Chapter 1

INTRODUCTION

Identification of the plant diseases is the key to preventing the losses in the yield and quantity of the agricultural product. The studies of the plant diseases mean the studies of visually observable patterns seen on the plant. Health monitoring and disease detection on plant is very critical for sustainable agriculture. It is very difficult to monitor the plant diseases manually. It requires tremendous amount of work, expertize in the plant diseases, and also require the excessive processing time. Hence, image processing and Machine learning techniques are used for the detection of plant diseases. Disease detection involves the steps like image acquisition, image pre-processing, image segmentation, feature extraction and classification.

1.1 MACHINE LEARNING

Machine learning (ML) is the study of algorithms and mathematical models that computer systems use to progressively improve their performance on a specific task. Machine learning algorithms build a mathematical model of sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning.

Machine learning tasks are classified into several broad categories. In supervised learning, the algorithm builds a mathematical model of a set of data that contains both the inputs and the desired outputs. For example, if the task were determining whether an image contained a certain object, the training data for a supervised learning algorithm would include images with and without that object (the input), and each image would have a label (the output) designating whether it contained the object. In special cases, the input may be only partially available, or restricted to special feedback. Semi-

supervised learning algorithms develop mathematical models from incomplete training data, where portions of the sample inputs are missing the desired output.

Plant disease identification by visual way is more laborious task and at the same time, less accurate and can be done only in limited areas. Whereas if automatic detection technique is used it will take less efforts, less time and become more accurate. In plants, some general diseases seen are brown and yellow spots, early and late scorch, and others are fungal, viral and bacterial diseases. Image processing is used for measuring affected area of disease and to determine the difference in the color of the affected area.

Image segmentation is the process of separating or grouping an image into different parts. There are currently many different ways of performing image segmentation, ranging from the simple thresholding method to advanced color image segmentation methods. These parts normally correspond to something that humans can easily separate and view as individual objects. Computers have no means of intelligently recognizing objects, and so many different methods have been developed in order to segment images. The segmentation process is based on various features found in the image. This might be color information, boundaries or segment of an image. Image is segmented using the K-means clustering technique. Then unnecessary part (green area) within leaf area is removed. After that we calculate the texture features for the segmented infected object. Finally, the extracted features are passed through a pre-trained neural network.

1.2 IMAGE PROCESSING

Image processing is any form of processing for which the input is an image or a series of images or videos, such as photographs or frames of video. The output of image processing can be either an image or a set of characteristics or parameters related to the image. It also means "Analysing and manipulating images with a computer". Image processing is performed this three steps:

- First, import images with an optical devices like a scanner or a camera or directly through digital processing.
- Second, manipulate or analyse the images in some way. This step can include image improvement and data summary, or the images are analysed to find rules

that aren't seen by the human eyes. For example, meteorologists use this processing to analyse satellite photographs.

- Last, output the result of image processing. The result might be the image changed by some way or it might be a report based on analysis or result of the images.

1.3 Disease classification with Convolutional neural network

When it comes to machine learning Artificial Neural Networks perform really well. Artificial Neural Networks are used in various classification tasks like image, audio, words. Different types of Neural Networks are used for different purposes, for example for predicting the sequence of words we use Recurrent Neural Networks more precisely an LSTM, similarly for image classification we use Convolutional Neural Network. In this Blog, we are going to build basic building block for CNN.

ConvNets derive their name from the "convolution" operator. The primary purpose of Convolution in case of a ConvNet is to extract features from the input image. Convolution preserves the spatial relationship between pixels by learning image features using small filters.

There are many important parts of the Convolution Network. These includes the following properties

1. Depth: Depth corresponds to the number of filters we use for the convolution operation.

2. Stride: Stride is the number of pixels by which we slide our filter matrix over the input matrix.

3. Zero-padding: Sometimes, it is convenient to pad the input matrix with zeros around the border, so that we can apply the filter to bordering elements of our input image matrix.

4. Non-Linearity: ReLU is an element wise operation (applied per pixel) and replaces all negative pixel values in the feature map by zero. The purpose of ReLU is to introduce non-linearity in our ConvNet, since most of the real-world data we would want our ConvNet to learn would be non-linear (Convolution is a linear operation element wise

matrix multiplication and addition, so we account for non-linearity by introducing a non-linear function like ReLU).

5. Spatial Pooling: Spatial Pooling (also called subsampling or down sampling) reduces the dimensionality of each feature map but retains the most important information. Spatial Pooling can be of different types: Max, Average, Sum etc. These networks have grown in the number of layers leading to architectures such as RestNet and AlexNet that have been trained on images such as Cifar-10 and then fined tune to other problems, such as plant classification.

1.4 EXISTING SYSTEM

Leaf shape description is that the key downside in leaf identification. Up to now, several form options are extracted to explain the leaf form. However, there's no correct application to classify the leaf once capturing its image and identifying its attributes, however. In plant leaf classification leaf is classed supported its completely different morphological options.

A number of the classification techniques used are:

- Fuzzy logic
- Principal component Analysis
- k-Nearest Neighbours Classifier

1.5 PROPOSED SYSTEM

The main purpose of proposed system is to detect the diseases of plant leaves by using feature extraction methods where features such as shape, color, and texture are taken into consideration. Convolutional neural network (CNN), a machine learning technique is used in classifying the plant leaves into healthy or diseased and if it is a diseased plant leaf, CNN will give the name of that particular disease. Suggesting remedies for particular disease is made which will help in growing healthy plants and improve the productivity.

First the images of various leaves are acquired using high resolution camera so as to get the better results & efficiency. Then image processing techniques are applied to these images to extract useful features which will be required for further analysis. The basic steps of the system are summarized as:

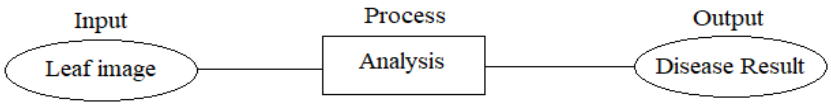

Fig 1.1 Proposed System.

The advantages of proposed algorithm are as follows:

- Use of estimators for automatic Initialization of cluster centres so there is no need of user input at the time of segmentation.
- The detection accuracy is enhanced with proposed algorithm.
- Proposed method is fully automatic while existing methods require user input to select the best segmentation of input image.
- It also provides environment friendly recovery measures of the identified disease.

1.6 OBJECTIVES

- To enhance the given input image by Image acquisition and Image pre-processing.
- Identify the affected part through texture analysis and Segmentation.
- Classify the healthy and affected leaf part by feature extraction and classification.
- Train the model by using testing data for accurate result.

1.7 PURPOSE

India is an agricultural country, where most of the population depends on agricultural products. So the cultivation can be improved by technological support. Diseases may cause by pathogen in plant at any environmental condition. In most of the cases diseases

are seen on the leaves of the plants, so the detection of disease plays an important role in successful cultivation of crops.

There are lots of techniques to detect the different types of diseases in plants in its early stages. Conventional methods of plant disease detection in naked eye observation methods and it is non-effective for large crops. Using digital image processing and machine learning the disease detection in plant is efficient, less time consuming and accurate. This technique saves time, efforts, labours and use of pesticides. Hope this approach will becomes a little contribution for agriculture fields.

Chapter 2

LITERATURE SURVEY

2.1 Yashpal Sen, Chandra Shekar Mithlesh, Dr. Vivek Baghel

[1] describes an approach for disease detection of crop for economic growth of rural area. This paper discussed about an automated system for identifying and classifying different diseases of the contaminated plants is an emerging research area in precision agriculture. This paper describes the approach to prevent the crop from heavy loss by careful detection of diseases. The region of internet is leaf because most of the diseases occur in leaf only. Histogram equalization is used to pre-process the input image to increase the contrast in low contrast image, K-mean clustering algorithm which classifies objects. Disease in crop leaf are detected accurately using image processing technique it is used to analyse the disease which will be useful to farmers.

2.2 K. Elangoran, S. Nalini [2]

presented a concept of plant disease classification using image segmentation and SVM techniques. This paper describes an image processing technique that identifies the visual symptoms of plant diseases using an analysis of colored images, work of software program that recognizes the color and shape of the leaf image. LABVIEW software was used to capture the image of plant RGB color model and MATLAB software is used to enable a recognition process to determine the plant disease through the leaf images. The color model respectively was used to reduce effect of illumination and distinguish between leaf colors efficiently and the resulting color pixels are clustered to obtain groups of color in the images.

2.3 Sandesh Raut, Karthik Ingale [3]

proposed fast and accurate method for detection and classification of plant diseases. The proposed algorithm is tested on main five diseases on the plant they are Early Scorch, Cottony mold, Ashen Mold, Late scorch, Tiny Whiteness. Initially the RGB image is acquired then a color transformation structure for the acquired RGB leaf image is created. After that color value in RGB converted to the space specified in the color transformation structure. In the next step,

the segmentation is done by using K-means clustering technique after that mostly green pixels are masked. Finally, the feature extracted was recognized through a pre-trained neural network. The result show that the proposed system can successfully detect and classify the diseases with a precision between 83% and 94%.

2.4 Sagar Patil, Anjali chandavale [4] this survey mainly concentrates on disease detection of dicot plants, here the image acquisition is done by taking RGB image pattern as input and transform it into HSI form, after that for texture analysis CCM and SGDM is used. In agricultural field, rice cultivation plays a vital role. But their growths are affected by various diseases. There will be decrease in the production if the diseases are not identified at an early stage. The main goal of this work is to develop an image processing system that can identify and classify the various rice plant diseases affecting the cultivation of rice namely brown spot disease, leaf blast diseases and bacterial blight disease. This work can be divided into 2 parts namely-rice plant disease detection and recognition of rice plant diseases. In disease detection, the disease affected portion of the rice plant is first identified using KNN and clustering classifier. After that, in disease recognition the rice plant disease type is recognized using classifiers namely KNN and SVM.

2.5 T. RUMPf, A-K , Mahlein, U.Steiner , E-C Oerke[5] Work expertise in plant diseases requires successive processing time hence image processing is used for the detection of plant diseases, this paper discuss the method used for the detection of plant diseases using the leaves images, and also various techniques to segment the disease part of the plant this paper also discussed some feature extraction and classification techniques to extract the features of infected leaf and the classification of plant diseases the accurately detection and classification of the plant diseases is very important for the successful cultivation of Crop and this can be done using image processing the use of an n methods for classification of disease in plant such as self-organizing feature map block back propagation algorithm SVM excreta can be efficiently used from these methods we can accurately identify and classify various plant diseases using image processing techniques machine learning methods such as

artificial neural network decision tree Cayman neighbour's and support vector machine have been applied in Agricultural Research this paper also discussed dichotomous classification between healthy leaves and leaves with disease symptoms the results showed that the specificity of the classification was always lower than the sensitivity the classification error range was 7% to almost 3% the classification accuracy increased with increasing disease severity the result showed that the classification accuracy was about to 65% for all diseases.

2.6 Mr. Sanjay Mirchandani, Mihir pendse, Prathamesh Rane, Ashwini Vedula[6] researchers proposed detection and classification of plant disease using image processing and artificial neural networks in this paper a software solution for fast accurate and automatic detection and classification of plant disease through image processing identification of disease is key to preventing losses in the quality and quantity of the agricultural product this paper discussed the detection and classification of plant diseases is divided into three steps such as identification of infected object extraction of feature set of the infected leaf images and detection and classification the type of disease using ANN. Different techniques are adopted for detection and diagnosis the disease but the better way is by image processing the author suggested a method in which initially the infected region is found then different features are extracted such as colour texture and shape finally parameter classification technique is used for detecting the diseases.

2.7 Savita N Ghaiwat, Parul Arora[7] presented an image processing technique for detection and classification of plant disease classification technique deals with classifying each pattern in one of the distinct classes the author suggested so many techniques for classification such as K nearest neighbour classifier probabilistic neural network genetic algorithm support vector machine and principal component analysis artificial neural network Fuzzy Logic the technique is presented using image processing as a tool to enhance the feature extraction of an image by using of local binary pattern as a parameter the local binary pattern approach has evolved to represent a significant breakthrough in texture analysis outperforming earlier method in many application

study of image analysis takes which have not been generally considered texture analysis problems is done results suggest that texture analysis and the ideas behind the lbp method could have a much wider role in image analysis and computer vision then was thought before.

2.8 Amar Kumar dey [8] This paper deals with leaf rot disease detection for betel vine based on image processing. The proposed methodology has three vital stages the initial stage was the image acquisition stage through which the real world sample is recorded in the digital form using flatbed digital scanner next stage is image processing segmentation classification and leaf area calculation. Here 21*30 sq. cm image is Canon under flatbed scanner during test phase it acquired a series of 12 color images using scanner and then the color images were digitalized at a resolution of 300 DPI to produce RGB digital color images. The digital version of the leaf sample consists of a about 30% of leaf area and rest 70% is background in order to achieve fast processing digital image of leaf cropped into a smaller dimension of size after this is a state the digital version of the sample leaf image consist about 70% of leaf area part and rest 30% as background color feature are used to identify the affected area. RGB, hsv, ycbcr color models are used to color identification these three channels result in 12 individual image then the queue is responsible for masking and approximated threshold value is applied to affected leaf was calculated.

2.9 Jayamala k Patil, Rajkumar [9] To retrieve the related images the search is done in Two Steps, first step is matches the images by comparing the standard deviations for three colour components the second step is weighted version of the Euclidean distance between the feature coefficients of an image selected in the first step, this reported following important image processing method first one image clipping separating the leaf with spots from the complex background. Noise resolution to filters simple filter and median filter work compared and at last median filter was chosen. Thresholding to segment or partition image into the spot background fourth one segmentation they used OSTU's method. K-means clustering and back propagation feed-forward neural network for clustering and classification. There are two main characteristics of plant disease detection using machine learning method that must be achieved their speed and accuracy hence innovative efficient and fast interpreting

algorithms which will help plants scientists in detecting disease work proposed by the researcher can be extended for development of hybrid algorithms such as genetic algorithms and neural networks in order to increase the recognition rate of the final classification process.

2.10 S Arivazhagan, R Newlin shebiah, S Ananthi, S Vishnu varthini [10] the approaches and methodologies which are used in this survey includes RGB acquisition: input image and consider the image color according to the RGB model, Color transformation structure: includes the transformation of colors from RGB to hsv (Hue saturation intensity) h component taken into analysis. Masking green pixels: It identifies the mostly green colored pixels based on threshold value. Segmentation: Infected portion of the leaf is extracted and divide them into patches, and extract the useful segments. CCM texture analysis is developed through sgdm. Classifier: as a classifier minimum distance criteria is used and SVM are used for better classification and regression. Hear the application of texture analysis is highlighted by this technique only some group of plant disease can be detected and SVM has introduced in less computation method.

Chapter 3

SYSTEM REQUIREMENTS SPECIFICATION

3.1 Hardware Requirements

- Processor : 2.5 gigahertz (GHz) frequency or above.
- RAM : A minimum of 4 GB of RAM.
- Hard disk : A minimum of 20 GB of available space.
- Input Device : High resolution camera
- Monitor : Minimum Resolution 1024 X 768.

3.2 Software Requirements

- Operating System : Windows 7 and above.
- Programing language : Python 2.7 and above.
- Platform : JetBrains PyCharm Community Edition 2018.3.5 x64
- Supporting libraries : Tensorflow, OpenCV, PIL, tkinter, os, SKlearn etc.

3.3 Functional requirements

Functional requirements describe the system functionality, while the non-functional requirements describe system properties and constraints.

Functional requirements capture the intended behaviour of the system. This behaviour may be expressed as services, tasks, or the functions the system is required to perform. This lays out important concepts and discusses capturing functional requirements in such a way they can drive architectural decisions and be used to validate the architecture. Features may be additional functionality, or differ from basic functionality along some quality attribute. In the proposed system, concert assesses the compliance of a workflow by analysing the five established elements required to check for the rule adherence in workflows: activities, data, location, resources, and time limits. A rule describes which activities may, must or must not be performed on what objects

by which roles. In addition, a rule can further prescribe the order of activities i.e. which activities have to happen before or after other activities.

3.4 Non-functional requirements:

1. Security

- System needs to control the user access and session
- It needs to store the data in a secure location and stored in a secure format
- It requires a secure communication channel for the data.

2. Concurrency and Capacity

System should be able to handle multiple computations executing simultaneously, and potentially interacting with each other.

3. Performance

Performance is generally perceived as a time expectation. This is one of the most important considerations especially when the project is in the architecture phase.

4. Reliability

It is necessary to ensure and notify about the system transactions and processing as simple as keep a system log will increase the time and effort to get it done from the very beginning. Data should be transferred in a reliable way and using trustful protocols.

5. Maintainability

Well-done system is meant to be up and running for long time. Therefore, it will regularly need preventive and corrective maintenance. Maintenance might signify scalability to grow and improve the system features and functionalities.

6. Usability

End user satisfaction and acceptance is one of the key pillars that support a project success. Considering the user experience requirements from the project

conception is a win bet, and it will especially save a lot of time at the project release, as the user will not ask for changes or even worst misunderstandings.

7. Documentation

All projects require a minimum of documentation at different levels. In many cases the users might even need training on it, so keeping good documentation practices and standards will do this task spread along the project development; but as well this must be establish since the project planning to include this task in the list.

Chapter 4

SYSTEM DESIGN

4.1 Data flow Diagram

A data flow Diagram 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.

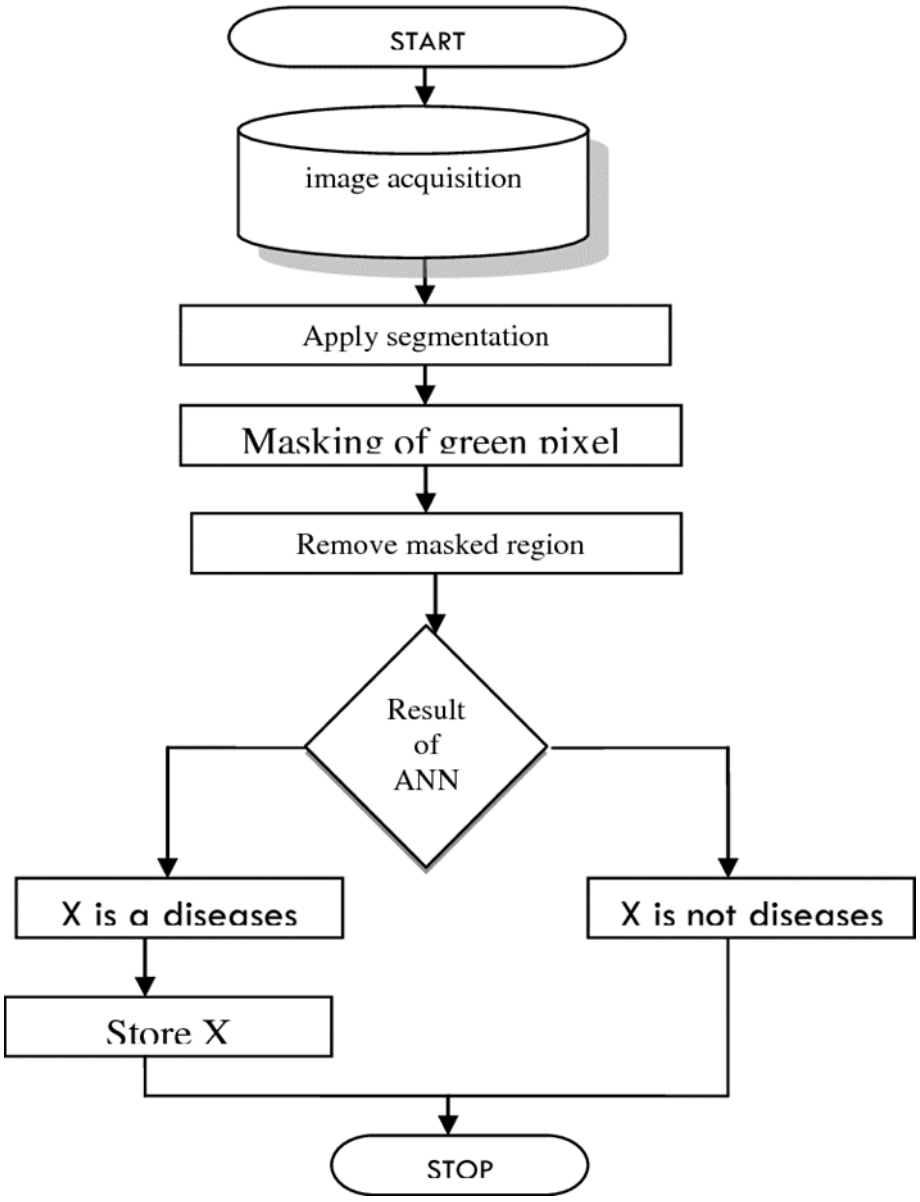

Fig.4.1 Flowchart for leaf classification

4.2 Use case Diagram

Use case diagram is a graphic depiction of the interactions among the elements of a system. Use cases will specify the expected behaviour, and the exact method of making it happen. Use cases once specified can be denoted both textual and visual representation.

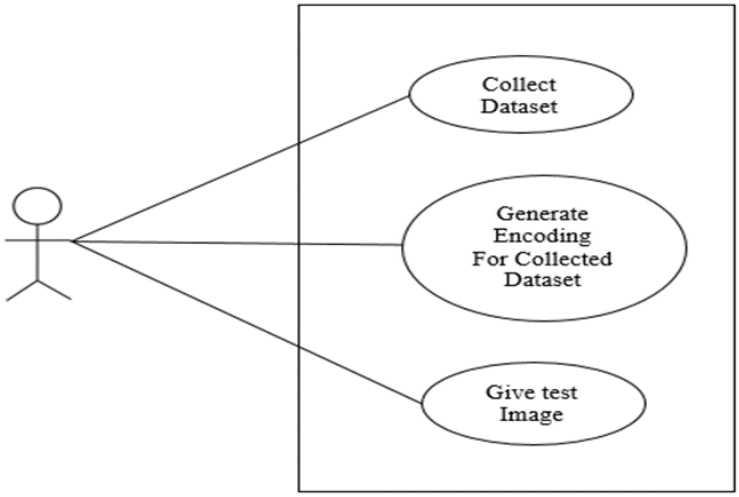

Fig.4.2 Use case diagram

Use case diagrams are used to specify:

- Requirements (external), required usages of a system under design or analysis - to capture what the system is supposed to do.
- The functionality offered by a subject – what the system can do.
- Requirements the specified subject poses on its environment - by defining how environment should interact with the subject so that it will be able to perform its services.

4.3 Sequence Diagram

Sequence diagrams are sometimes called event diagrams or event scenarios. A sequence diagram shows as parallel vertical lines (lifelines), different processes or objects that live simultaneously and as horizontal arrows, the messages exchanged between them in the order in which they occur.

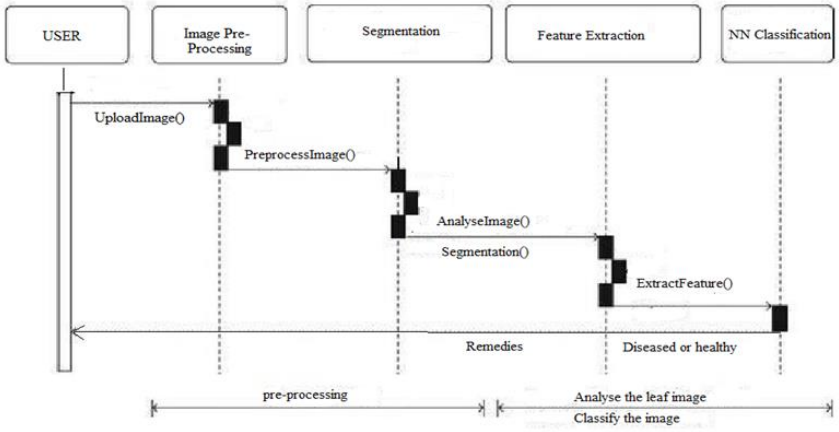

Fig.4.3 Sequence Diagram

1. Usage scenarios: A usage scenario is a description of a potential way your system is used. The logic of a usage scenario may be part of a use case, perhaps an alternate course. It may also be one entire pass through a use case, such as the logic described by the basic course of action or a portion of the basic course of action, plus one or more alternate scenarios. The logic of a usage scenario may also be a pass through the logic contained in several use cases.

2. The logic of methods: Sequence diagrams can be used to explore the logic of a complex operation, function, or procedure. One way to think of sequence diagrams, particularly highly detailed diagrams, is as visual object code

3. The logic of services: A service is effectively a high-level method, often one that can be invoked by a wide variety of clients. This includes web-services as well as business transactions implemented by a variety of technologies

4.4 Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as in operation of the system. The control flow is drawn from one operation to another.

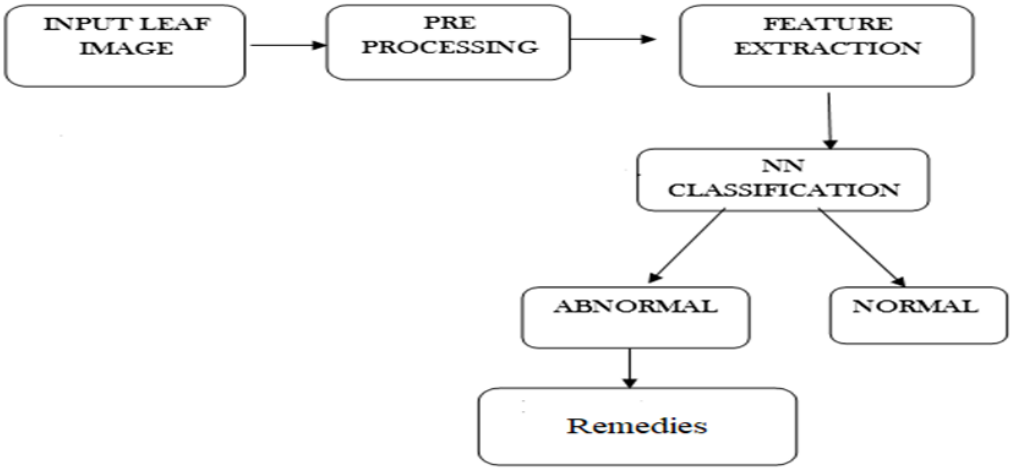

Fig.4.4 Activity Diagram

The above flow represents the flow from one activity to another activity, the activity starts from input leaf image through digital camera, and then input leaf is pre-processed and extract the features like color, shape, texture and so on. Now, the processed image is classified as Normal or Abnormal, if Abnormal is found in the leaf, then remedies will be suggested.

Chapter 5

SYSTEM IMPLEMENTATION

This part of the report illustrates the approach employed to classify the leaves into diseased or healthy and if the leaf is diseased, name of the disease is mentioned along with the remedies. Our methodology primarily revolves around the following five steps.

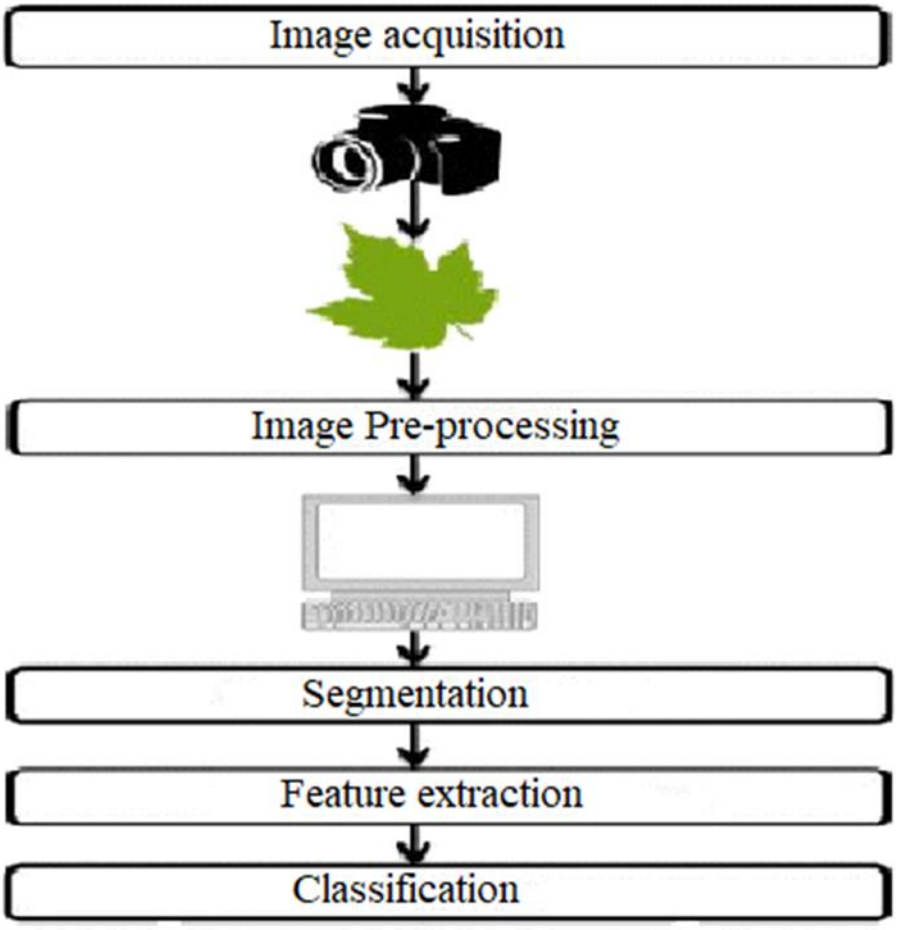

Fig.5.1 System architecture

Algorithm written below illustrated the step by step approach for the proposed image recognition and segmentation processes:

(1) Image acquisition is the very first step that requires capturing an image with the help of a digital camera.

(2) Pre-processing of input image to improve the quality of image and to remove the undesired distortion from the image. Clipping of the leaf image is performed to get the interested image region and then image smoothing is done using the smoothing filter. To increase the contrast Image enhancement is also done.

(3) Mostly green colored pixels, in this step, are masked. In this, we computed a threshold value that is used for these pixels. Then in the following way mostly green pixels are masked: if pixel intensity of the green component is less than the pre-computed threshold value, then zero value is assigned to the red, green and blue components of the this pixel.

(4) In the infected clusters, inside the boundaries, remove the masked cells.

(5) Obtain the useful segments to classify the leaf diseases.

5.1 Image Acquisition

The initial process is to collect the data from the public repository. It takes the image as input for further processing. We have taken most popular image domains so that we can take any formats like .bmp, .jpg, .gif as input to our process.

The image is captured, scanned and converted into a manageable entity. This process is known as image acquisition. During a test-phase, we acquire a series of color images using a digital scanner so as to acquire a single image of leaf. The color images were digitized to produce RGB digital color images.

5.2 Image Pre-processing

As the images are acquired from the real field it may contain dust, spores and water spots as noise. The purpose of data pre-processing is to eliminate the noise in the image, so as to adjust the pixel values. It enhances the quality of the image.

The main aim of Pre-processing is to suppress unwanted image data and to enhance some important image features. It includes RGB to Gray conversion, image resizing and median filtering. Here color image is converted to gray scale image to make the image device independent. The image is then resized to a size of 256*256. Then median filtering is performed on the image to remove the noise. The digital version of the rotten leaf sample consists of about 30% of leaf area and rest 70% is the background. Thus the redundant background requires high disk storage space and utilizes CPU time in the segmentation process. In order to have efficient disk storage and achieve fast processing speed the digital image of the leaf sample is cropped into a smaller dimension of size 16x20sq, cm. Thus the pre-processing step introduced saves about 30% of disk storage space and increases the CPU processing 1.4 times. The cropping process does not

introduce any loss in the region of interest, i.e. the selected leaf sample. After pre-processing stage, the digital version of the sample leaf image consists about 70% of leaf area part and rest 30% as background. For getting better results in further steps, image pre-processing is required because dust, dewdrops, insect's excrements may be present on the plant; these things are considered as image noise. Furthermore, captured images may have distortion of some water drops and shadow effect, which could create problems in the segmentation and feature extraction stages. Effect of such distortion can be weakened or removed using different noise removal filters. There may be low contrast in captured images; for such images contrast enhancement algorithms can be used. Sometimes background removal techniques may also be needed in case of region of interest needs to be extracted. In case of noise such as salt and pepper, a median filter can be used. Weiner filter can be used to remove a blurring effect. In case of the images captured using high definition cameras, the size of the pictures might be very large, for that reduction of image size is required. Also, image reduction helps in reducing the computing memory power.

5.3 Segmentation

Image segmentation is the third step in our proposed method. The segmented images are clustered into different sectors using Otsu classifier and k-mean clustering algorithm. Before clustering the images, the RGB color model is transformed into Lab color model. The advent of Lab color model is to easily cluster the segmented images.

5.3.1 K-means Clustering Algorithm

a) Load the input images.

b) Commute the RGB image into L*a*b color space.

c) RGB images are combination of primary colors (Red, Green, and Blue).

d) RGB image feature Pixel Counting technique is extensively applied to agricultural science.

e) The L*a*b* space consists of a radiance layer 'L*', chromaticity-layer 'a*' indicating where color falls along the red-green axis and chromaticity layer 'b*' indicating where

the color falls along the blue yellow axis. All of the color information is in the 'a*' and 'b*' layers.

f) Clustering the variant colors using k-mean method.

g) The Euclidean distance between two objects is defined as follows:

$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

……Eq. (1)

f) Each pixel is labelled under clusters based on its estimated variant cluster-centres.

5.3.2 Otsu's classifier

In image processing technique, Otsu's strategy is utilized to perform clustering based image Threshold. The diminishment of a gray level image to a binary image is done by Nobuyuki Otsu .This algorithm assumes , image contains two classes of pixels .It incorporates bi-modal histogram (foreground pixels and background pixels).We can calculate the optimum threshold by isolating the two classes and their combined spread (intra-class variance) is negligible or equivalently.

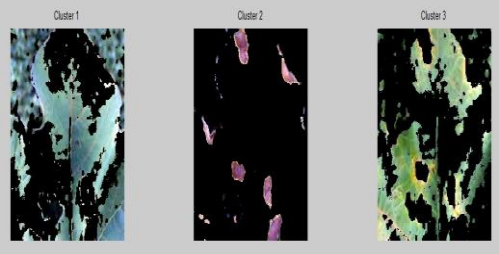

Fig.5.2 Image clustering

5.4 Feature extraction

In the proposed approach, the method adopted for extracting the feature set is called the Color Co-occurrence Method or CCM method in short. It is a method, in which both

the color and texture of an image are taken into account, to arrive at unique features, which represent that image.

5.4.1 Co-occurrence Methodology for Texture Analysis

The image analysis technique selected for this project was the CCM method. The use of color image features in the visible light spectrum provides additional image characteristic features over the traditional gray-scale representation.

The CCM methodology established in this work consists of three major mathematical processes. First, the RGB images of leaves are converted into Hue Saturation Intensity (HSI) color space representation. Once this process is completed, each pixel map is used to generate a color co-occurrence matrix, resulting in three CCM matrices, one for each of the H, S and I pixel maps. (HSI) space is also a popular color space because it is based on human color perception. Electromagnetic radiation in the range of wavelengths of about 400 to 700 nanometres is called visible light because the human visual system is sensitive to this range. Hue is generally related to the wavelength of a light and intensity shows the amplitude of a light. Lastly, saturation is a component that measures the "colorfulness" in HSI space.

Color spaces can be transformed from one space to another easily. In our experiments, the Equations 1, 2 and 3 were used.

$$Hue\ (H) = \begin{cases} 2 - ACOS\left\{\frac{[(R-G)+(R-B)]}{\sqrt[2]{(R-G)^2+(R-G)(G-B)}}\right\} & ,B > G \\ ACOS\left\{\frac{[(R-G)+(R-B)]}{\sqrt[2]{(R-G)^2+(R-G)(G-B)}}\right\} & ,B \leq G \end{cases} \quad \text{.....Eq. 1}$$

$$Saturation\ (S) = 1 - \frac{3.\min\ (R,G,B)}{(R+G+B)} \quad \text{.....Eq. 2}$$

$$Intensity\ (I) = \frac{R+G+B}{3} \quad \text{.....Eq. 3}$$

The color co-occurrence texture analysis method was developed through the use of Spatial Gray-level Dependence Matrices (SGDM"s). The gray level co-occurrence methodology is a statistical way to describe shape by statistically sampling the way certain grey-levels occur in relation to other grey-levels.

These matrices measure the probability that a pixel at one particular gray level will occur at a distinct distance and orientation from any pixel given that pixel has a second particular gray level. For a position operator p, we can define a matrix Pij that counts

the number of times a pixel with greylevel i occurs at position p from a pixel with grey-level j. The SGDMs are represented by the function P(i, j, d, Θ) where i represents the gray level of the location (x, y) in the image I(x, y), and j represents the gray level of the pixel at a distance d from location (x, y) at an orientation angle of Θ. The reference pixel at image position (x, y) is shown as an asterix. All the neighbors from 1 to 8 are numbered in a clockwise direction. Neighbors 1 and 5 are located on the same plane at a distance of 1 and an orientation of 0 degrees. An example image matrix and its SGDM are already given in the three equations above. In this research, a one pixel offset distance and a zero degree orientation angle was used.

After the transformation processes, we calculated the feature set for H and S, we dropped (I) since it does not give extra information. However, we use HSI function in Python to create gray-level co-occurrence matrix; the number of gray levels is set to 8, and the symmetric value is set to "true", and finally, offset is given a" 0" value.

Normalizing the CCM matrices:

The CCM matrices are then normalized using Equation 4.

$$p(i,j) = \frac{p(i,j,1,0)}{\sum_{i=0}^{N_g-1}\sum_{j=0}^{N_g-1} p(i,j,1,0)} \quad \text{.....Eq. 4}$$

P(i , j) is the image attribute matrix, *p(i, j, 1, 0)* represents the intensity co-occurrence matrix, and N_g represents the total number of intensity levels.

The marginal probability matrix (P_x) can be defined as shown in equation 5.

$$p_x(i) = \sum_{j=0}^{N_g-1} p(i,j) \quad \text{.....Eq. 5}$$

Sum and difference matrices (P_{x+y} , P_{x-y}) are defined as shown in equation 6 and 7 respectively.

$$p_{x+y}(k) = \sum_{i=0}^{N_g-1}\sum_{j=0}^{N_g-1} p(i,j) \quad \text{.....Eq. 6}$$

$$\text{Where } k = I + j; \text{ for } k = 0, 1, 2, \ldots 2(N_g - 1), \quad \text{.....Eq. 7}$$

Morphology: Erosion concept is applied for acquiring boundaries of all database images.

$$\text{Erosion} = \{Z/(Y)z \subseteq X\} \text{ ……..…….Eq. 8}$$

$$\text{Image Boundary} = \text{Original image} - \text{Eroded image……..Eq. 9}$$

Where,

X is erosion which indicates database images and Y as input image which is set of each points Z such that Y converted by Z and contained in X in morphology. Entire knowledge about structure is symbolized with the help discrete cosine conversion considering few co-efficient.

Texture: Visual patterns describe texture property, each having similarity. Texture identification is done by modeling textures as two-dimensional deviation of gray level.

$$\psi^{ab}(x) = |a|^{(-1/2)}\, \psi\, (x - \frac{b}{a}) \text{ ……..…...Eq. 10}$$

Then

$$W\, \psi = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(t)\psi \text{ ……..……Eq. 11}$$

Where,

$\psi(x) =$ mother wavelet
$\psi^{ab}(x) =$ daughter wavelet

5.5 Classification

Convolutional neural networks are used in the automatic detection of leaves diseases. CNN is chosen as a classification tool due to its well-known technique as a successful classifier for many real applications.

CNN architectures vary with the type of the problem at hand. The proposed model consists of three convolutional layers each followed by a max-pooling layer. The final layer is fully connected MLP. ReLu activation function is applied to the output of every convolutional layer and fully connected layer.

The first convolutional layer filters the input image with 32 kernels of size 3x3. After max-pooling is applied, the output is given as an input for the second convolutional layer with 64 kernels of size 4x4. The last convolutional layer has 128 kernels of size

1x1 followed by a fully connected layer of 512 neurons. The output of this layer is given to Softmax function which produces a probability distribution of the four output classes.

Gray Level Co-occurrence Matrix is created from gray scale images and used to describe the shape feature. The Gray Level Co-occurrence Matrix is based on the repeated occurrence of gray-level configuration in the texture. The spatial gray dependence matrix is used for texture analysis. A spatial gray dependence matrix is created based on hue, saturation and intensity. Run Length Matrix (RLM) is another type of matrix. Same gray pixel values are the part of run and those gray values forms a two dimensional matrix.

Example, RM- Q(x, y) x represents gray values and y represents run length.

5.5.1 Following are some common symptoms of bacterial and viral plant leaf diseases.

- **Bacterial disease symptoms**

Bacterial diseases include any type of illness caused by bacteria. Bacteria are a type of microorganism, which are tiny forms of life that can only be seen with a microscope. Other types of microorganisms include viruses, some fungi, and some parasites. Millions of bacteria normally live on the skin, in the intestines, and on the genitalia. The vast majority of bacteria do not cause disease, and many bacteria are actually helpful and even necessary for good health. These bacteria are sometimes referred to as "good bacteria" or "healthy bacteria."

- **Viral disease symptoms**

Symptoms of viral diseases vary depending on the specific type of virus causing infection, the area of the body that is infected, the age and health history of the patient, and other factors. The symptoms of viral diseases can affect almost any area of the body or body system. Symptoms of viral diseases can include:

- Flu-like symptoms (fatigue, fever, sore throat, headache, cough, aches and pains)
- Flu-like symptoms (fatigue, fever, sore throat, headache, cough, aches and pains)

- Gastrointestinal disturbances, such as diarrhoea, nausea and vomiting
- Irritability
- Malaise (general ill feeling)
- Rash

5.5.2 Convolutional Neural Network (ConvNet) Algorithm

1) A convolutional neural network is a class of deep neural network, most commonly applied to analysing visual imagery.

2) A Convolutional Neural Network is a Machine Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

INPUT LAYER

- In the figure below, we have an RGB image which has been separated by its three color planes—Red, Green, and Blue. There are a number of such color spaces in which images exist—Grayscale, RGB, HSV, CMYK, etc.
- The role of the ConvNet is to reduce the images into a form which is easier to process, without losing features which are critical for getting a good prediction.
- This is important when we are to design an architecture which is not only good at learning features but also is scalable to massive datasets.

CONVOLUTION LAYER--The Kernel

- In the below demonstration, the green section resembles our 5x5x1 input image. The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the Kernel/Filter, K, represented in the color yellow. We have selected K as a 3x3x1 matrix{{1,0,1},{0,1,0},{1,0,1}}.
- The Kernel shifts 9 times because of Stride Length = 1, every time performing a matrix multiplication operation between K and the portion P of the image

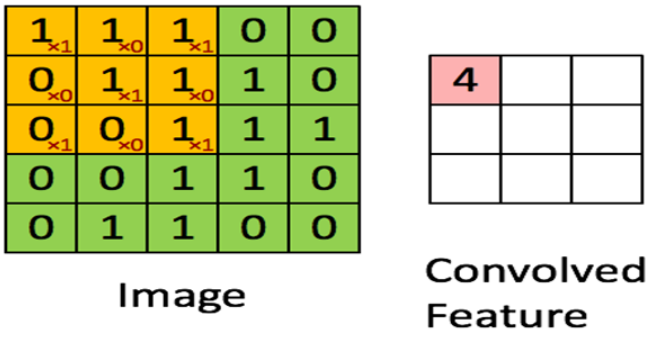

Fig.5.3 convoluting a 5x5x1 image with a 3x3x1 kernel to get a 3x3x1 convolved feature

- The filter moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.

POOLING LAYER

- The Pooling layer is responsible for reducing the spatial size of the Convolved Feature.
- It is even useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

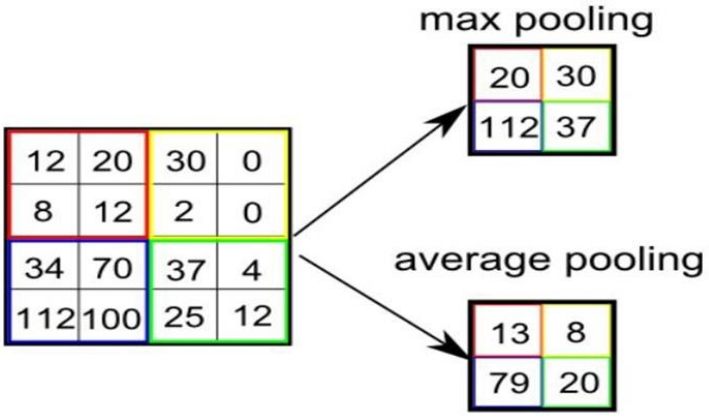

Fig.5.4 selecting maximum and average pooling matrix

- There are two types of Pooling: Max Pooling and Average Pooling.
- Max Pooling returns the maximum value from the portion of the image covered by the Kernel.
- On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel.

Chapter 6

CODING

6.1 LAYOUT CODE

This code module includes the necessary part of code that is used to create the layout of the application developed. It consists of 3 buttons and 1 image view component. The 3 buttons are used for choosing the image, analysing the image and viewing the remedies respectively.

```
import tkinter as tk
from tkinter.filedialog import askopenfilename
import shutil
import os
from PIL import Image, ImageTk


window = tk.Tk()


window.title("PLANT DISEASE DETECTION")


window.geometry("500x510")
window.configure(background="white")


title = tk.Label(text="Click below to choose picture for testing disease....", background = "white", fg="Brown", font=("", 15))
title.grid()


img = cv2.imread('1.JPG')
```

```
imghsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

cv2.imshow ('HSV Image', imghsv)

cv2.imshow ('HSV Image', imghsv[:, :, 0])
cv2.imshow ('Saturation', imghsv[:, :, 1])
cv2.imshow ('Value', imghsv[:, :, 2])

cv2.waitKey (0)

cv2.destroyAllWindows ()


img = cv2.imread('3.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
ret,thresh=cv2.threshold(img,0,255,cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)

plt.imshow(img)
plt.imshow(thresh)
plt.show()

img = cv2.imread('3.jpg')
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

```
ret, thresh = cv2.threshold(img, 0, 255,
cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)

# Noise removal
img = np.ones((3,3),np.uint8)
opening = cv2.morphologyEx(thresh,cv2.MORPH_OPEN,img, iterations = 2)
plt.imshow(opening)
plt.show()

# Sure background area
sure_bg = cv2.dilate (opening, img,iterations=3)

# Finding sure foreground area

img = cv2.distanceTransform(opening, cv2.DIST_L2, 5)
ret, sure_fg = cv2.threshold(img, 0.7*img.max(), 255, 0)
plt.imshow(sure_bg)
plt.show()

# Finding unknown region
sure_fg = np.uint8(sure_fg)
plt.imshow(sure_fg)
plt.show()

unknown = cv2.subtract(sure_bg, sure_fg)
```

```
# Marker labelling
ret, markers = cv2.connectedComponents(sure_fg)

# Add one to all labels so that sure background is not 0, but 1
markers = markers+1

# Now, mark the region of unknown with zero
markers[unknown == 255] = 0

markers = cv2.watershed(img, markers)
img[markers == -1] = [255, 0, 0]

plt.imshow(markers)
plt.show()
nemo = cv2.imread('1.jpg')
plt.imshow(nemo)
plt.show()

nemo = cv2.cvtColor(nemo, cv2.COLOR_BGR2RGB)
plt.imshow(nemo)
plt.show()

from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
```

```
from matplotlib import colors


r, g, b = cv2.split(nemo)

fig = plt.figure()

axis = fig.add_subplot(1, 1, 1, projection="3d")


pixel_colors = nemo.reshape((np.shape(nemo)[0]*np.shape(nemo)[1], 3))

norm = colors.Normalize(vmin=-1.,vmax=1.)

norm.autoscale(pixel_colors)

pixel_colors = norm(pixel_colors).tolist()


axis.scatter(r.flatten(), g.flatten(), b.flatten(), facecolors=pixel_colors, marker=".")

axis.set_xlabel("Red")

axis.set_ylabel("Green")

axis.set_zlabel("Blue")

plt.show()


h, s, v = cv2.split(nemo)

fig = plt.figure()

axis = fig.add_subplot(1, 1, 1, projection="3d")


axis.scatter(h.flatten(), s.flatten(), v.flatten(), facecolors=pixel_colors, marker=".")

axis.set_xlabel("Hue")

axis.set_ylabel("Saturation")
```

```
axis.set_zlabel("Value")

plt.show()


light_orange = (36, 210, 14)

dark_orange = (37, 215, 15)


#Displaying the HSV color choosen


from matplotlib.colors import hsv_to_rgb

lo_square = np.full((10, 10, 3), light_orange, dtype=np.uint8) / 255.0

do_square = np.full((10, 10, 3), dark_orange, dtype=np.uint8) / 255.0


mask = cv2.inRange(nemo, light_orange, dark_orange)


result = cv2.bitwise_and(nemo, nemo, mask=mask)


light_white = (0, 0, 200)

dark_white = (145, 60, 255)


mask_white = cv2.inRange(nemo, light_white, dark_white)

result_white = cv2.bitwise_and(nemo, nemo, mask=mask_white)



final_mask = mask + m
```

```
final_result = cv2.bitwise_and(nemo, nemo, mask=final_mask)


ask_white


blur = cv2.GaussianBlur(final_result, (7, 7), 0)


def bact():
    window.destroy()
    window1 = tk.Tk()


    window1.title("PLANT DISEASE DETECTION")


    window1.geometry("500x510")
    window1.configure(background="white")


    def exit():
        window1.destroy()
    rem = "The remedies for Bacterial Spot are:\n\n "
    remedies = tk.Label(text=rem, background="white", fg="Brown", font=("", 15))
    remedies.grid(column=0, row=7, padx=10, pady=10)
    rem1 = " Discard or destroy any affected plants. \n  Do not compost them. \n
    Rotate yoour tomato plants yearly to prevent re-infection next year. \n Use copper
    fungicites"
    remedies1 = tk.Label(text=rem1, background="white", fg="Black", font=("", 12))
```

```
    remedies1.grid(column=0, row=8, padx=10, pady=10)


    button = tk.Button(text="Exit", command=exit)
    button.grid(column=0, row=9, padx=20, pady=20)
    window1.mainloop()


def vir():
    window.destroy()
    window1 = tk.Tk()


    window1.title("Dr. Plant")


    window1.geometry("650x510")
    window1.configure(background="lightgreen")


    def exit():
        window1.destroy()
    rem = "The remedies for Yellow leaf curl virus are: "
    remedies = tk.Label(text=rem, background="lightgreen", fg="Brown", font=("",
    15))
    remedies.grid(column=0, row=7, padx=10, pady=10)
    rem1 = " Monitor the field, handpick diseased plants and bury them. \n  " \
           "Use sticky yellow plastic traps. \n  Spray insecticides such as
            organophosphates, carbametes during the seedliing stage. \n" \
```

```
        " Use copper fungicites"
    remedies1 = tk.Label(text=rem1, background="lightgreen",
                         fg="Black", font=("", 12))
    remedies1.grid(column=0, row=8, padx=10, pady=10)

    button = tk.Button(text="Exit", command=exit)
    button.grid(column=0, row=9, padx=20, pady=20)
    window1.mainloop()

def latebl():
    window.destroy()
    window1 = tk.Tk()
    window1.title("PLANT DISEASE DETECTION")
    window1.geometry("520x510")
    window1.configure(background="white")

    def exit():
        window1.destroy()
    rem = "The remedies for Late Blight are: "
    remedies = tk.Label(text=rem, background="white", fg="Brown", font=("", 15))
    remedies.grid(column=0, row=7, padx=10, pady=10)

    rem1 = " Monitor the field, remove and destroy infected leaves. \n  Treat
    organically with copper spray. \n  " \
```

```
        "Use chemical fungicides,the best of which for tomatoes is chlorothalonil."
    remedies1 = tk.Label(text=rem1, background="white", fg="Black", font=("", 12))
    remedies1.grid(column=0, row=8, padx=10, pady=10)


    button = tk.Button(text="Exit", command=exit)
    button.grid(column=0, row=9, padx=20, pady=20)
    window1.mainloop()


def analysis():
    import cv2
    import numpy as np
    import os
    # from random import shuffle
    from tqdm import tqdm
    verify_dir = 'testpicture'
    IMG_SIZE = 50
    LR = 1e-3
    MODEL_NAME = 'healthyvsunhealthy-{}-{}.model'.format(LR, '2conv-basic')


    def process_verify_data():
        verifying_data = []
        for img in tqdm(os.listdir(verify_dir)):
            path = os.path.join(verify_dir, img)
            img_num = img.split('.')[0]
```

```
        img = cv2.imread(path, cv2.IMREAD_COLOR)
        img = cv2.resize(img, (IMG_SIZE, IMG_SIZE))
        verifying_data.append([np.array(img), img_num])
    np.save('verify_data.npy', verifying_data)
    return verifying_data

verify_data = process_verify_data()
verify_data = np.load('verify_data.npy')

import tflearn
from tflearn.layers.conv import conv_2d, max_pool_2d
from tflearn.layers.core import input_data, dropout, fully_connected
from tflearn.layers.estimator import regression
import tensorflow as tf
tf.reset_default_graph()

convnet = input_data(shape=[None, IMG_SIZE, IMG_SIZE, 3], name='input')

convnet = conv_2d(convnet, 32, 3, activation='relu')
convnet = max_pool_2d(convnet, 3)

convnet = conv_2d(convnet, 64, 3, activation='relu')
convnet = max_pool_2d(convnet, 3)
```

```
convnet = conv_2d(convnet, 128, 3, activation='relu')

convnet = max_pool_2d(convnet, 3)


convnet = conv_2d(convnet, 32, 3, activation='relu')

convnet = max_pool_2d(convnet, 3)


convnet = conv_2d(convnet, 64, 3, activation='relu')

convnet = max_pool_2d(convnet, 3)


convnet = fully_connected(convnet, 1024, activation='relu')

convnet = dropout(convnet, 0.8)


convnet = fully_connected(convnet, 4, activation='softmax')

convnet = regression(convnet, optimizer='adam', learning_rate=LR,

loss='categorical_crossentropy', name='targets')


model = tflearn.DNN(convnet, tensorboard_dir='log')


if os.path.exists('{}.meta'.format(MODEL_NAME)):

    model.load(MODEL_NAME)

    print('model loaded!')


import matplotlib.pyplot as plt
```

```
fig = plt.figure()


for num, data in enumerate(verify_data):


    img_num = data[1]

    img_data = data[0]


    y = fig.add_subplot(3, 4, num + 1)

    orig = img_data

    data = img_data.reshape(IMG_SIZE, IMG_SIZE, 3)

    # model_out = model.predict([data])[0]

    model_out = model.predict([data])[0]


    if np.argmax(model_out) == 0:

        str_label = 'healthy'

    elif np.argmax(model_out) == 1:

        str_label = 'bacterial'

    elif np.argmax(model_out) == 2:

        str_label = 'viral'

    elif np.argmax(model_out) == 3:

        str_label = 'lateblight'


    if str_label == 'healthy':

        status = "HEALTHY"
```

```
else:
    status = "UNHEALTHY"


message = tk.Label(text='Status:'+status, background="lightgreen", fg="Brown",
font=("", 15))
message.grid(column=0, row=3, padx=10, pady=10)
if str_label == 'bacterial':
    diseasename = "Bacterial Spot "
    disease = tk.Label(text='Disease Name:' + diseasename,
    background="lightgreen", fg="Black", font=("", 15))
    disease.grid(column=0, row=4, padx=10, pady=10)
    r = tk.Label(text='Click below for remedies...', background="lightgreen",
    fg="Brown", font=("", 15))
    r.grid(column=0, row=5, padx=10, pady=10)
    button3 = tk.Button(text="Remedies", command=bact)
    button3.grid(column=0, row=6, padx=10, pady=10)
elif str_label == 'viral':
    diseasename = "Yellow leaf curl virus "
    disease = tk.Label(text='Disease Name: ' + diseasename,
    background="lightgreen", fg="Black", font=("", 15))
    disease.grid(column=0, row=4, padx=10, pady=10)
    r = tk.Label(text='Click below for remedies...', background="lightgreen",
   fg="Brown", font=("", 15))
    r.grid(column=0, row=5, padx=10, pady=10)
```

```
            button3 = tk.Button(text="Remedies", command=vir)
            button3.grid(column=0, row=6, padx=10, pady=10)
            elif str_label == 'lateblight':
            diseasename = "Late Blight "
            disease = tk.Label(text='Disease Name: ' + diseasename,
            background="lightgreen", fg="Black", font=("", 15))
            disease.grid(column=0, row=4, padx=10, pady=10)
            r = tk.Label(text='Click below for remedies...', background="lightgreen",
            fg="Brown", font=("", 15))
            r.grid(column=0, row=5, padx=10, pady=10)
            button3 = tk.Button(text="Remedies", command=latebl)
            button3.grid(column=0, row=6, padx=10, pady=10)
        else:
            r = tk.Label(text='Plant is healthy', background="lightgreen", fg="Black",
            font=("", 15))
            r.grid(column=0, row=4, padx=10, pady=10)
            button = tk.Button(text="Exit", command=exit)
            button.grid(column=0, row=9, padx=20, pady=20)


def openphoto():
    dirPath = "testpicture"
    fileList = os.listdir(dirPath)
    for fileName in fileList:
```

```
        os.remove(dirPath + "/" + fileName)
    fileName = askopenfilename(initialdir='C:/Users/Lokesh/Downloads/images',
    title='Select image for analysis ', filetypes=[('image files', '.jpg')])
    dst = "C:/Users/Lokesh/PycharmProjects/lokesh1/testpicture"
    shutil.copy(fileName, dst)
    load = Image.open(fileName)
    render = ImageTk.PhotoImage(load)
    img = tk.Label(image=render, height="250", width="500")
    img.image = render
    img.place(x=0, y=0)
    img.grid(column=0, row=1, padx=10, pady=10)
    title.destroy()
    button1.destroy()
    button2 = tk.Button(text="Analyse Image", command=analysis)
    button2.grid(column=0, row=2, padx=10, pady=10)


button1 = tk.Button(text="Get Photo", command=openphoto)
button1.grid(column=0, row=1, padx=10, pady=10)

window.mainloop()
```

Chapter 7

SYSTEM TESTING

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. There are various types of testing, explained in the following sections.

7.1 Testing Objective

- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding an as yet undiscovered error.
- A successful test one that uncovers an as –yet undiscovered error.

The above objectives imply a dramatic change in viewpoint. They move counter to the commonly held view that a successful test is one in which no errors are found. Testing cannot show the absence of detects, it can only show that software errors are present.

7.2 White Box Testing

White-box testing (also known as clear box testing, glass box testing, transparent box testing, or structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality as in Black Box Testing Using white box testing methods, the software engineer can derive test cases that:

- Guarantee that all independent paths with in a module have been exercised at least once.
- Exercise all logical decisions on their true and false sides.
- Execute all loops at their boundaries and within their validity.
- Logical errors and incorrect assumptions are inversely proportional to the probability that a program path will be executed.

- Often believe that logical path not likely to execute when, in fact, it may be executed on regular basis.

7.3 Levels of Testing

The different levels of testing that are to be conducted are:

- Code Testing
- Program Testing
- System Testing

7.3.1 Code Testing: The code test has been conducted to test the logic of the program. Here, we have tested with all possible combinations of data to find out logical errors. The code testing is done thoroughly with all possible data available with library.

7.3.2 Program Testing: Program testing is also called unit testing. The modules in the system are integrated to perform the specific function. The modules have been tested independently, later Assembled and tested thoroughly for integration between different modules.

7.3.3 System Testing: System testing has been conducted to test the integration of each module in the system. It is used to find discrepancies between the system and its original objective. It is found that there is an agreement between current specifications and system documentation. Software Testing is carried out in three steps.

The first step includes unit testing where in each module is tested to provide his correctness, validity and also determine any missing operations. Errors are noted down and corrected immediately. Unit testing is the import and major part of the project. So errors are rectified easily in particular module and program clarity is increased. In this project entire system is divided into several modules and is developed individually. So unit testing is conducted to individual modules.

The second step includes integration testing. It need not be the case, the software whose modules when run individually and showing perfect results, will also show perfect results when run as a whole. The individual modules are clipped under this major module and tested again and the results are verified.

The final step involves validation and testing which determines the software functions as the user expected. Here also there may be some modifications. In the completion of the project it is satisfied fully by the user.

7.4 Integration Test Plan

Data can be lost across an interface, one module can have an adverse effort on the other sub functions, when combined, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance.

7.5 Unit Testing

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example, the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Table 7.1: Unit Testing Table

Function Name	**Tests Results**
Uploading Image	Tested for uploading different types and sizes of images.
Detecting Disease	Tested for different images of plant leaves and diseases Bacterial Spot, yellow leaf curl virus.
Get Remedies	Tested if the remedies are displayed successfully.

7.6 Output Testing

After performance of the validation testing, the next step is output testing. The output displayed or generated by the system under consideration is tested by asking the user about the format required by system.

Table 7.2: Output Testing Table

Functionality to be tested	Input	Tests Results
Working of Choose File option	User convenience to access images stored	Different folders where images are stored can be uploaded
Working of View Remedies	User need to click the Remedies button	Details of disease remedies are displayed.

Chapter 8

RESULTS

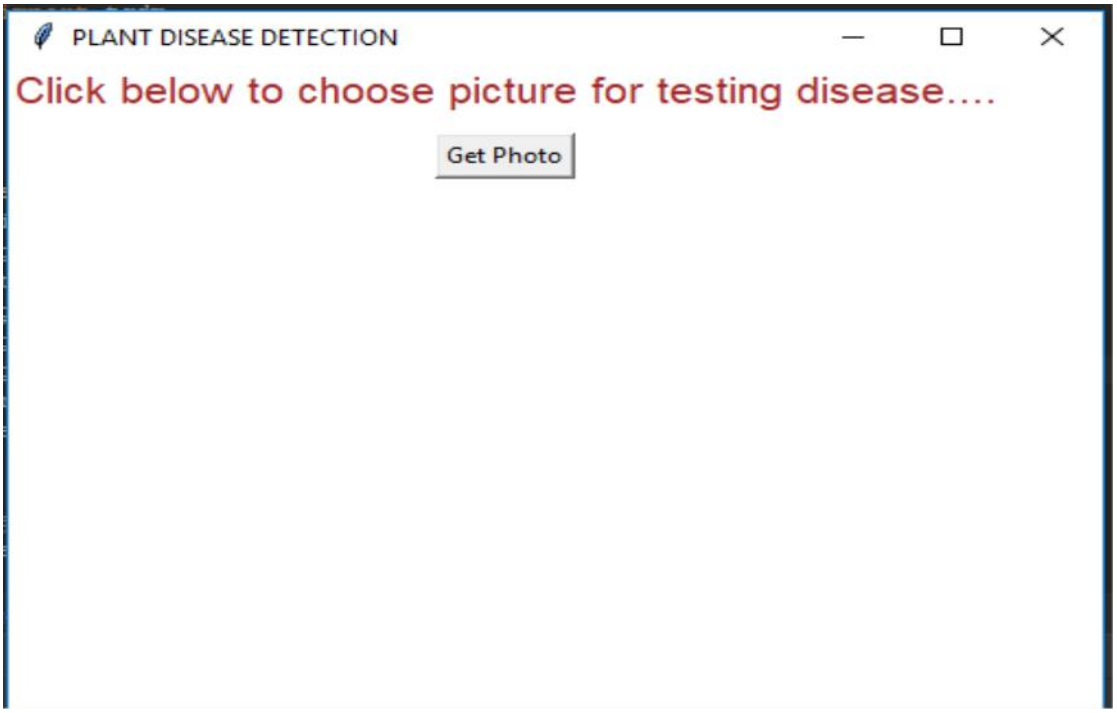

Fig.8.1. Home page

Figure 8.1 depicts the first look of our front end. We have a text message called “click below to choose picture for testing” so that a user can understand to click the below button. It has a button called “Get Photo” which can be used to browse the images on the system’s hard disk.

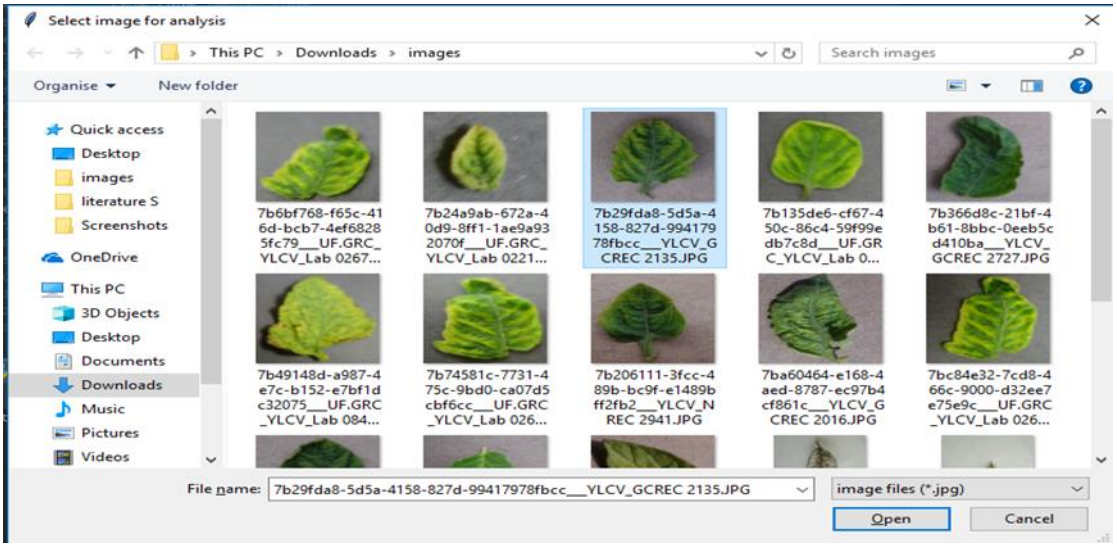

Fig.8.2. Selecting input from dataset

FIG. 8.2 shows the popup which appears when user clicks on 'get photo' button. The popup window will be having number of input images to be selected, this action should be confirmed with a double click or an open button.

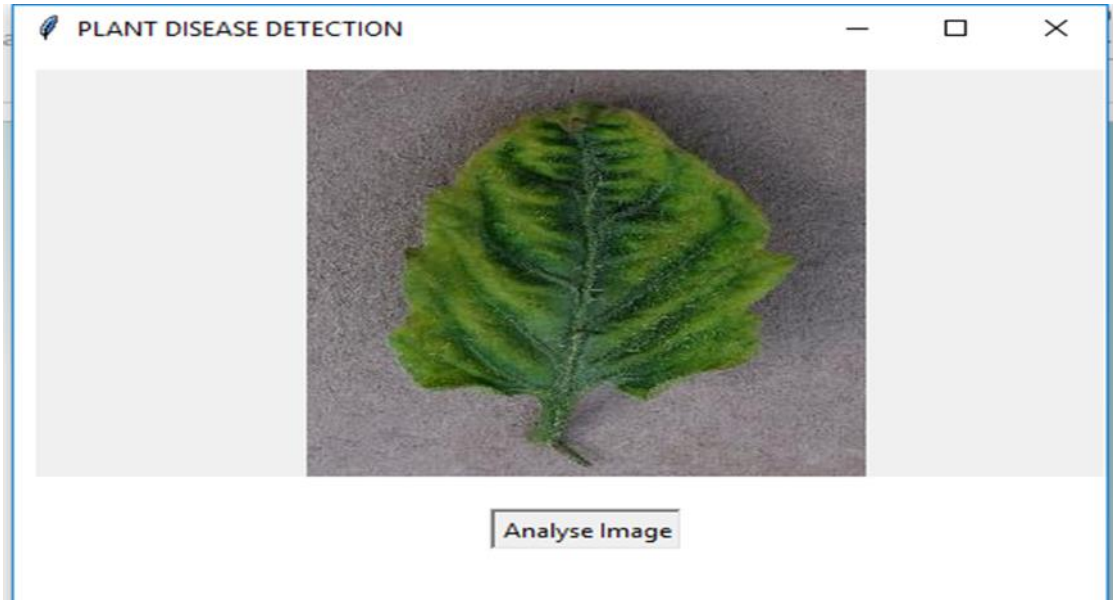

Fig.8.3 Analysing input data

FIG. 8.3 shows the selected image from the system directory, there will be a button provided called as 'Analyse Image' for analysing the input image to detect the condition of the leaf.

Fig.8.4 Result after analysis of input data

FIG. 8.4 displays the status of leaf i.e. Healthy or Unhealthy, if it is unhealthy it displays the particular disease name. Here is a button called 'Remedies' by clicking this button it displays the Remedies for particular diseases.

Fig.8.4 Remedies window

FIG. 8.5 suggest the Remedies for the disease found in the leaf.

Chapter 9

CONCLUSION AND FUTURE WORK

9.1 CONCLUSION

There are number of ways by which we can detect disease of plants and suggest remedies for them. Each has some pros as well as limitations .On one hand visual analysis is least expensive and simple method, it is not as efficient and reliable. Image processing is a technique which is most spoken for very high accuracy and least time consumption are major advantages offered. The applications of K-means clustering and Neural Networks (NNs) have been formulated for clustering and classification of diseases that effect on plant leaves. Recognizing the disease accurately and efficiently is mainly the purpose of the proposed approach. The experimental results indicate that the proposed approach is a valuable approach, which can significantly support an accurate detection of leaf diseases in a little computational effort. Alongside the supply of cultivation tools, the farmers also need access to accurate information that they can use for efficient crop management and there is no better way than providing them a service that they can use through the software.

9.2 FUTURE WORK

- To improve recognition rate of final classification process hybrid algorithms like Artificial Neural Network, Bayes classifier, Fuzzy Logic can also be used.
- Mobile application can be developed which is handy and easy to use.
- An extension of this work will focus on automatically estimating the severity of the detected disease.
- As future enhancement of the project is to develop the open multimedia (Audio/Video) about the diseases and their solution automatically once the disease is detected.

References

1. A survey on crop disease detection using image processing technique for economic growth of rural area. **Yashpal Sen1, Chandra Shekhar Mithlesh2, Dr. Vivek Baghel3**

2. K. Elangoran, S. Nalini, 2011 "Detection and classification of leaf diseases using K- means-based segmentation and neural-networks-based classification." Inform. Technol. J., 10: 267-275. DOI: 10.3923/itj.2011.267.275.

3. Sandesh Raut, Karthik Ingale, "Review on leaf disease detection using Image Processing techniques."

4. "A Survey on Methods of Plant Disease Detection" **Sagar Patil, Anjali Chandavale**

5. T. RUMPF, A-K Mahlein, U sleiner, H. W. Dehne. "Texture analysis for diagnosing paddy disease." In International Conference on Electrical Engineering and Informatics, 2009. ICEEI'09., vol. 1, pp. 23-27. IEEE, 2009.

6. "Plant disease detection and classification using image processing and artificial neural networks." **Mr. Sanjay Mirchandani1, Mihir Pendse2, Prathamesh Rane3, Ashwini Vedula4**

7. "Detection and Classification of Plant Leaf Diseases Using Image Processing Techniques:" **Savita N. Ghaiwat, Parul Arora**

8. "Image Processing Based Leaf Rot Disease, Detection of Betel Vine (Piper Betel.)" **Amar Kumar Deya*, Manisha Sharmaa, M.R. Meshramb**

9. "Advances in image processing for plant disease detection" **Jayamala k Patil, Rajkumar**

10. S Arivazhagan, R Newlin shebiah, S Ananthi, S Vishnu varthini "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features."