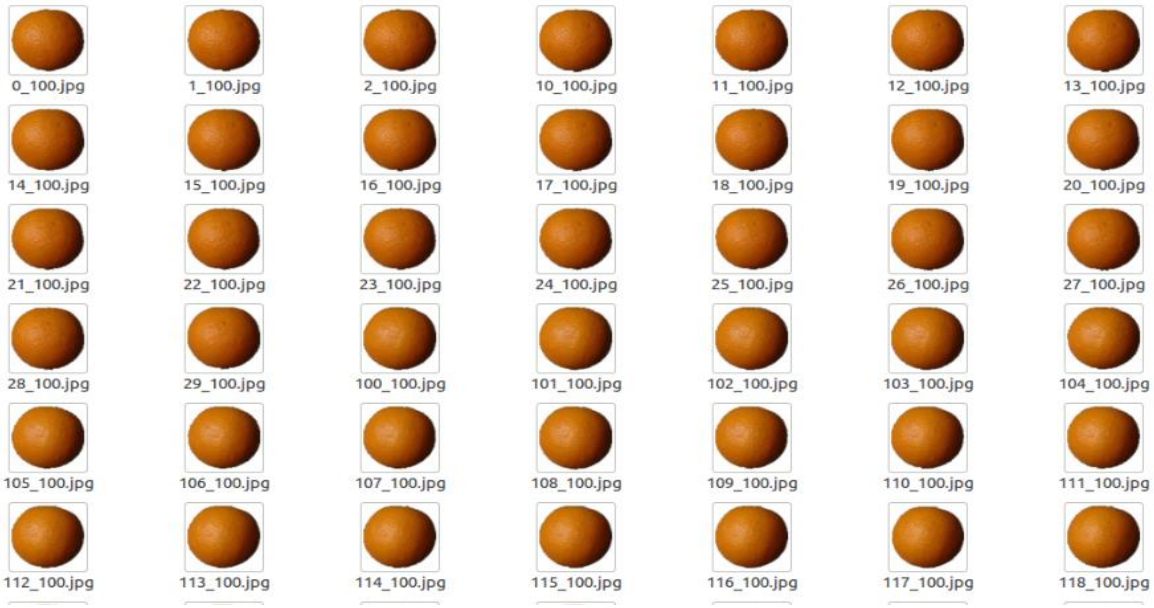
Following are the images of various fruits, which are used for traning purpose:

B.1: Images of Granny-Smith Apple

B.2 : Images of Banana



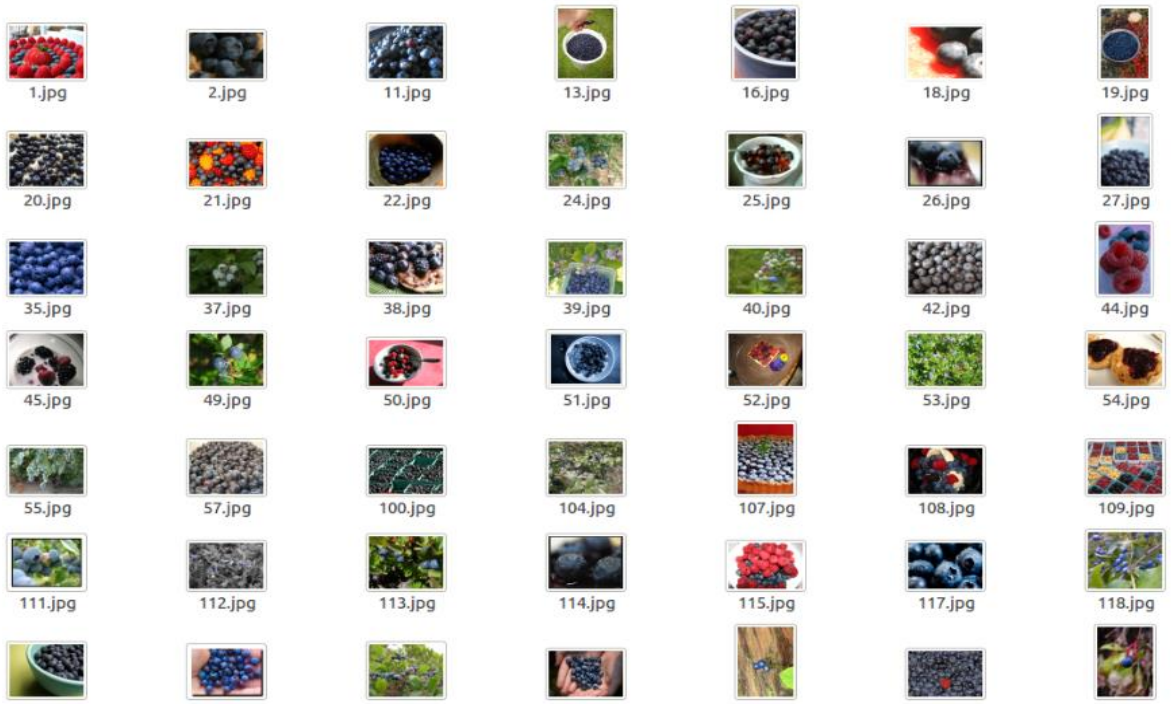
B.3 : Images of Lemon



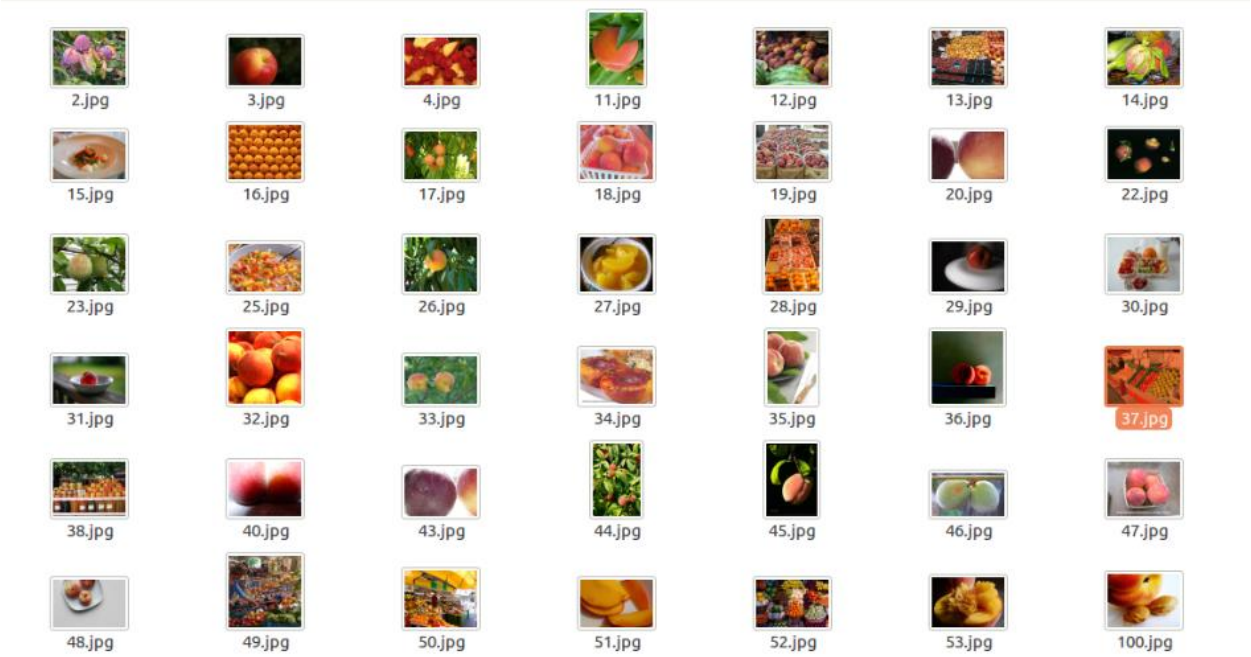
B.4 : Images of Orange

B.5 : Images of Strawbeery

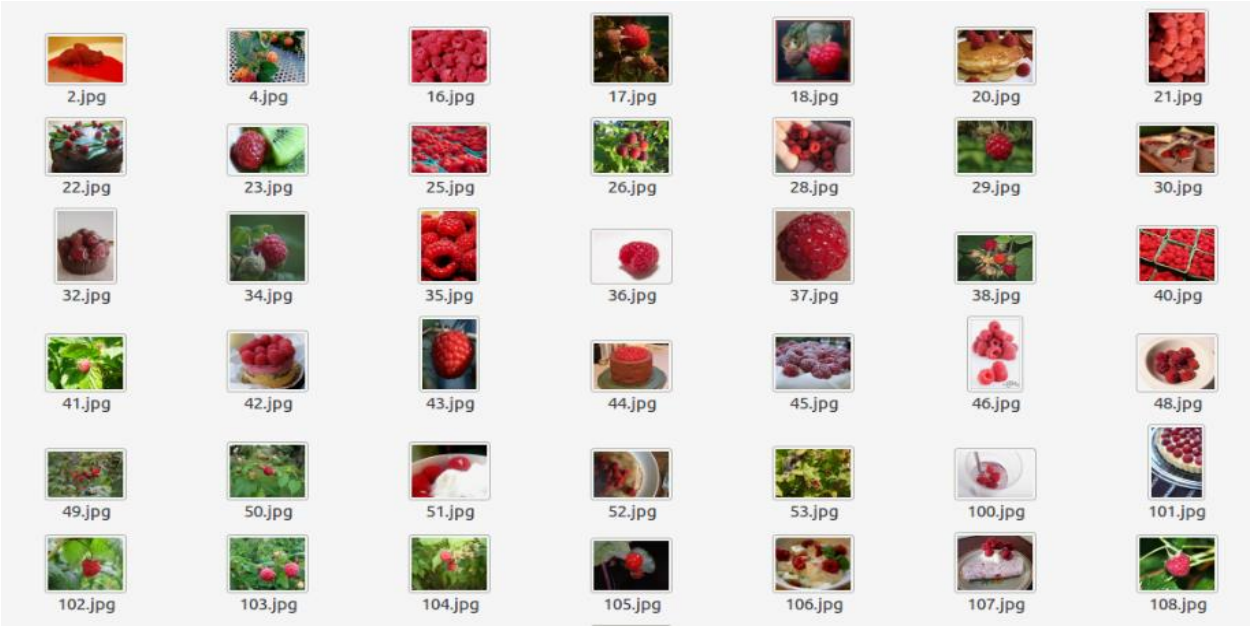
B.6 : Images of Peach



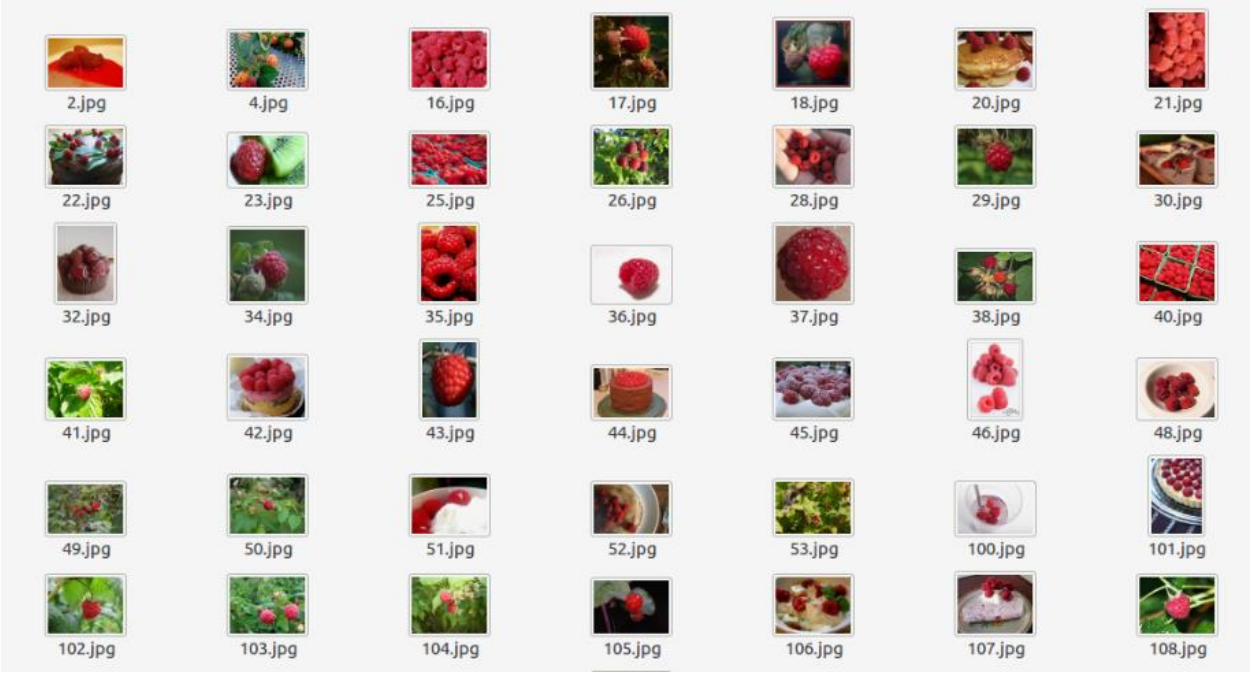
B.7: Images of Blueberry



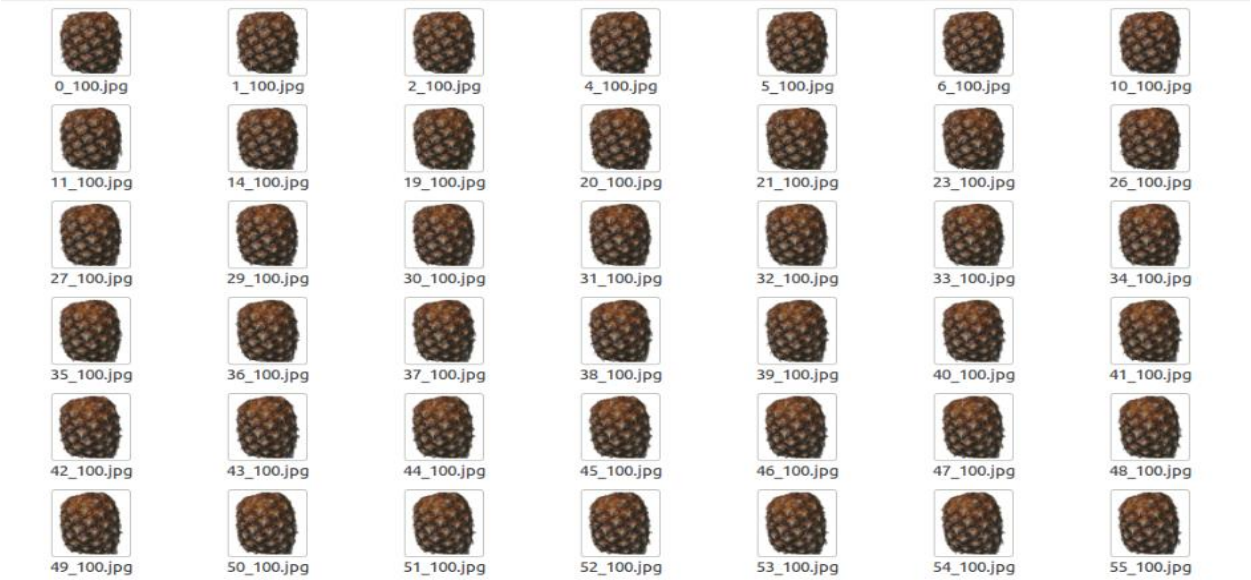
B.8: Images of Peach



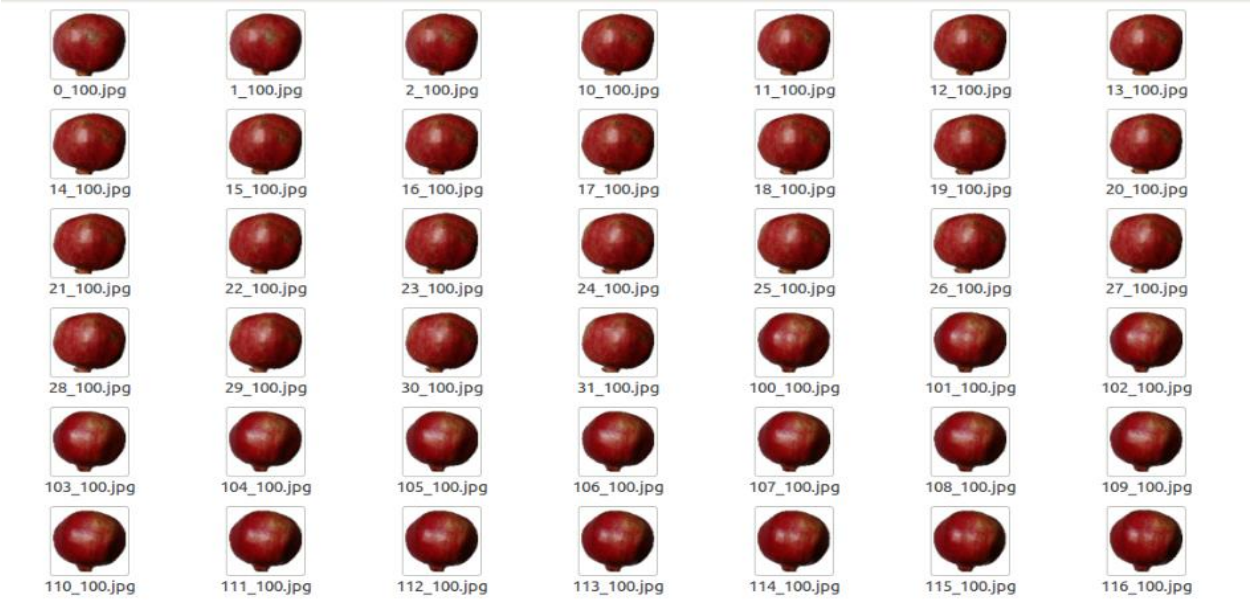
B.9: Images of Cheery



B.10: Images of Raspberry



B.11: Images of Pineapple



B.12 : Images of Pomegranate

APPENDIX C

DISSERTATION PLAN

C.1 EFFORT ESTIMATION

The project has been completed in the following phases as represented by the Life cycle and the Gantt chart.

- **Project Life Cycle**

An Iterative model has been used for implementing this project. All the phases starting from the Requirements to the testing have been enhanced throughout the project development. Fig. A.1 gives a brief pictorial representation of the phases in the project

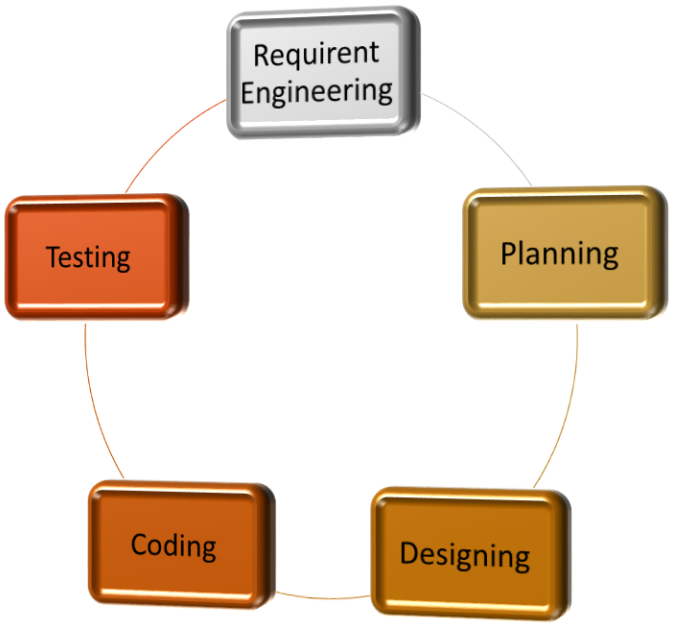


Fig C.1: Project Life Cycle

- **Project Planner**

Project planner shows detailed of the project schedule as shown in the given table.

S No	Activity	Timeline (2017-18)						
		Jul-Aug	Sep	Oct	Nov	Dec-Mar	Apr-May	Jun
1	Literature Survey							
2	Analysis and Problem formulation							
3	Design							
4	Feasibility, Scope definition							
5	Requirement Gathering							
6	System Implementation							
7	Paper Publication							
8	Results and Analysis							
9	Report Writing							

APPENDIX D

APPLICABLE UML DIAGRAMS

D.1 DESIGN OF WORK

In this chapter we will discuss different UML diagram related to the work. The Unified Modeling Language is a general-purpose modeling language in the field of software engineering, which is design to provide a standard way to visualize the designs of the system. There are different languages based on our work and datasets. these diagrams give an idea to process further in reliable way. Let us discuss different diagrams:

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

D.1.1 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted. The purpose of use case diagram is to capture the dynamic aspect of a system. But this definition is too generic to describe the purpose. Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. So when a system is analyzed to gather its functionalities use cases are prepared and actors are identified. Now when the initial task is complete use case diagrams are modelled to present the outside view.

Figure D.1: Use Case Diagram

D.1.2 DFD Diagram

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing

D.1.2.1 DFD 0 level for classification of fruit

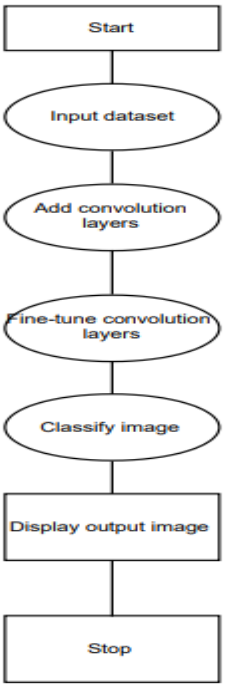


Fig D.1: DFD for classification of fruit

D.1.2.1 DFD 1 level for detect and count fruit

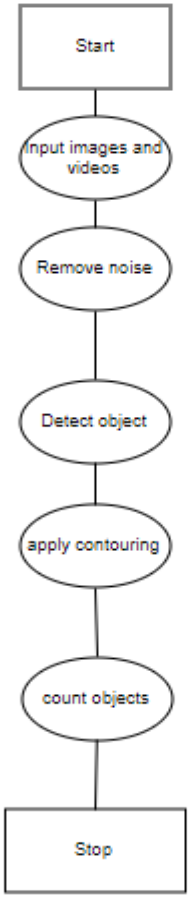


Fig D.2: DFD for detect and count fruit

D.1.2.3Activity Diagram.

A activity diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics.

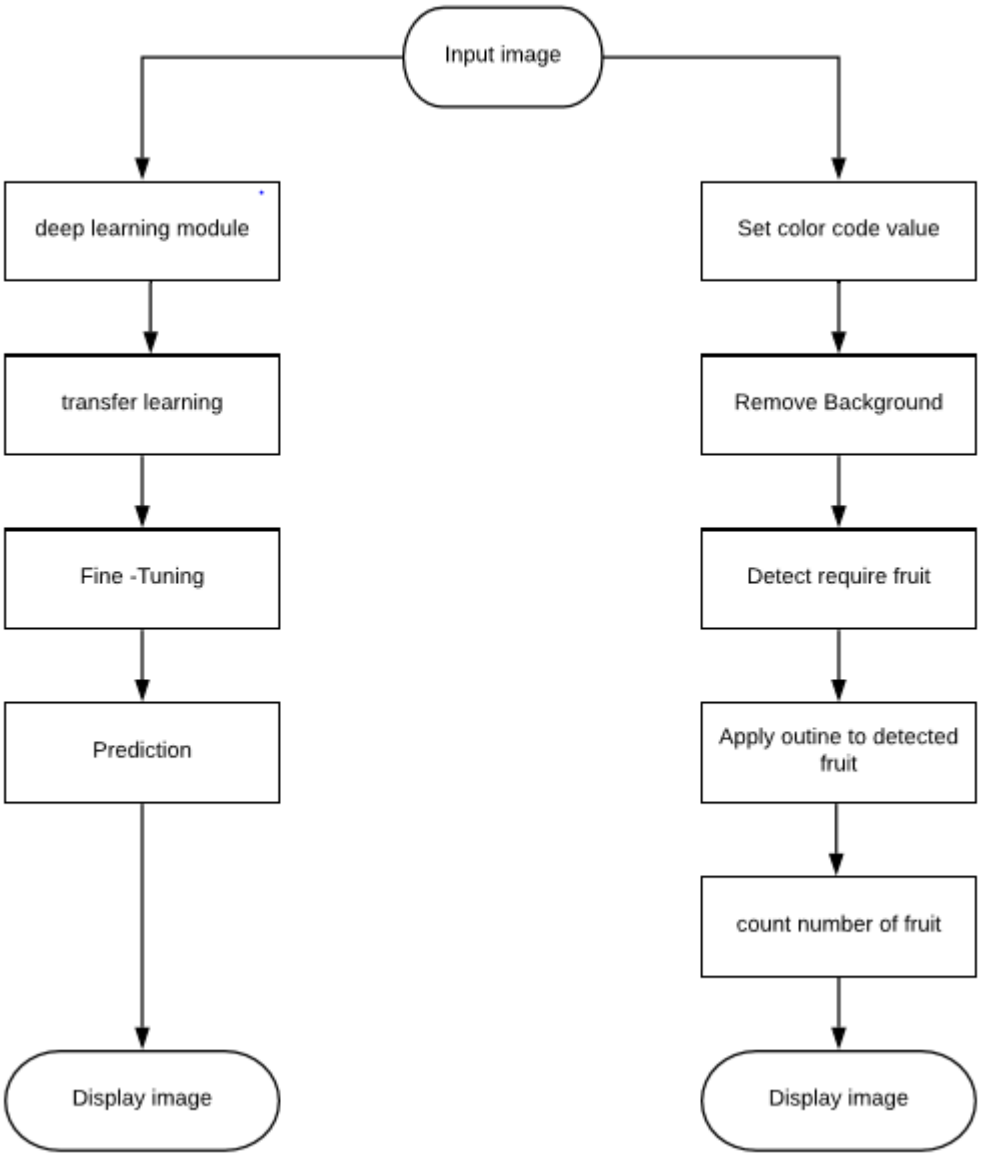


Figure D.3: Activity Diagram

APPENDIX E

FUNCTION POINT ANALYSIS

Function Point Analysis (FPA) :

Function Point Analysis is an ISO recognized method to measure the functional size of an information system. The functional size reflects the amount of functionality that is relevant to and recognized by the user in the business. It is independent of the technology used to implement the system.

All software applications will have numerous elementary processes or independent processes to move data. Transactions (or elementary processes) that bring data from outside the application domain (or application boundary) to inside that application boundary are referred to as external inputs. Transactions (or elementary processes) that take data from a resting position (normally on a file) to outside the application domain (or application boundary) are referred as either an external outputs or external inquiries. Data at rest that is maintained by the application in question is classified as internal logical files. Data at rest that is maintained by another application in question is classified as external interface files.

Function Point calculation

The function point method was originaly developed by Bij Albrecht. A function point is a rough estimate of a unit of delivered functionality of a software project. Function points (FP) measure size in terms of the amount of functionality in a system. Function points are computed by first calculating an unadjusted function point count (UFC). Counts are made for the following categories

1.Number of user inputs

Each user input that provides distinct application oriented data to the software is counted.

2. Number of user outputs

Each user output that provides application oriented information to the user is counted. In this context "output" refers to reports, screens, error messages, etc. Individual data items within a report are not counted separately.

3. Number of user inquiries

An inquiry is defined as an on-line input that results in the generation of some immediate software response in the form of an on-line output. Each distinct inquiry is counted.

4. Number of files

Each logical master file is counted.

5. Number of external interfaces

All machine-readable interfaces that are used to transmit information to another system are counted.Once this data has been collected, a complexity rating is associated with each count according to Table 1

Measurement Parameter	**Weighting Factor**		
	Simple	**Average**	**Complex**
Number of user inputs	3	4	6
Number of user outputs	4	5	7
Number of user inquiries	3	4	6
Number of files	7	10	15
Number of external interfaces	5	7	10

Table 1. Function point complexity weights

Average complexity weights= {4,5,4,10,7} for the 5 complexites respectively. Each count is multiplied by its corresponding complexity weight and the result are summed to provide the UFC . The adjusted function point count (FP) is calculated by multiplying the UFC by a technical complexity factor (TCF) also referred to as Value Adjustment Factor (VAF). Components of the TCF are listed in Table 2

1	Reliable back-up and recovery	F2	Data communications
F3	Distributed functions	F4	Performance
F5	Heavily used configuration	F6	Online data entry
F7	Operational ease	F8	Online update
F9	Complex interface	F10	Complex processing
F11	Reusability	F12	Installation ease
F13	Multiple sites	F14	Facilitate change

Table 2: Components of the technical complexity factor

Each component is rated from 0 to 5, where 0 means the component has no influence on the system and 5 means the component is essential . The VAF can then be calculated as:

VAF = 0.65 + (Sum of GSCs X 0.01) Where Sum of GSCs = SUM(Fi) (1)

The final function point calculation is:

Final Adjusted *FP = UFC x VAF* *(2)*

For this system function point is calculate as follows:

- Number of user inputs = 50
- Number of user outputs = 40
- Number of user enquiries = 35
- Number of user files = 12
- Number of external interfaces = 03

Thus by using above formula from equation (2)

FP for system is 687.81 **FP**