PLANTS DISEASE DETECTION USING CNN

A PROJECT REPORT

Submitted by

ROHINI.S [REGISTER NO: 211417104226]

PADMASHREE.M[REGISTER NO: 211417104176]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI- 600123.

ANNA UNIVERSITY: CHENNAI 600 025

AUGUST 2021

BONAFIDE CERTIFICATE

Certified that this project report **"PLANTS DISEASE DETECTION USING CNN"** is the bonafide work of **"S.ROHINI (211417104226), M.PADMASHREE (211417104176) "** who carried out the project under my supervision

SIGNATURE

Dr.S.MURUGAVALLI,M.E.,Ph.D.,
HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
COLLEGE
NAZARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE

S.T.SANTHANALAKSHMI,M.Tech
SUPERVISOR
Professor CSE Department

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING
NAZARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on 6th August 2021.

INTERNALEXAMINER

EXTERNALEXAMINER

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI,Thiru.C.SAKTHIKUMAR,M.E.,**

and **Tmt.SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank our **Project Guide Mrs.Santhanalakshmi M.Tech.,** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

ROHINI.S

PADMASHREE.M

ABSTRACT

Crop cultivation plays an essential role in the agricultural field. Presently, the loss of food is mainly due to infected crops, which reflexively reduces the production rate. To identify the plant diseases at an untimely phase is not yet explored. The main challenge is to reduce the usage of pesticides in the agricultural field and to increase the quality and quantity of the production rate. Proposed system explore the leaf disease prediction at an untimely action. The enhanced CNN algorithm to predict the infected area of the leaves. A color based segmentation model is defined to segment the infected region and placing it to its relevant classes. Experimental analyses were done on samples images in terms of time complexity and the area of infected region. Plant diseases can be detected by image processing technique. Disease detection involves steps like image acquisition, image pre-processing, image segmentation, feature extraction and classification.

LIST OF FIGURES

LIST OF ABBREVIATIONS

S.NO	ABBREVIATIONS	EXPLAINATION
1	**CNN**	Convolutional Neural Network
2	**NSI**	New Spectral Indices
3	**PMI**	Powdery Mildew-Index
4	**DI**	Disease Index
5	**YRI**	Yellow Rust Index
6	**API**	Application Programming Interface
7	**SGD**	Stochastic Gradient Descent
8	**GUI**	Graphical User Interface
9	**STING**	Statistical Information Grid
10	**ML**	Machine Learning

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 Overview

Neural network are made out of basic components . These components are handled by the nervous system of the human body. As in nature, the components are associated in such a way that they operate the working of the system to large extent. You can prepare a neural system to play out a specific capacity by changing the estimations of the associations (weights) between components. Regularly, neural network are balanced, or prepared, with the goal that a specific info prompts a particular target yield. The following figure shows such a circumstance. Here, the system is balanced, in light of an examination of the yield and the objective, until the point that the system yield coordinates the objective. In order to train a network the input/target are necessary

1.2 Scope of Project

The primary occupation in India is agriculture. India ranks second in the agricultural output worldwide. Here in India, farmers cultivate a great diversity of crops. Various factors such as climatic conditions, soil conditions, various disease, etc. affect the production of the crops. The existing system can Only identify the type of diseases which affects the leaf. It's not efficient. Result will be provided within fraction of seconds and guided throughout the project. In recent years, the rapid development of artificial intelligence has made life more convenient, and AI has become a well-known technology. For example, AlphaGo defeated the world champion of Go. Siri and Alexa as voice assistants of Apple and Amazon are all applications of artificial intelligence technology represented by deep learning in various fields.

As the key research object of computer vision and artificial intelligence, image recognition has been greatly developed in recent years. In agricultural applications, the goal of image.

recognition is to identify and classify different types of pictures, and analyze the types of crops, disease types, severity and so on. Then we can formulate corresponding countermeasures to solve various problems in agricultural production in a timely and efficient manner. So as to further ensure and improve the yield of crops and help the better development of agriculture. With the rapid development of deep learning, especially in image recognition, speech analysis, natural language processing and other fields, it shows the uniqueness and efficiency of deep learning. Compared with the traditional methods, deep learning is more efficient in the diagnosis of crop diseases in the field of agricultural production .The deep learning model canmonitor, diagnose and prevent the growth of crops in time. Image recognition of crop diseases and insect pests can reduce the dependence on plant protection technicians in agricultural production, so that farmers can solve the problem in time. Compared with artificial identification, the speed of intelligent network identification is much faster than that of manual detection. And the recognition accuracy is getting higher and higher in the continuous development. The establishment of a sound agricultural network and the combination of Internet and agricultural industry cannot only solve the problems related to crop yield affected by diseases and insect pests, but also be conducive to the development of agricultural informatization.

CHAPTER 2

LITERATURE SURVEY

[1] K. Padmavathi, and K. Thangadurai, "Implementation of RGB and Gray scale images in plant leaves disease detection –comparative study," Indian J. of Sci. and Tech., vol. 9, pp. 1- 6, Feb. 2016.

In recent days, agriculturalists faces many problems because of the disease that affected on plants. Due to the crop disease, Year productivity of farmers gets lower. Manual monitoring of plant disease tends to require tremendous amount of work and also require the excessive processing time. Hence, the method used in this paper is image processing. The steps that are implemented here is image acquisition, segmentation, feature extraction and classification of plant diseases.

[2] Kiran R. Gavhale, and U. Gawande, "An Overview of the Research on Plant Leaves International Journal of Pure and Applied Mathematics Special Issue 882 Disease detection using Image Processing Techniques

Diseases in plants cause major production and economic losses as well as reduction in both quality and quantity of agricultural products. Now a day's plant diseases detection has received increasing attention in monitoring large field of crops. Farmers experience great difficulties in switching from one disease control policy to another. The naked eye observation of experts is the traditional approach adopted in practice for detection and identification of plant diseases. In this paper we review the need of simple plant leaves disease detection system that would facilitate advancements in agriculture. Early information on crop health and disease detection can facilitate the control of diseases through proper management strategies. This technique will improves productivity of crops. This paper also compares the benefits and limitations of these potential methods. It includes several steps viz. image acquisition, image pre-processing, features extraction and neural network based classification.

[3] Y. Q. Xia, Y. Li, and C. Li, "Intelligent Diagnose System of Wheat Diseases Based on Android Phone," J. of Infor. & Compu. Sci., vol.12,pp.6845-6852,Dec.2015.

The most important factor in reduction of quality and quantity of crop is due to plant disease. Identifying plant disease is a key to prevent agricultural losses. The aim of this paper is to develop a software solution which automatically detect and classify plant disease. Agricultural productivity is something on which economy highly depends. This is the one of the reasons that disease detection in plants plays an important role in agriculture field, as having disease in plants are quite natural. If proper care is not taken in this area then it causes serious effects on plants and due to which respective product quality, quantity or productivity is affected. Detection of plant disease through some automatic technique is beneficial as it reduces a large work of monitoring in big farms of crops, and at very early stage itself it detects the symptoms of diseases i.e. when they appear on plant leaves. A System has been proposed which can detect the disease and provide cure using an android mobile application. In the system the photos of plant leaves are captured and are sent to the cloud server, which is further processed and compared with the diseased plant leaf images in the cloud database. Based on the comparison a list of plant diseases suspected are given to the user via the android mobile application.

[4]Wenjiang Huang, Qingsong Guan, JuhuaLuo, Jingcheng Zhang, Jinling Zhao, Dong Liang, Linsheng Huang, and Dongyan Zhang, "New Optimized Spectral Indices for Identifying and Monitoring Winter Wheat Diseases IEEE journal of selected topics in applied earth observation and remote sensing,Vol. 7, No. 6, June 2014

The vegetation indices from hyperspectral data have been shown to be effective for indirect monitoring of plant diseases. However, a limitation of these indices is that they cannot distinguish different diseases on crops. We aimed to develop new spectral indices (NSIs) that would be useful for identifying different diseases on crops. Three different pests (powdery mildew, yellow rust, and aphids) in winter wheat were used in this study. The new optimized spectral indices were derived from a weighted combination of a single band and a normalized wavelength difference of two bands. The most and least relevant wavelengths for different diseases were first extracted from leaf spectral data using the RELIEF-F algorithm.

Reflectance of a single band extracted from the most relevant wavelengths and the normalized wavelength difference from all possible combinations of the most and least relevant wavelengths were used to form the optimized spectral image.

The classification accuracies of these new indices for healthy leaves and leaves infected with powdery mildew, yellow rust, and aphids were 86.5%, 85.2%, 91.6%, and 93.5%, respectively. We also applied these NSIs for non imaging canopy data of winter wheat, and the classification results of different diseases were promising. For the leaf scale, the powdery mildew-index (PMI)correlated well with the disease index (DI), supporting the use of the PMI to invert the severity of powdery mildew. For the canopy scale, the detection of the severity of yellow rust using the yellow rust- index (YRI) showed a high coefficient of determination (${mbi{R}}^{bf 2 =}{bf 0.86}$) between the estimated DI and its observations, suggesting that the NSIs may improve disease detection in precision agriculture application.

[5]Monica Jhuria, Ashwani Kumar, an RushikeshBorse, "Image Processing For Smart Farming: Detection Of Disease And Fruit Gradin ", Proceedings of the 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013).

Due to the increasing demand in the agricultural industry, the need to effectively grow a plant and increase its yield is very important. In order to do so, it is important to monitor the plant during its growth period, as well as, at the time of harvest. In this paper image processing is used as a tool to monitor the diseases on fruits during farming, right from plantation to harvesting. For this purpose artificial neural network concept is used. Three diseases of grapes and two of apple have been selected. The system uses two image databases, one for training of already stored disease images and the other for implementation of query images. Back propagation concept is used for weight adjustment of training database. The images are classified and mapped to their respective disease categories on basis of three feature vectors, namely, color, texture and morphology. From these feature vectors morphology gives 90% correct result and it is more than other two feature vectors. This paper demonstrates effective algorithms for spread of disease and mango counting. Practical implementation of neural networks has been done using MATLAB.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

The existing system can only identify the type of diseases which affects the leaf. Its not efficient. Result will be provided within fraction of seconds and guided throughout the project. Samples of 1000 images are collected that comprised of different plant diseases like Alternaria Alternata, Anthracnose, Bacterial Blight, Cercospora leaf spot and Healthy Leaves.Different number of images is collected for each disease that was classified into dataset and input images.

3.2PROPOSED SYSTEM:

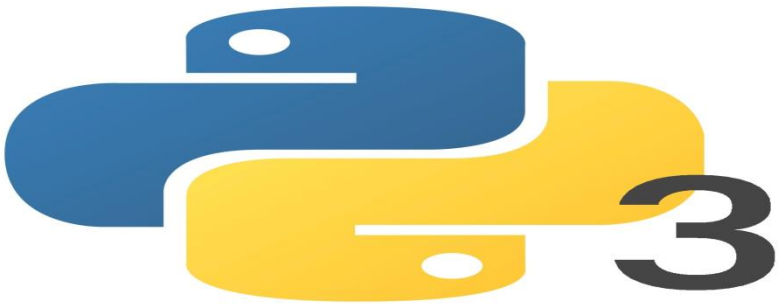
Plant diseases are detected and the solutions to recover from the leaf diseases will be provided. The affected part of the leaf are shown by image processing technique.

3.3HARWARE REQUIREMENTS:

Processor	: Intel Pentium Dual Core 2.00GHz Hard
disk	: 500 GB
RAM	: 8 GB (minimum).

3.4SOFTWARE REQUIREMENTS:

- Python 3.6.4 Version



3.5 TECHNOLOGY STACK:

- **PYTHON**
- **NUMPY.**
- **SCIKIT-LEARNING.**
- **TENSORFLOW &KERAS.**
- **JUPYTER NOTEBOOK.**

PYTHON

Python is a free ,open source programming language. Python is also across- platform compatible language. Python is also a great visualization tool. It provideslibraries such as Mat plot lib, sea born and bokeh to create stunning visualizations.In addition, Python is the most popular language for machine learning and deep learning. As a matter of fact , today , all top organizations are investing in Python to implement machine learning in the back-end.

NUMPY

NumPy is the core library for scientific computing in Python. It provides a high- performance multidimensional array object, and tools for working with these arrays. NumPyis a Python library that is the core library for scientific computing in Python. It contains a collection of tools and techniques that can be used to solve on a computer mathematical models of problems in Science and Engineering. One of these tools is a high- performance multi dimensional array object that is a powerful data structure for efficient computation of arrays and matrices.

SCIKIT-LEARN

Scikit-learn (formerly **scikits.learn**and also known as **sklearn**) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn integrates well with many other Python libraries, such as mat plot lib and plot lyfor plotting, NumPy for array vectorization, pandas data frames, SciPy, and many more.

TENSORFLOW &KERAS:

Tensor Flow is a Python library for fast numerical Computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. It was created and is maintained by Google and released under the Apache 2.0 open-source license. The API is nominally for the Python programming language, although there is access to the under lying C++ API.Kerasisan API designed for human beings, notmachines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of useractions required for common use cases, and it provides clear & actionable errormessages. It also has extensive documentation and developer guides.

JUPYTER NOTEBOOK:

The Jupyter Notebook is an open-source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter. A Notebook extension (nbextension) is a Java Script module that you load in most of the views in the Notebook's frontend. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones. Jupyter Notebook provides you with an easy-to-use, interactive data science environment across many programming languages that doesn't only work **as an IDE**, but also as a presentation or education tool.

SOFTWARE SPECIFICATION

3.5.1 Deep Learning

Deep learning (**ML**) is the study of computer algorithms that improve automatically through experience and by the use of data. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based

on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

The types of machine learning algorithms are mainly divided into four categories:

- **Supervised learning,**
- **Un-supervised learning,**
- **Semi-supervised learning,**
- **Reinforcement learning**.

- **Supervised learning**

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task. Some popular examples of supervised machine learning algorithms are: Linear regression for regression problems. Random forest for classification and regression problems. Support vector machines for classification problems.

associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

CLASSIFICATION

As the name suggests, Classification is the task of "classifying things" into sub- categories. But, by a machine. If that doesn't sound like much, imagine your computer being able to differentiate between you and a stranger. Between a potato and a tomato. Between an A grade and a F. In Machine Learning and Statistics, Classification is the problem of identifying to which of a set of categories (sub populations), a new observation belongs to, on the basis of a training set of data containing observations and whose categories membership is known.

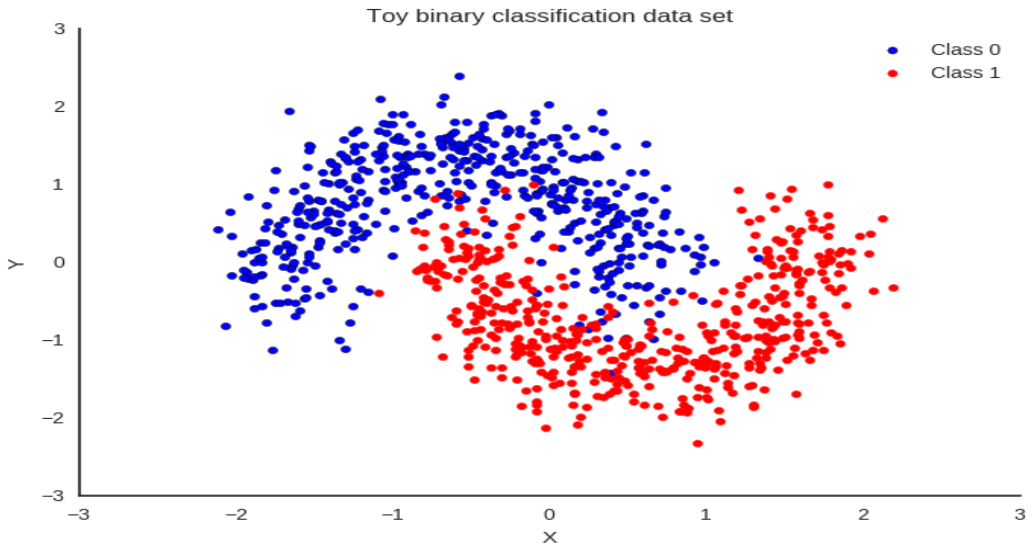
TYPES OF CLASSIFICATION

Classification is of two types:

- Binary Classification
- Multiclass Classification

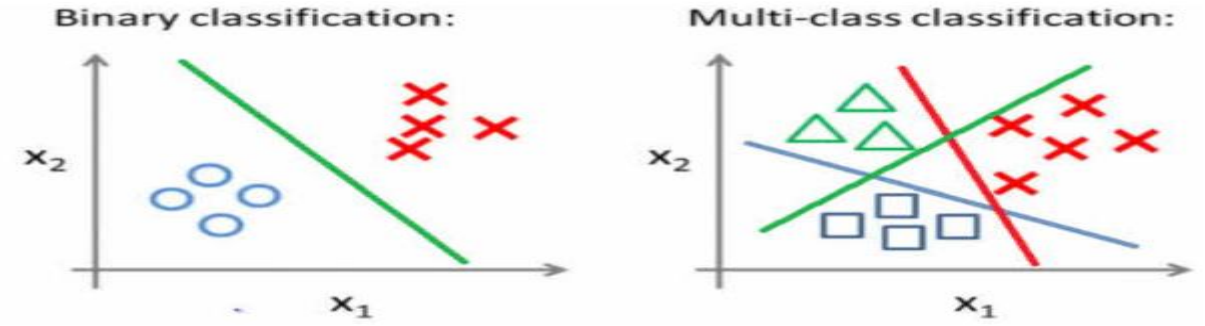
Binary Classification

When we have to categorize given data into 2 distinct classes. Example – On the basis of given health conditions of a person, we have to determine whether the person has a certain disease or not. Binary classification is the task of classifying the elements of a set into two groups on the basis of a classification rule. The most popular algorithms used by the binary classification are- Logistic Regression k-Nearest Neighbors Decision Trees Support Vector Machine Naive Bayes.



Multiclass Classification

The number of classes is more than 2. For Example



– On the basis of data about different species of flowers, we have to determine which specie does our observation belong to

Binary and Multiclass Classification. Here x1 and x2 are our variables upon which the class is predicted. Suppose we have to predict whether a given patient has a certain disease or not, on the basis of 3 variables, called features. Which means there are two possible outcomes:

1. The patient has the said disease. Basically a result labelled “Yes” or “True”.

2. The patient is disease free. A result labelled “No” or “False”.

This is a binary classification problem.We have a set of observations called training data set, which comprises of sample data with actual classification results. We train a model, called Classifier on this data set, and use that model to predict whether a certain patient will have the

1. X : pre-classified data, in the form of a N*M matrix. N is the no. of observations and M is the number of features

2. y : An N-d vector corresponding to predicted classes for each of the N observations.

3. Feature Extraction : Extracting valuable information from input X using a series of transforms.

4. ML Model : The “Classifier” we’ll train.

5. y’ : Labels predicted by the Classifier.

6. Quality Metric : Metric used for measuring the performance of the model.

7. ML Algorithm : The algorithm that is used to update weights w', which update the model and "learns" iteratively.

Types of Classifiers (Algorithms)

There are various types of classifiers. Some of them are :

- Linear Classifiers : Logistic Regression
- Tree Based Classifiers : Decision Tree Classifier
- Support Vector Machines
- Artificial Neural Networks
- Bayesian Regression
- Gaussian Naive Bayes Classifiers
- Stochastic Gradient Descent (SGD) Classifier
- Ensemble Methods : Random Forests, AdaBoost, Bagging Classifier, Voting Classifier, ExtraTrees Classifier

Practical Applications of Classification

- Google's self driving car uses deep learning enabled classification techniques which enables it to detect and classify obstacles.
- Spam E-mail filtering is one of the most widespread and well recognized uses of Classification techniques.
- k-Nearest Neighbors.
- Decision Trees.
- Naive Bayes.
- Random Forest.
- Gradient Boosting.

- Detecting Health Problems, Facial Recognition, Speech Recognition, Object Detection, Sentiment Analysis all use Classification at their core.

REGRESSION

A regression problem is when the output variable is a real or continuous value, such as "salary" or "weight". Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper- plane which goes through the points.

- **Un-supervised learning**

Un-supervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features

CLUSTERING

It is basically a type of unsupervised learning method. An unsupervised learning method is a method in which we draw references from datasets consisting of input datawithoutlabelled responses.

In machine learning too, we often group examples as a first step to understand a subject (data set) in a machine learning system. Grouping unlabeled examples is called clustering. As the examples are unlabeled, clustering relies **on unsupervised machine learning**.

Clustering is an unsupervised machine learning method of identifying and grouping similar data points in larger datasets without concern for the specific outcome. Clustering (sometimes called cluster analysis) is usually used to classify data into structures that are more easily understood and manipulated.

Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of

examples. Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is

basically a collection of objects on the basis of similarity and dissimilarity between them.For example, the data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture

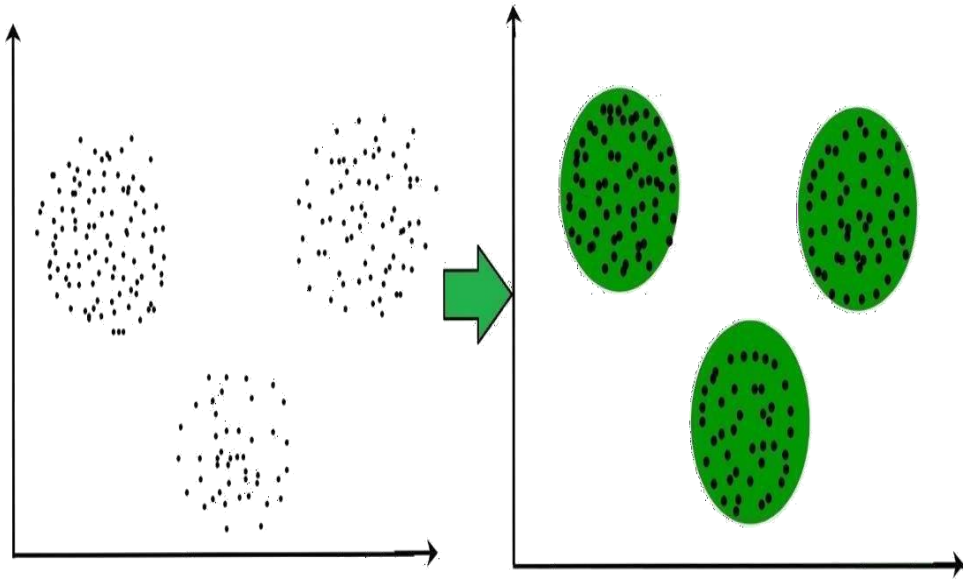


Fig 3.6.2 Clustering

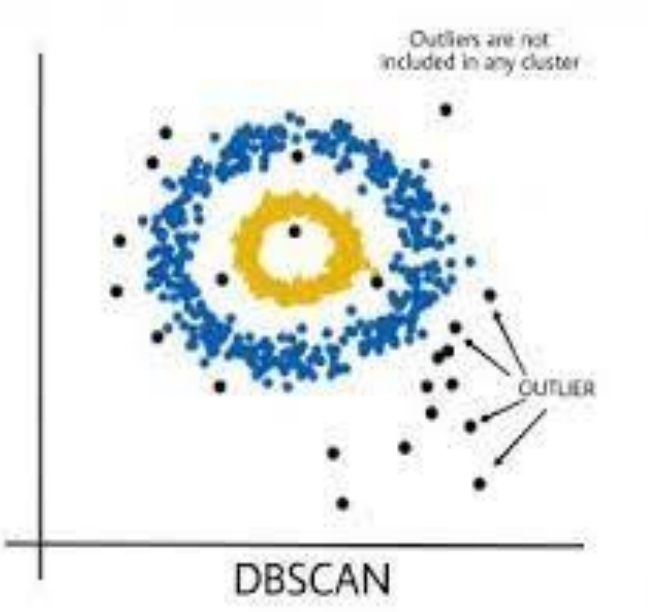
These data points are clustered by using the basic concept that the data point lies within the given constraint from the cluster center. Various distance methods and techniques are used for calculation of the outliers. Cluster analysis, or clustering, is an unsupervised machine learning task. It **involves automatically discovering natural grouping in data**. Unlike supervised learning (like predictive modeling), clustering algorithms only interpret the input data and find natural groups or clusters in feature space.

Clustering is very much important as it determines the intrinsic grouping among the unlabeled data present. There are no criteria for a good clustering. It depends on the user, what is the criteria they may use which satisfy their need. For instance, we could be interested in finding representatives for homogeneous groups (data reduction), in finding "natural clusters" and describe their unknown properties ("natural" data types), in finding useful and suitable groupings ("useful" data classes) or in finding unusual data objects (outlier detection). This algorithm must make some assumptions which constitute the similarity of points and each assumption make different and equally valid clusters.

3.6.5.1.1 Clustering Methods :

1. **Density-Based Methods :** These methods consider the clusters as the dense region having some similarity and different from the lower dense region of the space. These methods have good accuracy and ability to merge two clusters. Example

- ❖ DBSCAN(Density-Based Spatial Clustering of Applications with Noise) ,
- ❖ OPTICS (OrderingPoints to Identify Clustering Structure) .

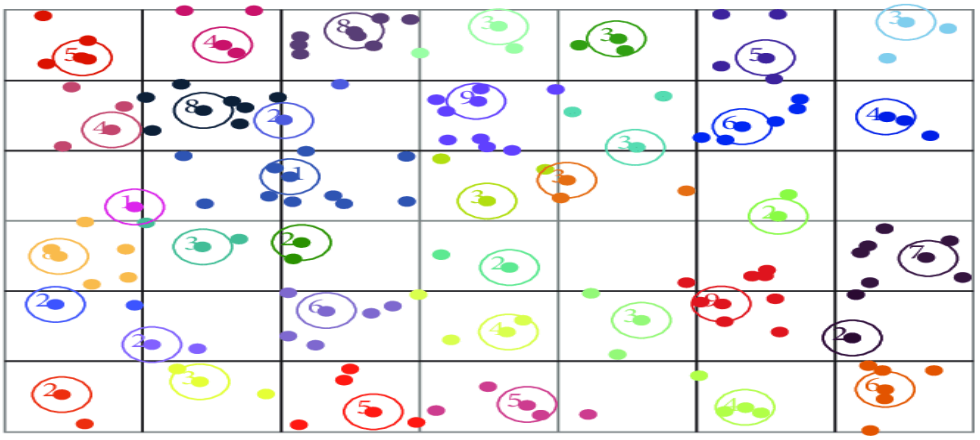


2. **Hierarchical Based Methods :** The clusters formed in this method forms a tree type structure based on the hierarchy. New clusters are formed using the previously formed one. It is divided into two category

- Agglomerative (bottom up approach)
- Divisive (top down approach) .

3. **Partitioning Methods :** These methods partition the objects into k clusters and each partition forms one cluster. This method is used to optimize an objective criterion similarity function such as when the distance is a major parameter example K-means, CLARANS (Clustering Large Applications based upon randomized Search) etc.

4. **Grid-based Methods :** In this method the data space are formulated into a finite number of cells that form a grid-like structure. All the clustering operation done on these grids are fast and independent of the number of data objects example STING (Statistical Information Grid), wave cluster, CLIQUE (CLusteringIn Quest) etc.



Clusters = 37

Clustering Algorithms:

- K-Means Clustering.
- Mean-Shift Clustering for a single sliding window.

- The entire process of Mean-Shift Clustering.
- DBSCAN Smiley Face Clustering.
- EM Clustering using GMMs.
- Agglomerative Hierarchical Clustering.

- **Semi-supervised learning**

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

- **Reinforcement learning**

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agentsystems, swarmintelligence, statistics and genetic algorithms. In machine learning, the environment is typically represented as a Markov decision process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible. Reinforcement learning is the training of machine learning models to make a sequence of decisions. The agent learns to achieve a goal in an uncertain, potentially complex environment. In reinforcement learning, an artificial intelligence faces a game-like situation. Its goal is to maximize the total reward. Example: The problem is as follows: We have an agent and a reward, with many hurdles in between. Reinforcement Learning is a subset of machine learning. It enables an agent to learn through the consequences of actions in a specific environment. It can be used to **teach a robot new tricks**.

3.6.5.2 ANACONDA

Anaconda is a free and open source distribution of the Python and programming languages for data science and machine learning related applications (large-scale data processing, predictive analytics, scientific computing), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. Anaconda Distribution is used by over 6 million users, and it includes more than 250 popular data science packages suitable for Windows, Linux, and MacOS.

Python is a high-level programming language devised by Guido van Rossum & first released in 1991. It's the most popular coding language used by software developers to build, control, manage and for testing. It is also an interpreter which executes Python programs. The python interpreter is called python.exe on Windows.

Python Packages

Packages or additional libraries help in scientific computing and computational modelling. In Python, the packages are not the part of the Python standard library. Few major packages are

- numpy (NUMeric Python): matrices and linear algebra .
- scipy(SCIentific Python): many numerical routines.
- matplotlib: (PLOTting LIBrary) creating plots of data.
- sympy (SYMbolic Python): symbolic computation .
- pytest(Python TESTing): a code testing framework.

Together with a list of Python packages, tools like editors, Python distributions include the Python interpreter. Anaconda is one of several Python distributions. Anaconda is a new distribution of the Python and R data science package. It was formerly known as Continuum Analytics. Anaconda has more than 100 new packages.

This work environment, Anaconda is used for scientific computing, data science, statistical analysis, and machine learning. The latest version of Anaconda 5.0.1 is released in October 2017.The released version 5.0.1 addresses some minor bugs and adds useful features, such as updated R language support. All of these features weren't available in the original 5.0.0 release.

This package manager is also an environment manager, a Python distribution, and a collection of open source packages and contains more than 1000 R and Python Data Science Packages.

IPYTHON NOTEBOOKS

IPython is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history. To launch a Jupyter notebook, open your terminal and navigate to the directory where you would like to save your notebook. Then type the command jupyter notebook and the program will instantiate a local server at localhost:8888 (or another specified port). It is an interactive computational environment, in which you can combine code execution, rich text, mathematics, plots and rich media. IPython continued to exist as a Python shell and kernel for Jupyter, but the notebook interface and other language-agnostic parts of IPython were moved under the Jupyter name. The IPython console is now deprecated and if you want to start it, you'll need to use the Jupyter Console, which is a terminal-based console frontend for Jupyter kernels.

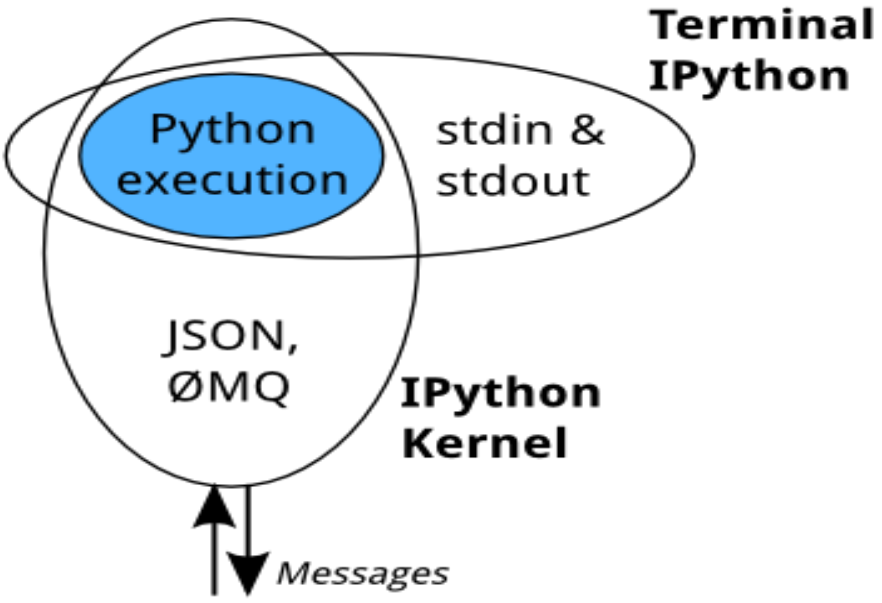
IPython provides the following features:

- Interactive shells (terminal and Qt-based)

- A browser-based notebook interface with support for code, text,mathematical expressions, inline plots and other media**.**
- Support for interactive data visualization and use of GUI toolkits.
- Flexible, embeddable interpreters to load into one's own projects.
- Tools for parallel computing.

IPython is based on an architecture that provides parallel and distributed computing. IPython enables parallel applications to be developed, executed, debugged and monitored interactively. Hence, the I (Interactive) in IPython.[3]This architecture abstracts out parallelism, which enables IPython to support many different styles of parallelism[4]including:

With the release of IPython 4.0, the parallel computing capabilities have been made optional and released under the ipyparallel python package.IPython frequently draw from SciPy stack[5] libraries like NumPy and SciPy, often installed alongside from one of many Scientific Python distributions. IPython provide integration some library of the SciPy stack like matplotlib, like inline graph when in used with the Jupyter notebook. Python libraries can implement IPython specific hooks to customize object Rich object display. SymPy for example implement rendering of Mathematical Expression as rendered LaTeX when used within IPython context.



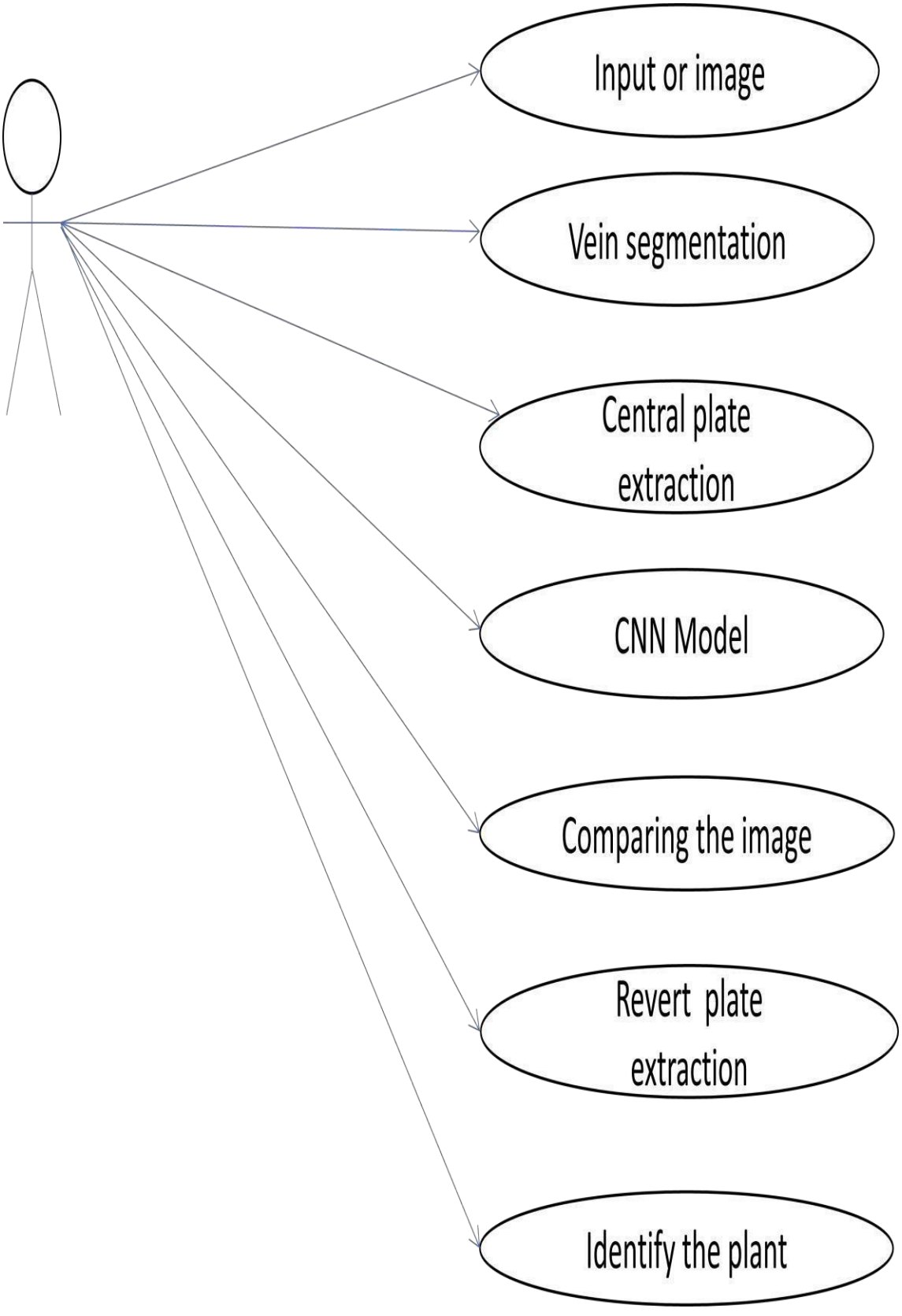
Feature:

IPython also allows non-blocking interaction with Tkinter, PyGTK, PyQt/PySide and wxPython (the standard Python shell only allows interaction with Tkinter). IPython can interactively manage parallel computing clusters using asynchronous status call-backs and/or MPI. IPython can also be used as a system shell replacement. Its default behaviour is largely similar to Unix shells, but it allows customization and the flexibility of executing code in a live Python environment. Using IPython as a shell replacement is less common and it is now recommended to use Xonsh which provide most of the IPython feature with better shell integrations. IPython has many features for tab-completion, object introspection, system shell access, command history retrieval across sessions, and its own special command system for adding functionality when working interactively. IPython is a growing project, with increasingly language-agnostic components. IPython 3.x was the last monolithic release of IPython, containing the notebook server, qtconsole, etc. As of IPython 4.0, the language-agnostic parts of the project: the notebook format, message protocol, qtconsole, notebook web application, etc. have moved to new projects under the name Jupyter. IPython itself is focused on interactive Python, part of which is providing a Python kernel for Jupyter. IPython (Interactive Python) is a command shell for interactive computing in multiple programming languages, originally developed for the Python programming language, that offers introspection, rich media, shell syntax, tab completion, and history.

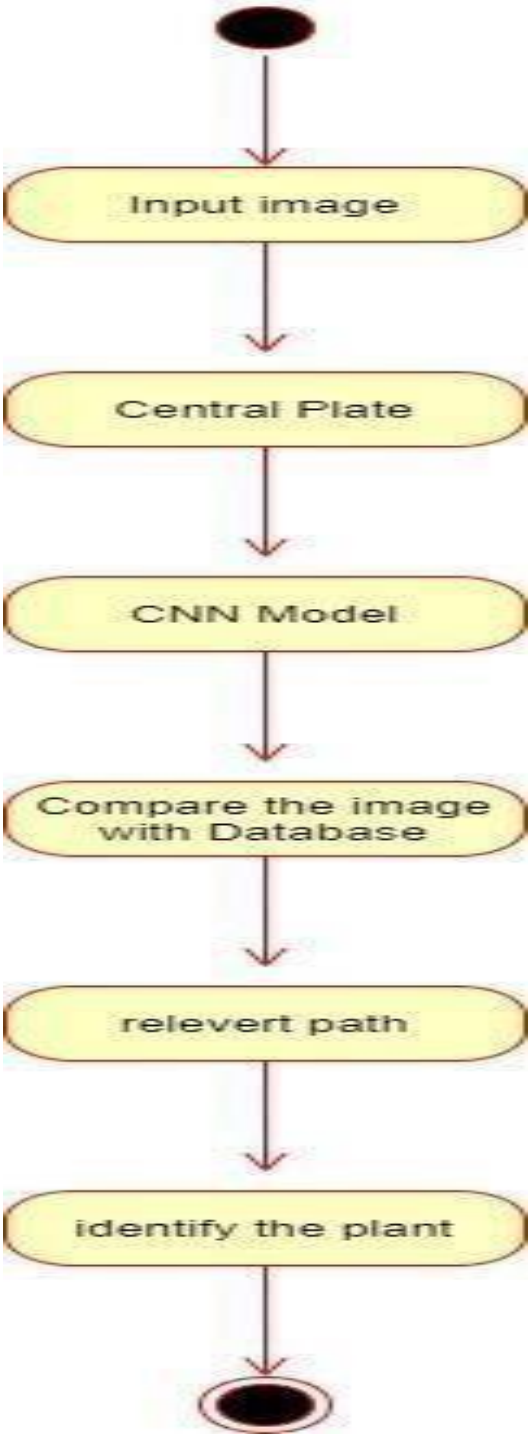
CHAPTER 4

SYSTEM DESIGN

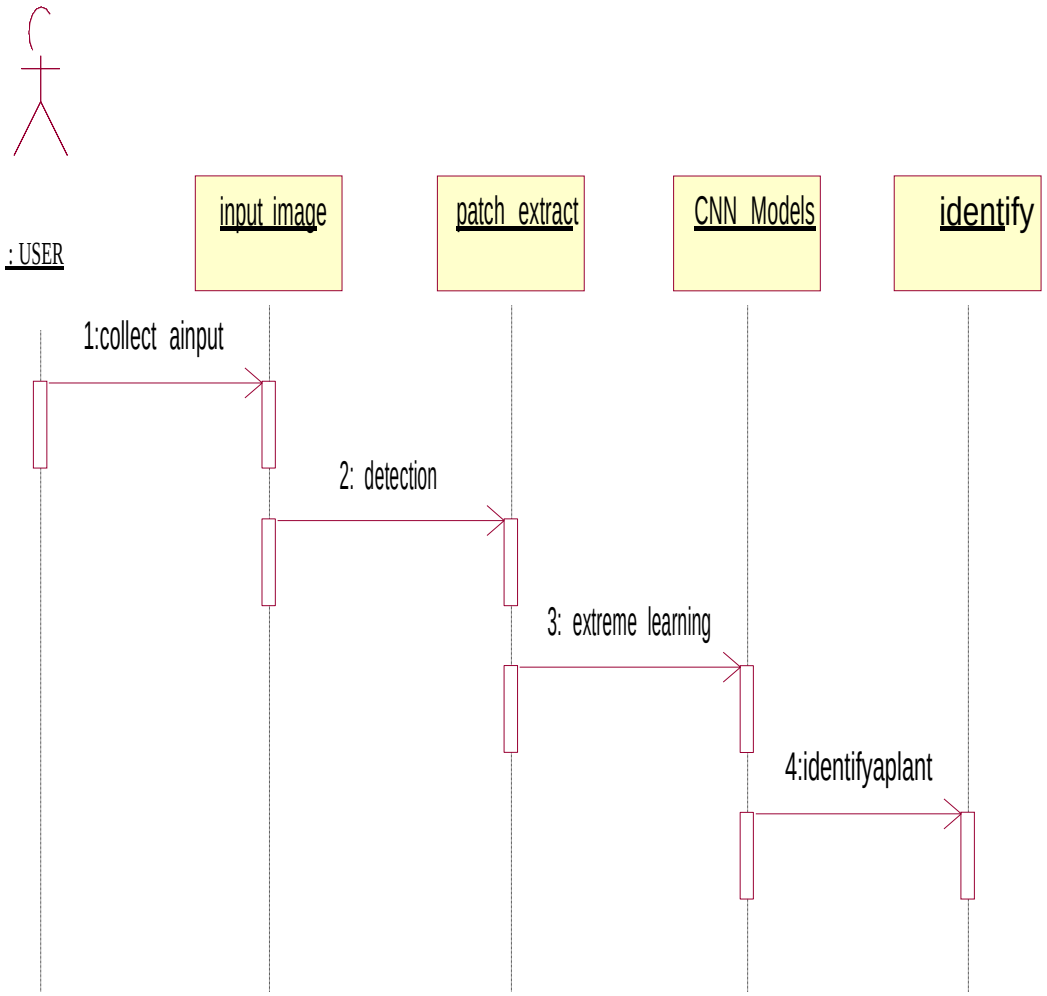
4.1 USECASE DIAGRAM



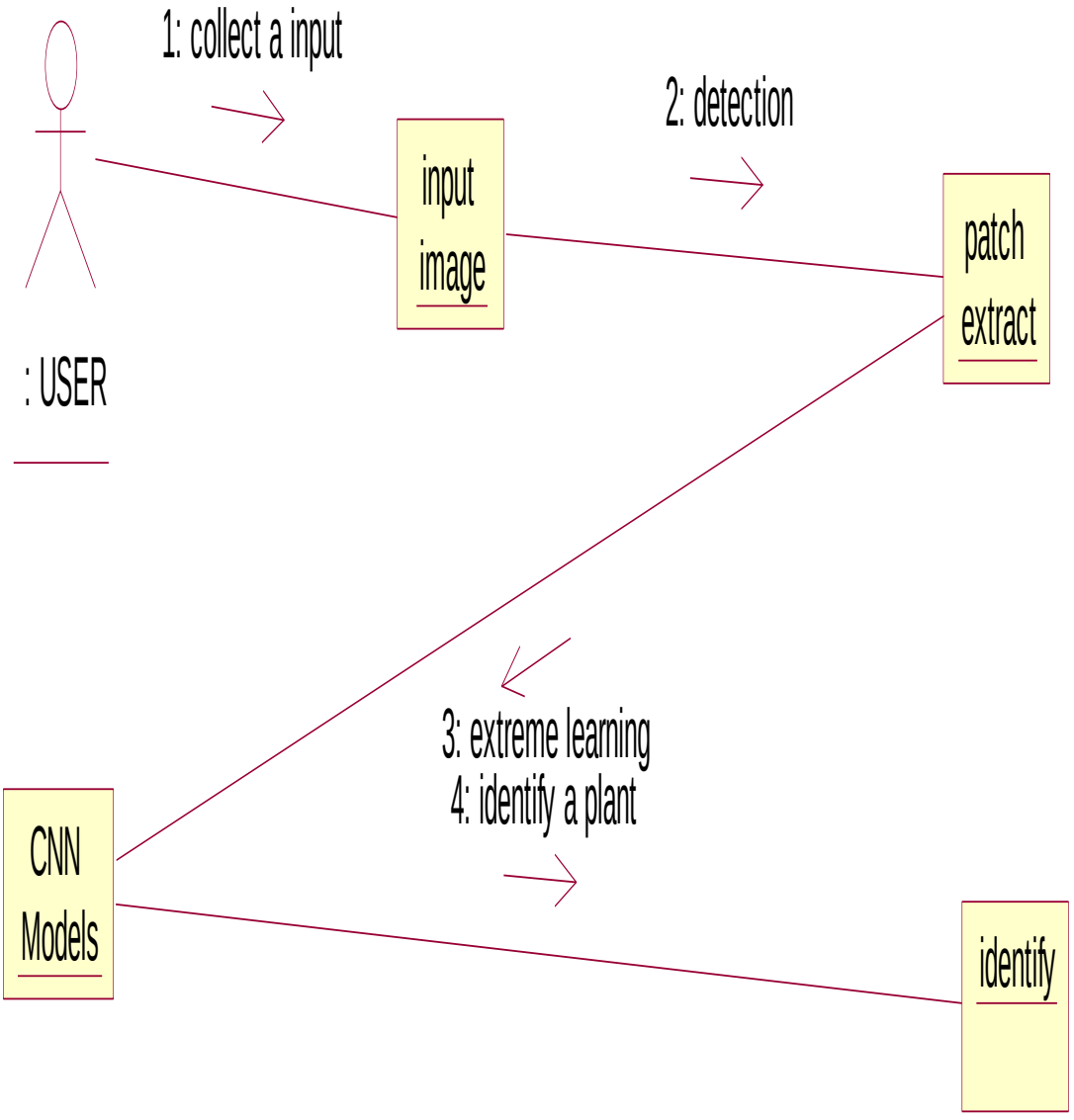
4.2 ACTIVITY DIAGRAM



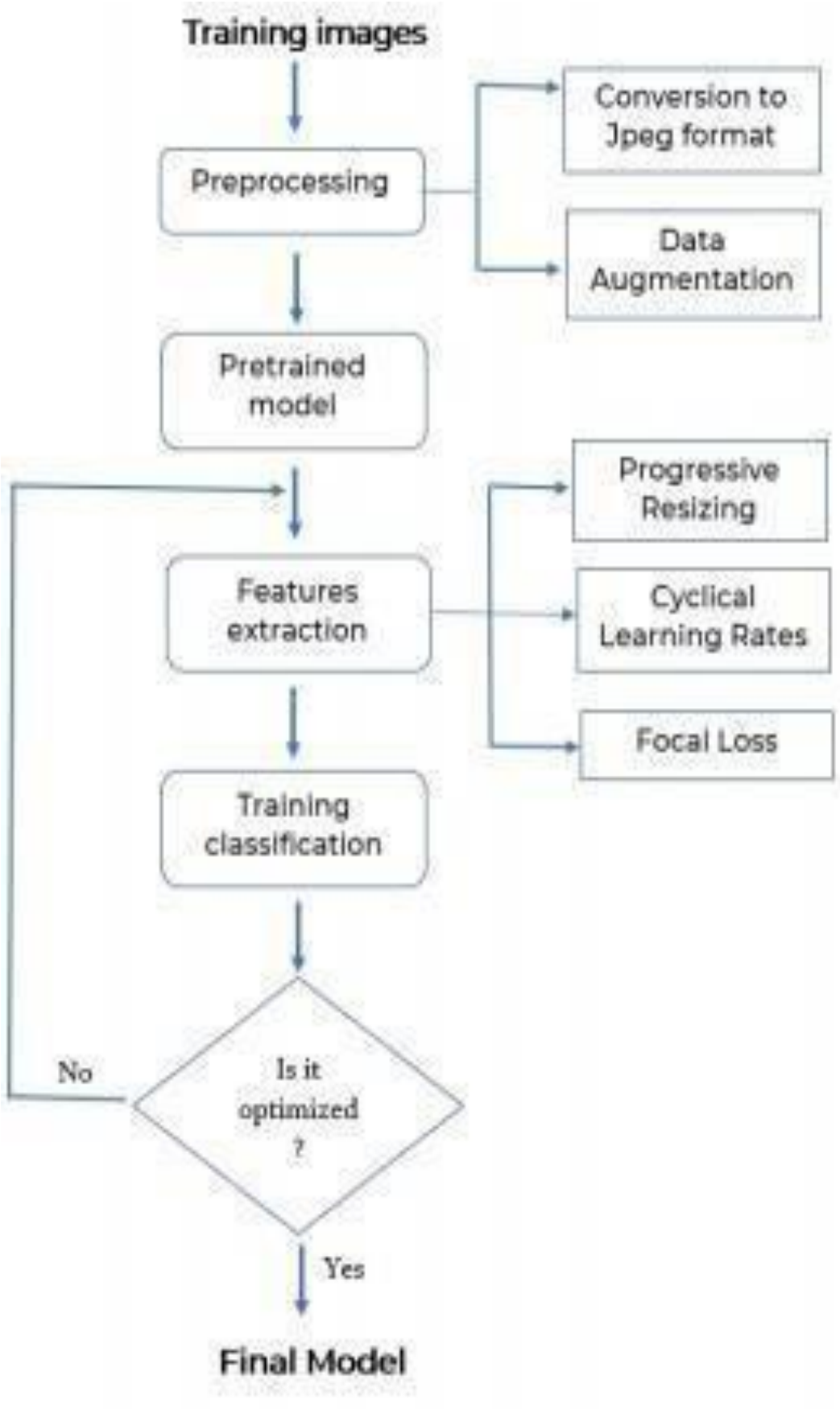
4.3 SEQUENCE DIAGRAM



4.4 COLLABORATION DIAGRAM



4.5 FLOWCHART DIAGRAM



CHAPTER 5

SYSTEM ARCHITECTURE

5.1 ARCHITECTURE OVERVIEW

Initially the data set is divided into three namely: Training Data, Testing data and Validation data .The images from the Testing Dataset undergo feature extraction process to extract all the required feature to analyze the image and to produce the appropriate result. Then those data will be provided further for the feature selection and the system is trained using deep learning algorithm. Once training is complete, the system is ready for authenticating the users.

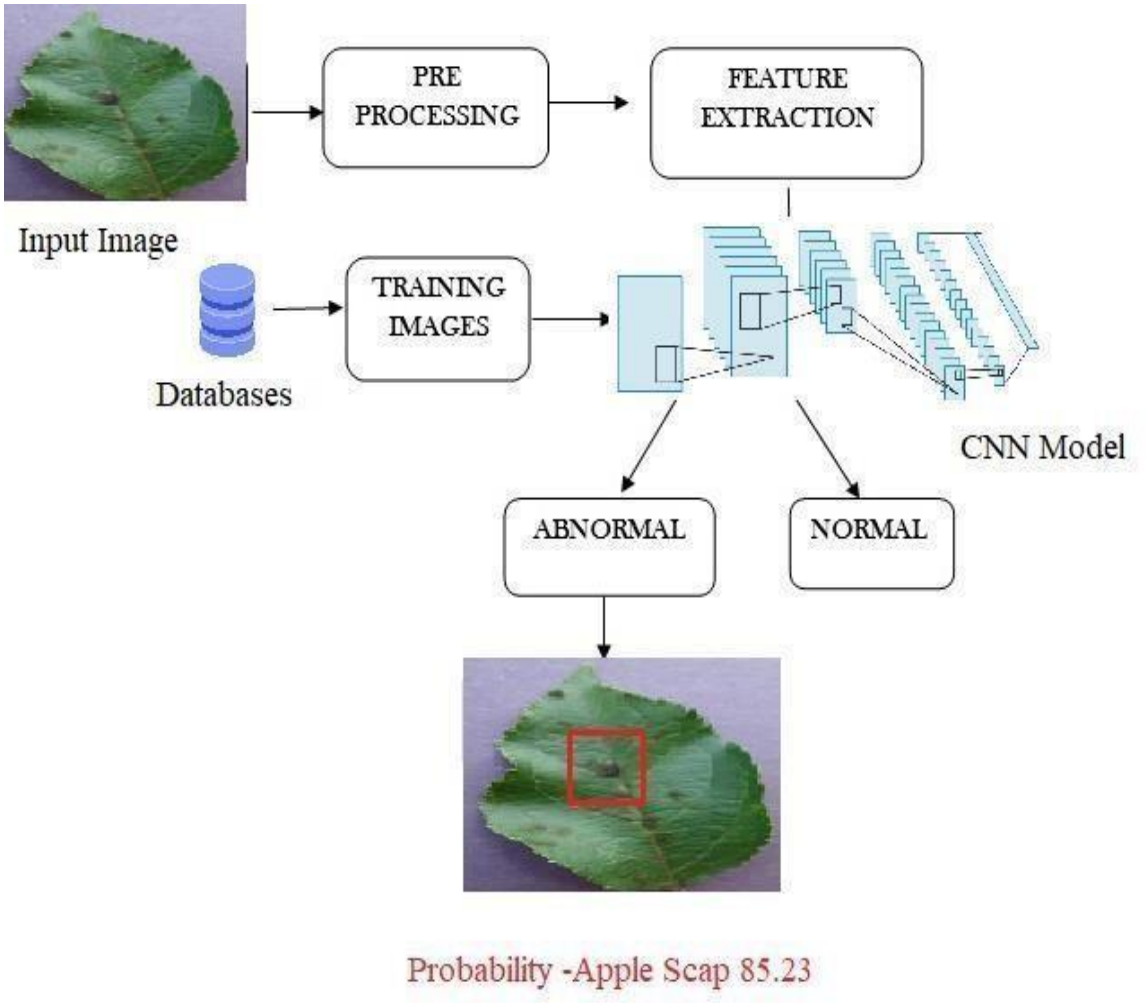


Fig 5.1.1 System Architecture

5.2 MODULE

- Image Acquisition
- Image Preprocessing
- Image Segmentation
- Feature Extraction

5.3 MODULE DESCRIPTION

Image Acquisition

- The initial process is to collect the data from the public repository.
- It takes the image as input for further processing.
- Popular image domains are take so that any formats can be given as input to the process (.bmp, .jpg, .gif).

Fig 5.3 Image Acquisition

Image Preprocessing

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

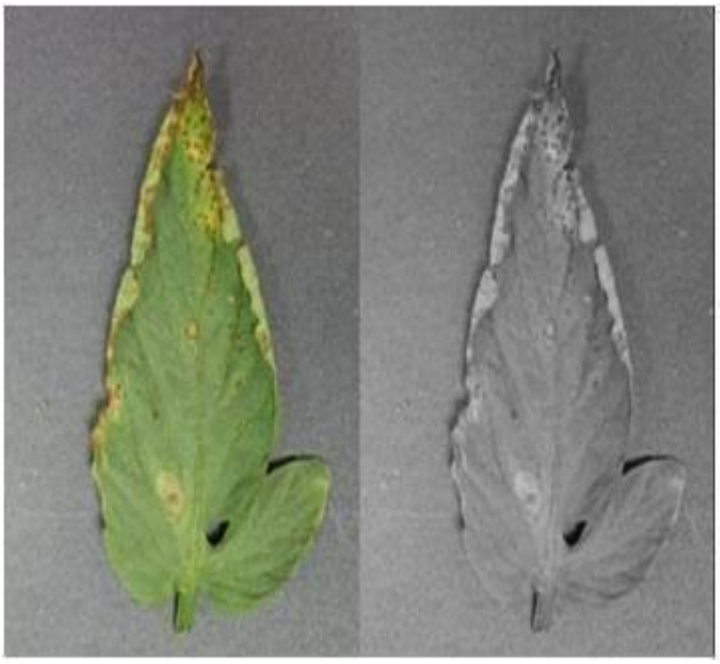


Fig 5.4 Image Preprocessing

Image Segmentation

- Image segmentation is the third step in the 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.

Fig 5.5 Image Segmentation

Feature extraction

- Feature extraction is the important part to gracefully predict the infected region.
- Here shape and textural feature extraction is done.
- The shape oriented feature extraction like Area, Color axis length, eccentricity, solidity and perimeter are calculated.
- Similarly the texture oriented feature extraction like contrast, correlation, energy, homogeneity and mean is captured and processed to determine the health of each plant.

Fig 5.6 Feature Extraction

CHAPTER 6

SYSTEM IMPLEMENTATION

```
import os
import cv2
import numpy as np import
pandas as pd from PIL import
Image
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split from tensorflow
import keras
from keras.models import Sequential

from keras.layers import  Conv2D,MaxPooling2D,Dense,Flatten,Dropout
from keras.layers.normalization import BatchNormalization print("Loaded
required libraries...")
fpath = "../input/plantdisease/PlantVillage/" random_seed

= 111

categories = os.listdir(fpath)

print("List of categories = ",categories,"\n\nNo. of categories = ", len(categories))
def load_images_and_labels(categories): img_lst=[]
    labels=[]
for index, category in enumerate(categories):
            for image_name in os.listdir(fpath+"/"+category)[:300]: file_ext =image_name.split(".")[-1]
             if (file_ext.lower() == "jpg") or (file_ext.lower() == "jpeg"):
                #print(f"\nCategory = {category}, Image name ={image_name}")

                 img = cv2.imread(fpath+"/"+category+"/"+image_name) img
                =cv2.cvtColor(img, cv2.COLOR_BGR2RGB) img_array =
                Image.fromarray(img, 'RGB')
```

```
                #resize image to 227 x 227 because the input image resolution for
                AlexNet is 227 x 227
                resized_img = img_array.resize((227, 227))
                 img_lst.append(np.array(resized img))
                 labels.append(index)
     return img_lst, labels

images, labels = load_images_and_labels(categories)

print("No. of images loaded = ",len(images),"\nNo. of labels loaded = ",len(labels))
print(type(images),type(labels)) images =
np.array(images) labels = np.array(labels)

print("Images shape = ",images.shape,"\nLabels shape = ",labels.shape)

print(type(images),type(labels))

def display_rand_images(images, labels): plt.figure(1 ,
    figsize = (19 , 10))
    n = 0

    for i in range(9): n +=
        1
        r = np.random.randint(0 , images.shape[0] , 1) plt.subplot(3 ,
         3 , n)
        plt.subplots_adjust(hspace = 0.3 , wspace = 0.3)
        plt.imshow(images[r[0]])
        plt.title('Plant label : {}'.format(labels[r[0]])) plt.xticks([])
        plt.yticks([])
         plt.show()
display_rand_images(images, labels)
#1-step in data shuffling

#get equally spaced numbers in a given range n =
np.arange(images.shape[0])
print("'n' values before shuffling = ",n) #shuffle all the
equally spaced values in list 'n'
np.random.seed(random_seed) np.random.shuffle(n)
```

```
  print("\n'n' values after shuffling = ",n)
  #2-step in data shuffling
 #shuffle images and corresponding labels data in both the lists images = images[n]labels = labels[n]
print("Images shape after shuffling = ",images.shape,"\nLabels shape after shuffling = ",labels.shape)

display_rand_images(images, labels)

x_train, x_test, y_train, y_test = train_test_split(images, labels, test_size = 0.2,
random_state = random_seed)

print("x_train shape = ",x_train.shape)
print("y_train shape = ",y_train.shape)
print("\nx_test shape = ",x_test.shape) print("y_test
shape = ",y_test.shape)
display_rand_images(x_train, y_train)
model=Sequential()
#1 conv layer

model.add(Conv2D(filters=96,kernel_size=(11,11),strides=(4,4),padd
ing="valid",activation="relu",input_shape=(227,227,3)))

#1 max pool layer model.add(MaxPooling2D(pool_size=(3,3),strides=(2,2)))
model.add(BatchNormalization())
#2 conv layer

model.add(Conv2D(filters=256,kernel_size=(5,5),strides=(1,1),paddi
ng="valid",activation="relu"))

#2 max pool layer model.add(MaxPooling2D(pool_size=(3,3),strides=(2,2)))
model.add(BatchNormalization())
#3 conv layer

model.add(Conv2D(filters=384,kernel_size=(3,3),strides=(1,1),paddi
ng="valid",activation="relu"))

#3 max pool layer model.add(MaxPooling2D(pool_size=(3,3),strides=(2,2)))
model.add(BatchNormalization())
```

```
#4 conv layer

model.add(Conv2D(filters=384,kernel_size=(3,3),strides=(1,1),paddi
ng="valid",activation="relu"))

#5 conv layer

model.add(Conv2D(filters=256,kernel_size=(3,3),strides=(1,1),paddi
ng="valid",activation="relu"))


model.add(Flatten()) #1
dense layer
model.add(Dense(4096,input_shape=(227,227,3),activation="relu")) model.add(Dropout(0.4))
model.add(BatchNormalization()) #2 dense
layer
model.add(Dense(4096,activation="relu"))
(model.add(Dropout(0.4))
model.add(BatchNormalization())


#3 denselayer

model.add(Dense(1000,activation="relu"))
model.add(Dropout(0.4)) model.add(BatchNormalization))


#output layer model.add(Dense(20,activation="softmax"))
model.summary() model.compile(optimizer="adam",
loss="sparse_categorical_crossentropy", metrics=["accuracy"]) model.fit(x_train,
y_train, epochs=100)
loss, accuracy = model.evaluate(x_test, y_test)
print(loss,accuracy)
pred = model.predict(x_test)
pred.shape
plt.figure(1 , figsize = (19 , 10)) n = 0
for i in range(9):
 n +=1

 r = np.random.randint( 0, x_test.shape[0], 1)
```

```
st.shape[0], 1)

plt.subplot(3, 3, n)

plt.subplots_adjust(hspace = 0.3, wspace = 0.3)

plt.imshow(x_test[r[0]])

    plt.title('Actual = {}, Predicted =
{}'.format(y_test[r[0]] ,
y_test[r[0]]*pred[r[0]][y_test[r[0]]]) )

    plt.xticks([]) ,
    plt.yticks([])plt.show()
# save model in JSON
format model_json =
model.to_json()
json_file = open("../working/model1.json",
"w")json_file.write(model_json)
print("Model saved in JSON format!")

# save training weights in h5 file
model.save_weights("../working/model1.h5") print("\nModel
weights saved!")

# save model in JSON
format model_json =
model.to_json()
json_file = open("../working/model1.json",
"w")json_file.write(model_json)
print("Model saved in JSON format!")

# save training weights in h5 file
model.save_weights("../working/model1.h5")
print("\nModel weights saved”)
```

CHAPTER 7

TESTING

7.1 TESTING TECHNIQUES:

Testing is a process of executing a program with the intent of finding an error. A goodtest case is one that has a high probability of finding an as-yet – undiscovered error. A successful test is one that uncovers an as-yet- undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that as a probability of finding an yet undiscovered error. A successful test is one that uncovers an yet undiscovered error. Any engineering product can be tested in one of the two ways:

- WHITE BOX TESTING.
- BLACK BOX TESTING.

WHITE BOX TESTING:

This testing is also called as Glass box testing. In this testing, by knowing the specific functions that a product has been design to perform test can be conducted that demonstrate each function is fully operational at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis path testing:

- Flow graph notation.
- Kilometric complexity.
- Deriving test cases.
- Graph matrices Control.

BLACK BOX TESTING

In this testing by knowing the internal operation of a product, test can be conducted to ensure that "all gears mesh", that is the internal operation performs according to specification and allinternal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are

- Create test plans. Create prioritized test plans for black box testing.
- Test the external interfaces.
- Perform load testing.
- Perform stress testing.
- Perform security testing.
- Perform globalization testing.

SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned in advance and conducted systematically. For this reason a template for software testing a set of steps into which we can place specific test case design methods should be strategy should have the following characteristics:

- Testing proceeds in an outward manner.
- Testing techniques used during different phases of software development are different.
- Testing is conducted by the software developer and by an ITG.
- Testing and debugging should not be used synonymously.

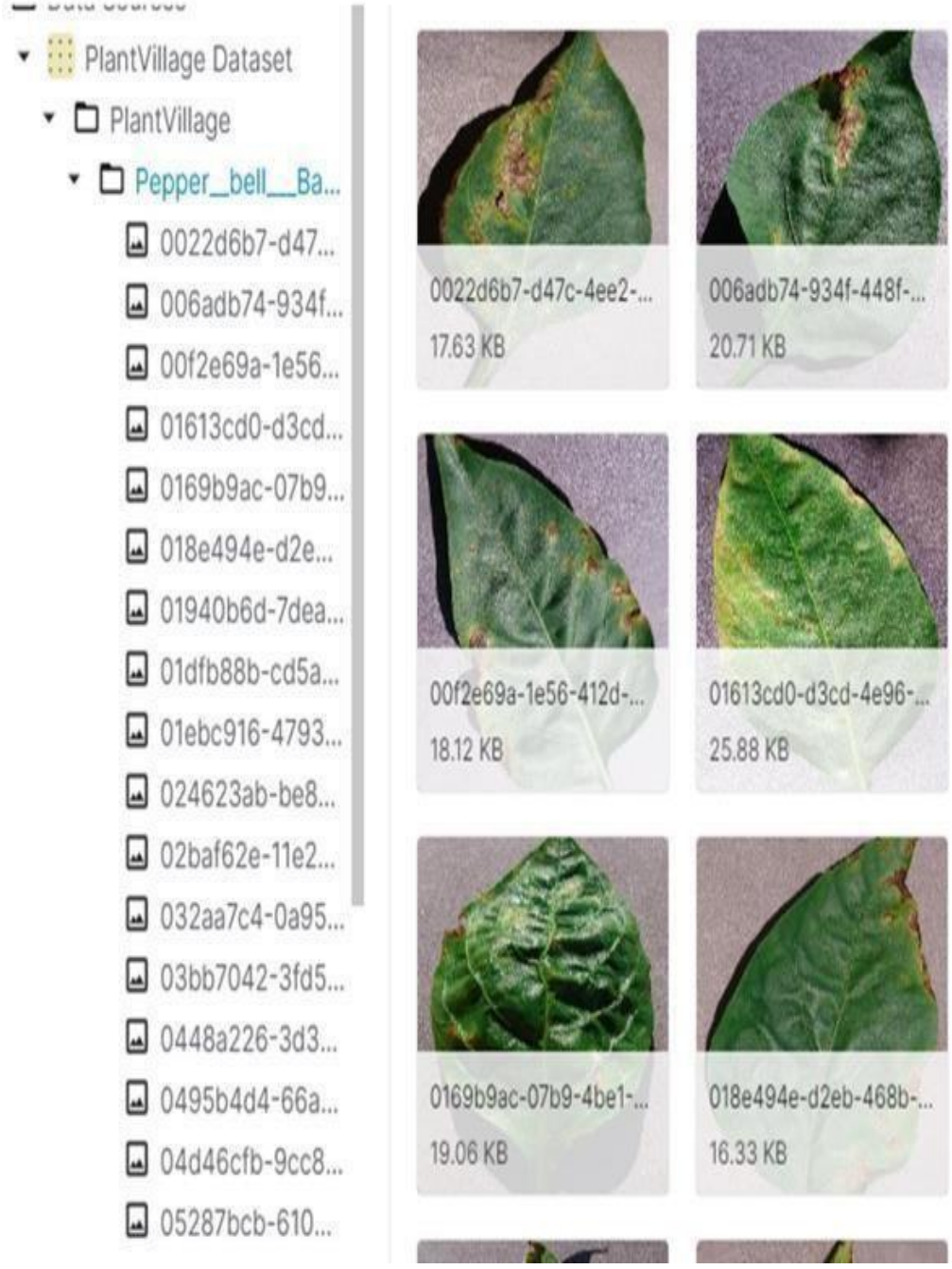
CHAPTER 8

CONCLUSION

*P*lant diseases were studied. It consist of 38 Categories of plant disease were examined. The VGG 16 model is constructed by using deep learning theory and convolution neural network technology. Analyses show that the model can successfully recognize the informational collection, Experiments show that the model can effectively identify the data set, and the overall recognition accuracy is as high as 95%. It can be effectively applied to the identification and detection of plant diseases.

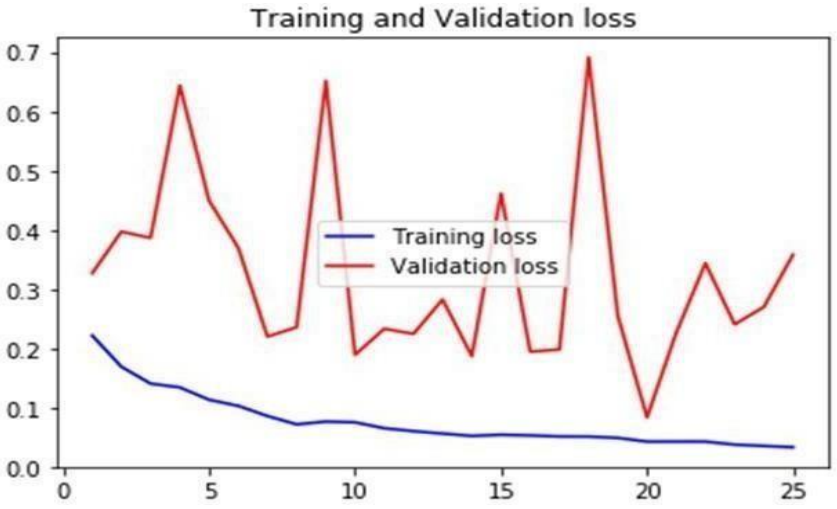
CHAPTER 9

SCREENSHOT

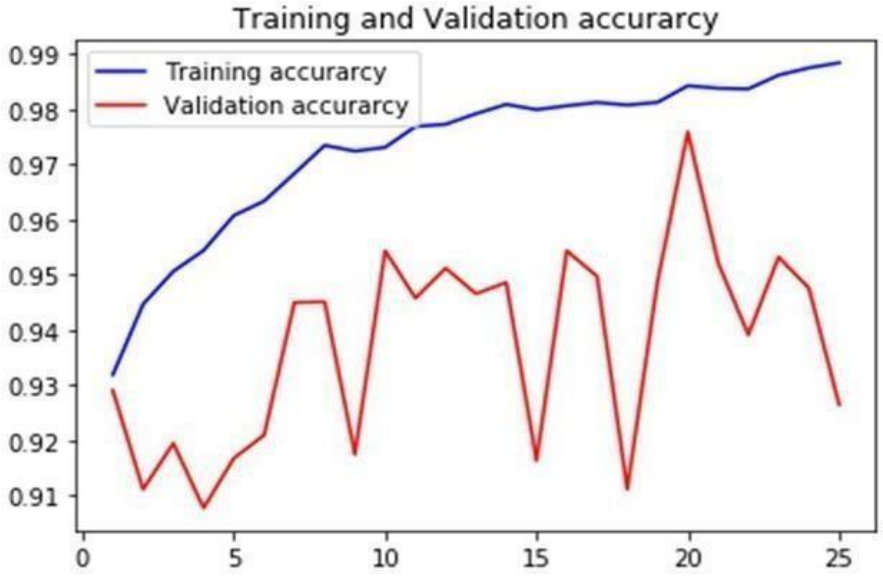


A1.1 Plant village Dataset

A1.2 Data Source



A1.3 Training and validation loss



A1.4 Training and validation accuracy

REFERENCE

[1] K. Padmavathi, and K. Thangadurai, "Implementation of RGB and Gray scale images in plant leaves disease detection –comparative study," Indian J. of Sci. and Tech., vol. 9, pp. 1- 6, Feb. 2016.

[2] Kiran R. Gavhale, and U. Gawande, "An Overview of the Research on Plant Leaves International Journal of Pure and Applied Mathematics Special Issue 882 Disease detection using Image Processing Techniques

[3] Y. Q. Xia, Y. Li, and C. Li, "Intelligent Diagnose System of Wheat Diseases Based on Android Phone," J. of Infor. & Compu. Sci., vol. 12, pp. 6845-6852, Dec. 2015.

[4] Wenjiang Huang, Qingsong Guan, JuhuaLuo, Jingcheng Zhang, Jinling Zhao, Dong Liang, Linsheng Huang, and Dongyan Zhang, "New Optimized Spectral Indices for Identifying and Monitoring Winter Wheat Diseases", IEEE journal of selected topics in applied earth observation and remote sensing,Vol. 7, No. 6, June 2014

[5] Monica Jhuria, Ashwani Kumar, and RushikeshBorse, "Image Processing For Smart Farming: Detection Of Disease And Fruit Grading", Proceedings of the 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013)

[6] Zulkifli Bin Husin, Abdul Hallis Bin Abdul Aziz, Ali Yeon Bin MdShakaffRohaniBinti S Mohamed Farook, "Feasibility Study on Plant Chili Disease Detection Using Image Processing Techniques", 2012Third International Conference on Intelligent Systems Modelling and Simulation.

[7] Mrunalini R. Badnakhe, Prashant R. Deshmukh, "Infected Leaf Analysis and Comparison by Otsu Threshold and k-Means Clustering", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 3, March 2012.

[8] H. Al-Hiary, S. Bani-Ahmad, M. Reyalat, M. Braik and Z. ALRahamneh, "Fast and Accurate Detection and Classification of Plant Diseases", International Journal of Computer Applications (0975 – 8887) Volume 17– No.1, March 2011.