LEAF DISEASE DETECTION USING DEEP LEARNING

BY

MADDUKURI NIVAS (19BCE1010)

KOVVURI UDAYU SURYA DEVESWAR REDDY (19BCE1253)

A project report submitted to

Prof. ILAKIYASELVAN. N

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

in partial fulfilment of the requirements for the course of

CSE4028 – OBJECT ORIENTED SOFTWARE DEVELOPMENT

In

B.Tech. COMPUTER SCIENCE ENGINEERING

VIT-CHENNAI

Vandalur-Kelambakkam Road

Chennai-600127

INDEX

1. Acknowledgement

We wish to express our sincere thanks and deep sense of gratitude to our project guide, prof. Ilakiyaselvan N. School of Computing Science and Engineering, for her consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

MADDUKURI NIVAS **19BCE1010**

KOVVURI UDAY SURYA DEVESWAR REDDY **19BCE1253**

2. ABSTRACT

Leaf Disease Detection can be a helpful aspect on huge area of field of crops. In our country most of the people in many countries are dependent on the agricultural productivity. This project is also conveyed us by knowing artificial intelligence is helpful in solving real life problems. This project gives us the way to find the disease caused to the leaf by using CNN algorithm and another some classification algorithms and methods. Detection of plant disease through some automatic technique is beneficial as it requires a large amount of work of monitoring in big farm of crops, and at very early stage itself it detects symptoms of diseases means where they appear on plant leaves. In order to detect the disease some steps are to be followed using image processing. Detection of leaf disease will help to farmers and as well as society in Improving the quality and production of agricultural crops. This also makes the development of artificial intelligence technology in solving real life problems in model of machine level.

3. Keywords

This document uses the following conventions.

CNN – Convolutional Neural Networks
Alexnet – A type of cnn algorithm
Mobile Net – A type of cnn algorithm
VGG16 – A type of cnn algorithm

4. INTRODUCTION

Agriculture is just not helpful for human feeding or energy it is much more like energy and global warming. Most of the rural area doesn't have awareness on the diseases of the planning. So, by this detective system we can help few farmers. In some of the area which are not yet developed, farmers are facing problems with the leaf diseases. To make a solution for this we are trying to build a detective system for leaf disease. To identify the disease, we need an image of the leaf which is affected.

For the purpose of detecting the diseased leaf, to measure the affected area by the disease, colour of the affected area and to identify the object(leaf), image processing is used. In this we are also using neural networks to do the methodology. We can analyse the image of disease leaves by using computer image processing technology and extract the features of disease spot according to colour, texture and other characteristics from a quantitative point of view. At an equivalent time, in some countries, farmers don't have correct facilities or maybe concept that they'll contact specialists. Because of that consulting specialists even price high still as time overwhelming too. In such condition, the advised technique proves to be helpful in watching massive fields of crops. And automatic detection of the diseases by simply seeing the symptoms on the plant leaves makes it easier still as cheaper.

ADVANTAGES OF THE SYSTEM:

- By doing this project we can introduce a new way of agricultural developing technology. Large number of variables can be processed at the same time.
- While doing this project the algorithms which are used will optimizes efficiently and continues or discrete.
- Our project solution includes high processing speed and high classification accuracy.
- After training of dataset our machine level product is useful to distinguish various types of leaf diseases.

- Leaf disease detection system will more useful to farmers in the way that it increases the productivity of the crop.

5. FUNCTIONAL REQUIREMENTS

Functional requirements define the basic system behaviour. Essentially, they are what the system does or must not do, and can be thought of in terms of how the system responds to inputs. Functional requirements usually define if/then behaviours and include calculations, data input, and business processes. Functional requirements are the primary way that the customer communicates their needs to the team. They keep everyone on the project team going in the same direction. Without an agreed functional requirements document to clearly define the scope, the end product is likely to miss the mark.

5.1. SOFTWARE REQUIREMENTS:

Language : Python

OS : Windows 10(64 bit)

IDE : Anaconda Navigator (Jupiter Notebook)

5.2. HARDWARE REQUIREMENTS

Processor : Above 1.5GHZ

Hard Disk : 80GB

RAM : 2GM

6. INTERFACE REQUIREMENTS

Interface requirements refers to the need of the software to work efficiently the requirements are further classified into certain topics. In our project there is a simple process which can explained as a modules

6.1. User Interface:

Basically, our project is leaf disease detection by image processing, to identify the disease caused to the leaf all want to do is the particular user needs to upload the image of the leaf which is caused to the leaf, then it will train that particular image using ml-based learning and it will compare and train with the dataset and then it finds the accuracy of the result by using cnn algorithms, after completion of the finding the disease, it will print out the disease caused to the leaf. We also provided a step-by-step process in the website; it helps to understand how to upload the image for the people who are not aware of with the software system.

6.2. Backend Development:

To develop this project, we used Jupiter notebook to implement our project and the process of our work is at first, we had gathered the dataset from Kaggle.com which is required to train and test the images. Dataset plays a major role in these types of projects because to find the disease of particular leaf which is to be asked by the user to identify that, we should train the required dataset. A training dataset is a dataset of examples used during the learning process and is used to fit the parameters.

7. NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are requirements which specify criteria that can be used to judge the operation of a system, rather than specific behaviours. This should be contrasted with functional requirements that specify specific behaviour or functions. Typical non-functional requirements are reliability, scalability, and cost. Non-functional requirements are often called the ileitis of a system. Other terms for non-functional requirements are "constraints", "quality attributes" and "quality of service requirements".

Reliability:

If any exceptions occur during the execution of the software, it should be caught and thereby prevent the system from crashing.

Scalability:

The system should be developed in such a way that new modules and functionalities can be added, thereby facilitating system evolution.

Cost:

The cost should be low because a free availability of software package.

8. SYSTEM DESIGN

8.1. Application Development

The web application for the agriculture was developed using the Jupiter notebook. Jupiter notebook is the official Integrated Development Environment (IDE) for web platform development. The development of the website had various stages such as creation of server, training images in the dataset, accessing images from dataset, updating information into the dataset and extracting message from the dataset. The procedure first was started by creating a server by creating account in anaconda navigator and a dataset and consequently a website for uploading images should create. Later a python was written for connecting to the server and the dataset to the web application.

8.2. Creating the model:

Creating the model from the data set in the process involved in the ml-based learning. First, import the libraries and then import the dataset by giving the path and then we should train the data that is we have to trained the images by generating the images using imagedatagenerator by using this we can resize the images then we will train all the images by adjusting the height and weight of the images. And then we will create the predictions on it by using the algorithms vgg16, resnet45, mobilenet these all algorithms are cnn algorithms.

8.3. Upload Image:

When the user opens the website so that the user can select image from the folder. On selection of the picture the picture gets displayed. On selection of right image and the image being displayed in the app the farmer can click on Upload image option. The image gets uploaded to the server and gets input in the model training. An ID is returned to the farmer with a message with the output result, that is the disease caused to the leaf.

8.4. Image Processing:

Digital image processing is the use of computer algorithms to perform image processing on digital images. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be model in the form of [multidimensional systems].

The following steps are followed for detecting disease in crop:

- Image Preprocessing
- Segmentation
- Feature Extraction
- Classification

8.5. Algorithms Used:

There are many algorithms which can be used to identify the disease of the leaf, in our project we used alexnet, vgg16, mobilenet algorithms.

Alexnet cnn algorithm:

AlexNet is a classic convolutional neural network architecture. It consists of convolutions, max pooling and dense layers as the basic building blocks. Grouped convolutions are used in order to fit the model across two GPUs.

VGG16 cnn algorithm:

VGG16 is a pretrained convolutional neural network model achieved a 92.7% accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes.

Mobilenet cnn algorithm:

Mobile Net uses depthwise separable convolutions. It significantly reduces the number of parameters when compared to the network with regular convolutions with the same depth in the nets.

9. SDLC MODEL

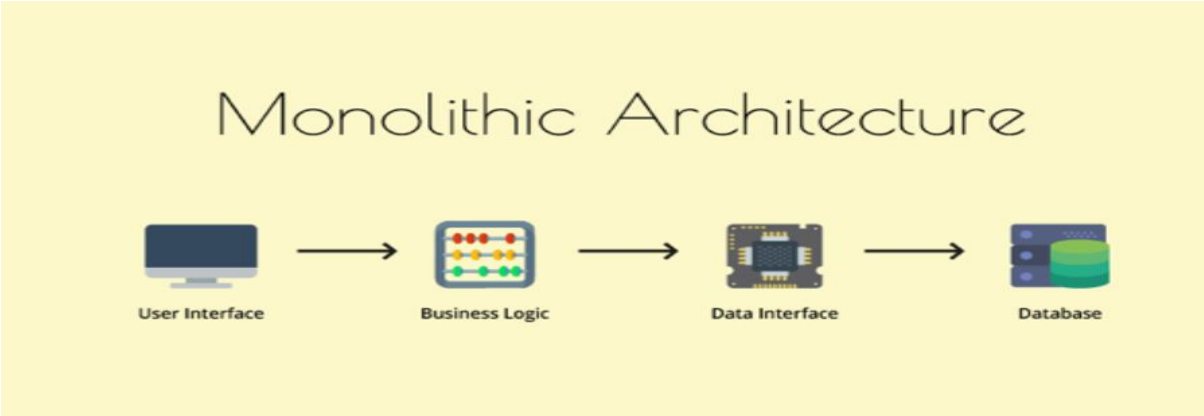
Agile development model is a type of Incremental model. Software is developed in incremental, rapid cycles. This results in small incremental releases with each release building on previous functionality. Agile methods can be used for Machine Learning projects. Many standard practices of software development continue to develop for AI development.
Reproducibility is a critical component of software systems that faces challenges for AI Systems. ML systems have dependencies not only in code but potentially in data as well, track them carefully.ML system development is still a work in progress, new needs are being surfaced but best practices and tools are still needed.

The agile model was perfect to meet our requirements. There was customer and developer interaction which helped us to understand and implement customer need and make it more user friendly. With the help of customer feedback, we were able to eliminate issues and improvise our project according to customer needs. Our team developed each module and after the completion of every module. We follow proper testing and validation. Then customers were allowed to use our programs and give us feedback. Then we work on the specifications suggested by customers. We again follow the same no more changes can be made. In this way we followed the iterative model. Then we follow incremental model approach and proceeds to the next

model. After the development of model, we perform testing and validation. Then we move on to customer feedback and this cycle continues until our project is completed. In this way we followed and implemented agile model in our project.

10. SOFTWARE ARCHITECTURE

Monolithic applications are designed to handle multiple related tasks. They're typically complex applications that encompass several tightly coupled functions. Monolithic software is designed to be self-contained; components of the program are interconnected and interdependent rather than loosely coupled as is the case with modular software programs. In a tightly-coupled architecture, each component and its associated components must be present in order for code to be executed or compiled.



This approach was the traditional way of designing a system. Typically, a monolithic software application consists of many different modules working together to deliver many different functionalities. In normal, most of the ML-based large systems or projects can be suitable for this monolithic architecture. You can do end-to-end testing simply by launching the application. As, our project includes the data set and the python-based implementation, the data set is to be trained and tested, as they are many modules in this system which required to work as together this architecture can be used.

11. DESIGN PATTERN

Design patterns are implemented to architect a software design solution. Implementing industrial best practices for object-oriented programming and code implementation require design patterns at a higher-level as an abstract solution.

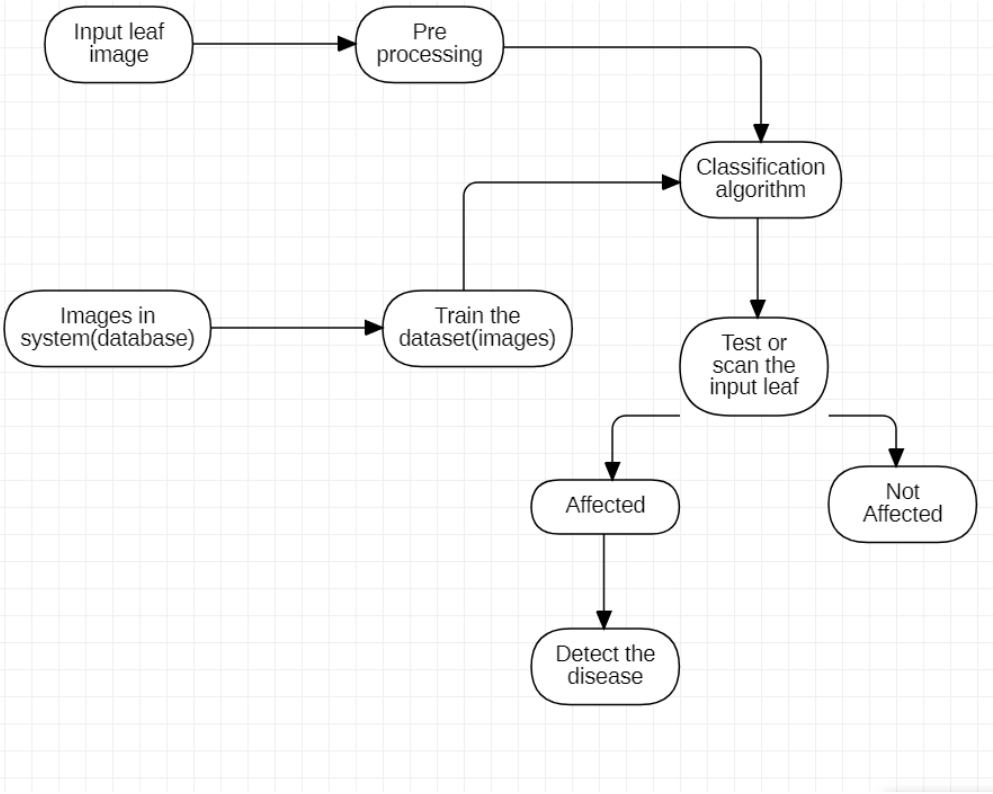
Adapter pattern works as a bridge between two incompatible interfaces. This type of design pattern comes under structural pattern as this pattern combines the capability of two independent interfaces. This pattern involves a single class which is responsible to join functionalities of independent or incompatible interfaces.

It will be to access the information between two different environments with some functionalities.

This pattern involves a single class which is responsible to join functionalities of independent or incompatible interfaces. A real-life example could be a case of card reader which acts as an adapter between memory card and a laptop. You plug in the memory card into card reader and card reader into the laptop so that memory card can be read via laptop.

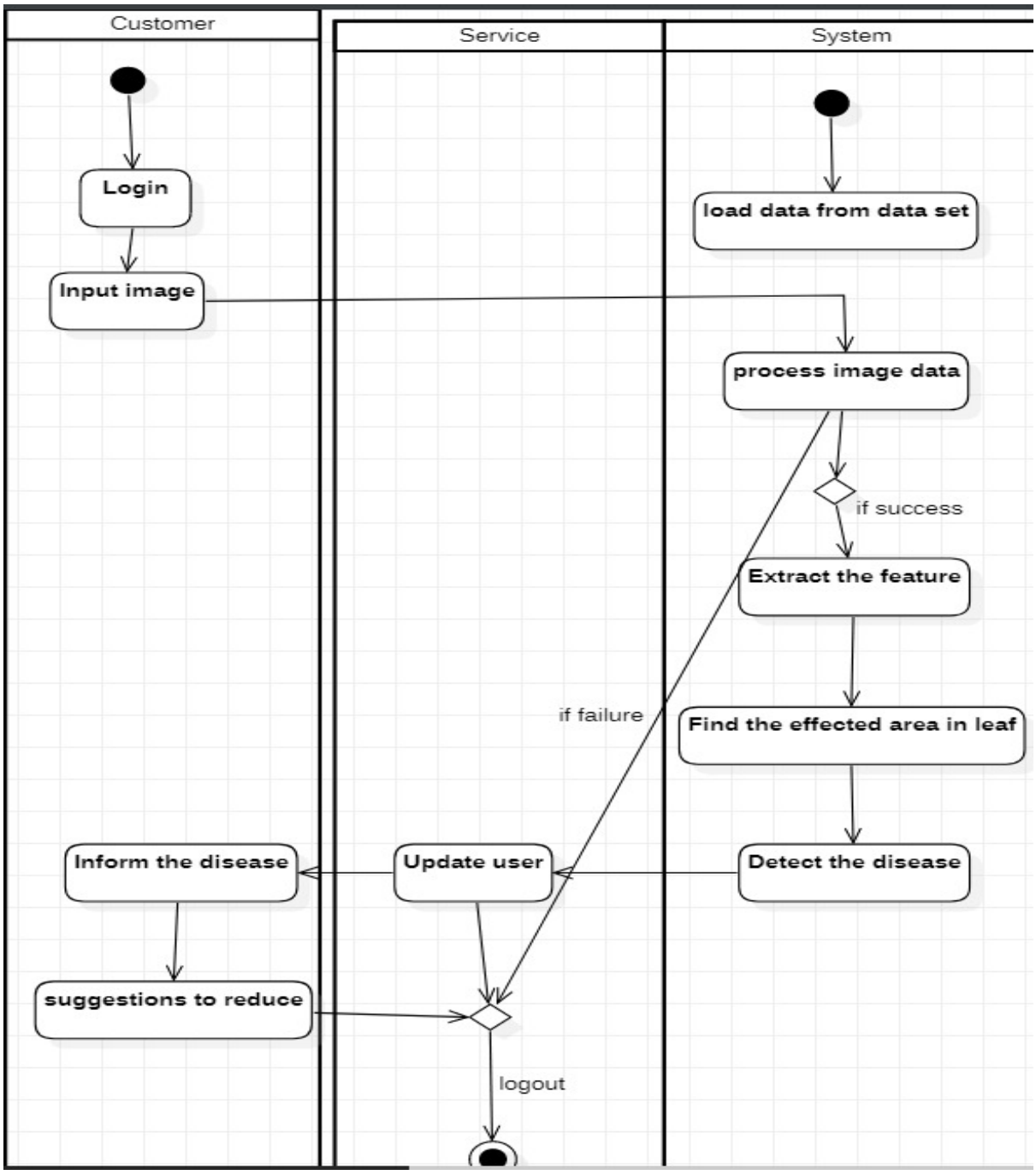
12. UML DIAGRAMS

12.1 Work Flow Diagram:



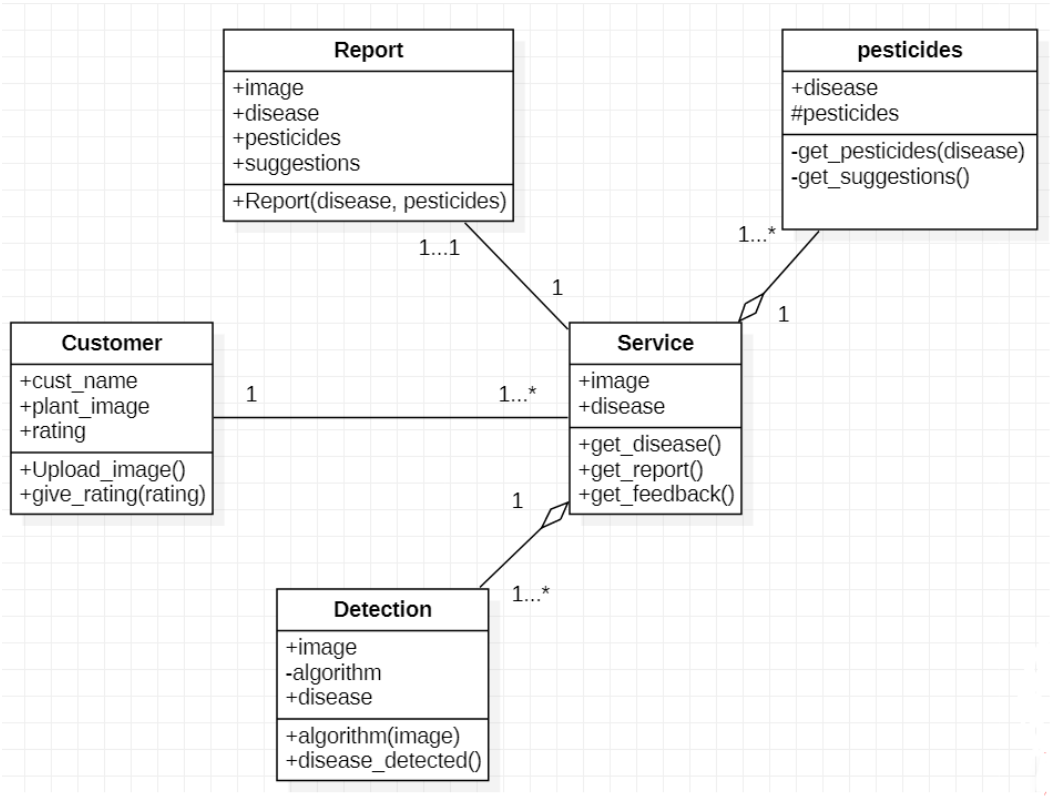
This diagram explains the process or the steps to be followed to develop our project. It is a step-by-step, linear representation of a business process from start to finish. It shows how individual tasks, actions, or resources flow between different people or groups. It also shows what needs to be done in order for that task to be finished.

12.2. Activity Diagram



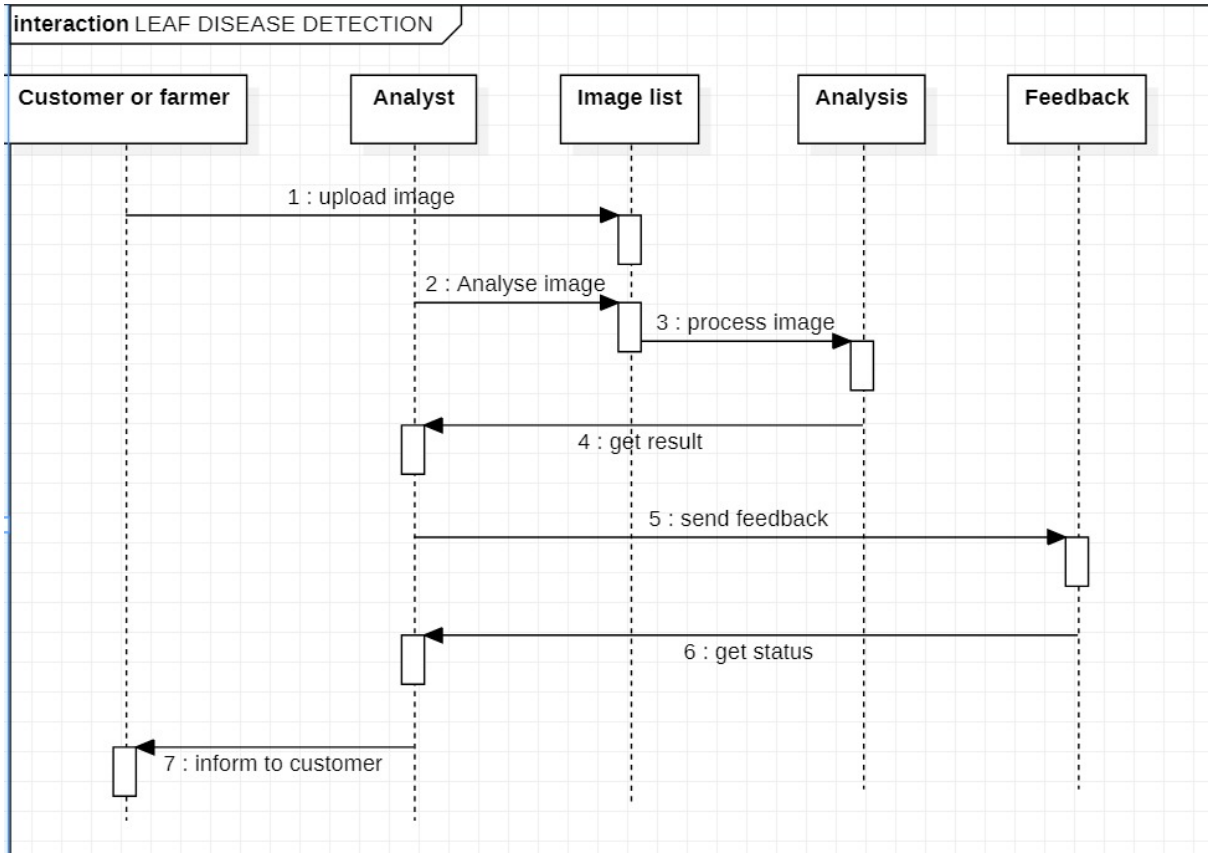
Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation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

12.3. Class Diagram



The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling, translating the models into programming code. Class diagrams can also be used for data modeling.

12.4. Sequence Diagram:



A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process.

13. Results

1. ALEXNET ALGORITHM ACCURACY:

```
Epoch 1/2
549/549 [==============================] - 2870s 5s/step - los
s: 0.0931 - acc: 0.9688 - val_loss: 0.1075 - val_acc: 0.9632

Epoch 00001: val_acc improved from -inf to 0.96316, saving mod
el to best_weights_9.hdf5
Epoch 2/2
549/549 [==============================] - 2769s 5s/step - los
s: 0.0927 - acc: 0.9688 - val_loss: 0.1064 - val_acc: 0.9634
```

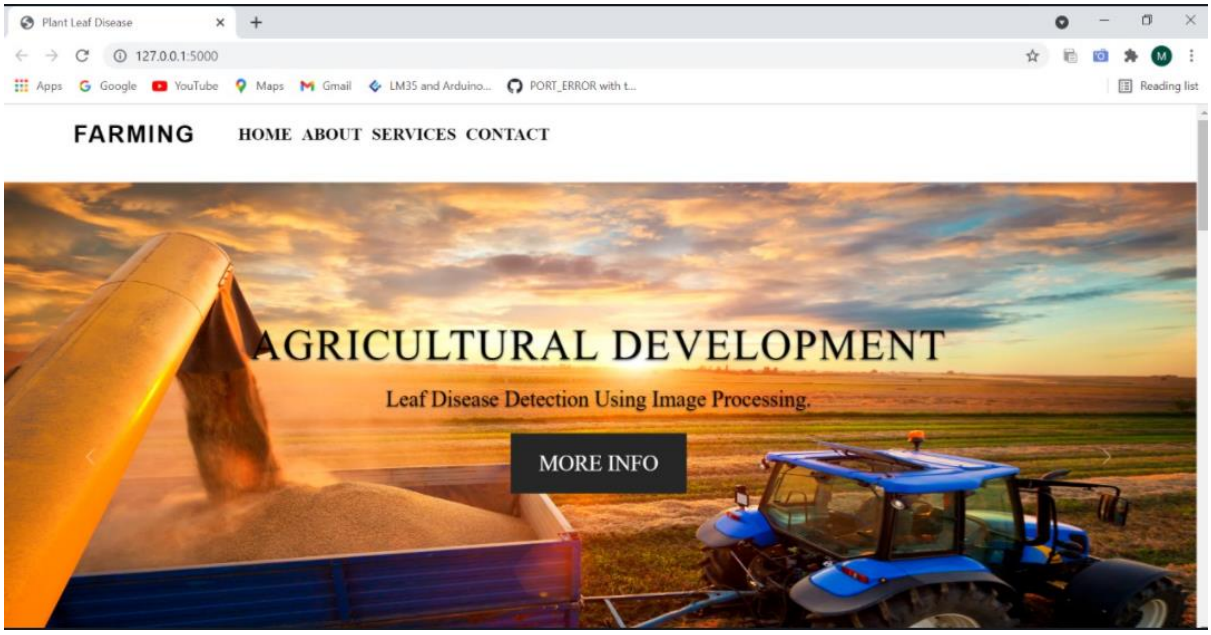
2. VGG16 ALGORITHM ACCURACY:

```
150/150 [==============================] - 71s 470ms/step - los
s: 1.1481 - categorical_accuracy: 0.7179 - val_loss: 0.9344 - va
l_categorical_accuracy: 0.7872
Epoch 23/25
150/150 [==============================] - 71s 474ms/step - los
s: 1.1090 - categorical_accuracy: 0.7275 - val_loss: 0.9221 - va
l_categorical_accuracy: 0.7847
Epoch 24/25
150/150 [==============================] - 71s 472ms/step - los
s: 1.0958 - categorical_accuracy: 0.7252 - val_loss: 0.9160 - va
l_categorical_accuracy: 0.7903
Epoch 25/25
150/150 [==============================] - 71s 474ms/step - los
s: 1.1061 - categorical_accuracy: 0.7144 - val_loss: 0.8749 - va
l_categorical_accuracy: 0.7894
```

MOBILE NET ALGORITHM ACCURACY:

```
Epoch 10/18
300/300 [==============================] - 496s 2s/step - loss: 2.8021 - categorical_accuracy: 0.8986 - val_loss: 2.7814 - v
al_categorical_accuracy: 0.9162
Epoch 11/18
300/300 [==============================] - 497s 2s/step - loss: 2.7978 - categorical_accuracy: 0.9038 - val_loss: 2.7809 - v
al_categorical_accuracy: 0.9134
Epoch 12/18
300/300 [==============================] - 496s 2s/step - loss: 2.7876 - categorical_accuracy: 0.9126 - val_loss: 2.7725 - v
al_categorical_accuracy: 0.9217
Epoch 13/18
300/300 [==============================] - 490s 2s/step - loss: 2.7890 - categorical_accuracy: 0.9092 - val_loss: 2.7690 - v
al_categorical_accuracy: 0.9247
Epoch 14/18
300/300 [==============================] - 492s 2s/step - loss: 2.7855 - categorical_accuracy: 0.9128 - val_loss: 2.7708 - v
al_categorical_accuracy: 0.9220
Epoch 15/18
300/300 [==============================] - 492s 2s/step - loss: 2.7899 - categorical_accuracy: 0.9057 - val_loss: 2.7700 - v
al_categorical_accuracy: 0.9218
Epoch 16/18
300/300 [==============================] - 508s 2s/step - loss: 2.7800 - categorical_accuracy: 0.9185 - val_loss: 2.7659 - v
al_categorical_accuracy: 0.9255
Epoch 17/18
300/300 [==============================] - 506s 2s/step - loss: 2.7771 - categorical_accuracy: 0.9190 - val_loss: 2.7695 - v
al_categorical_accuracy: 0.9227
Epoch 18/18
300/300 [==============================] - 492s 2s/step - loss: 2.7820 - categorical_accuracy: 0.9121 - val_loss: 2.7620 - v
al_categorical_accuracy: 0.9302
```

WEBSITE HOMEPAGE:



ABOUT PAGE:

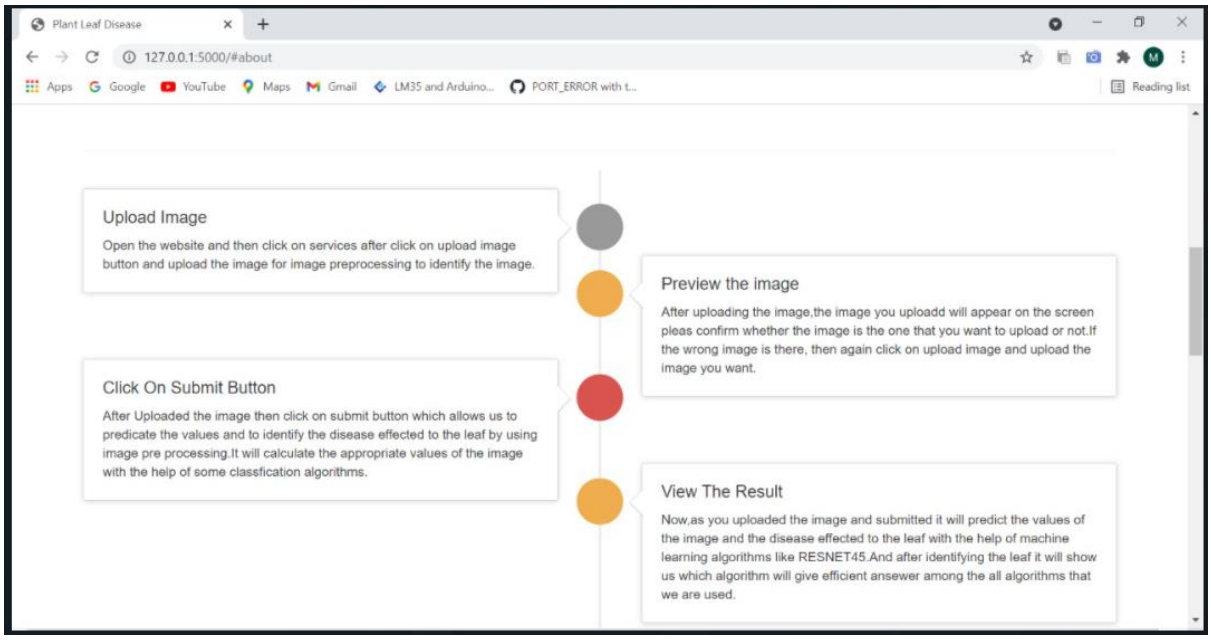
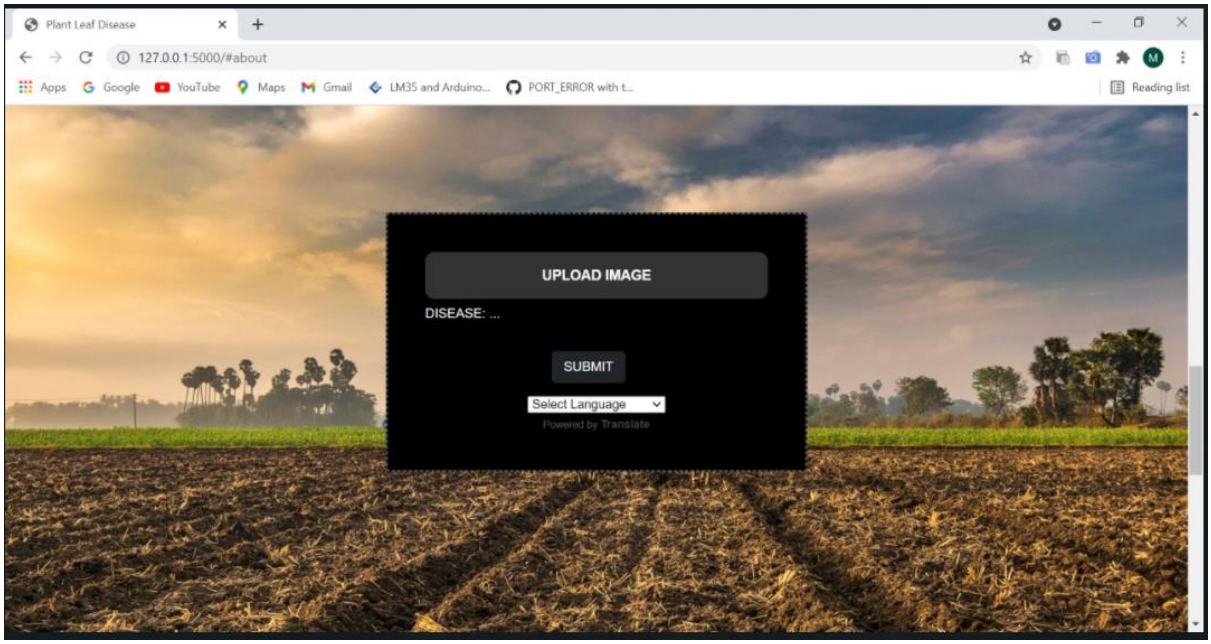
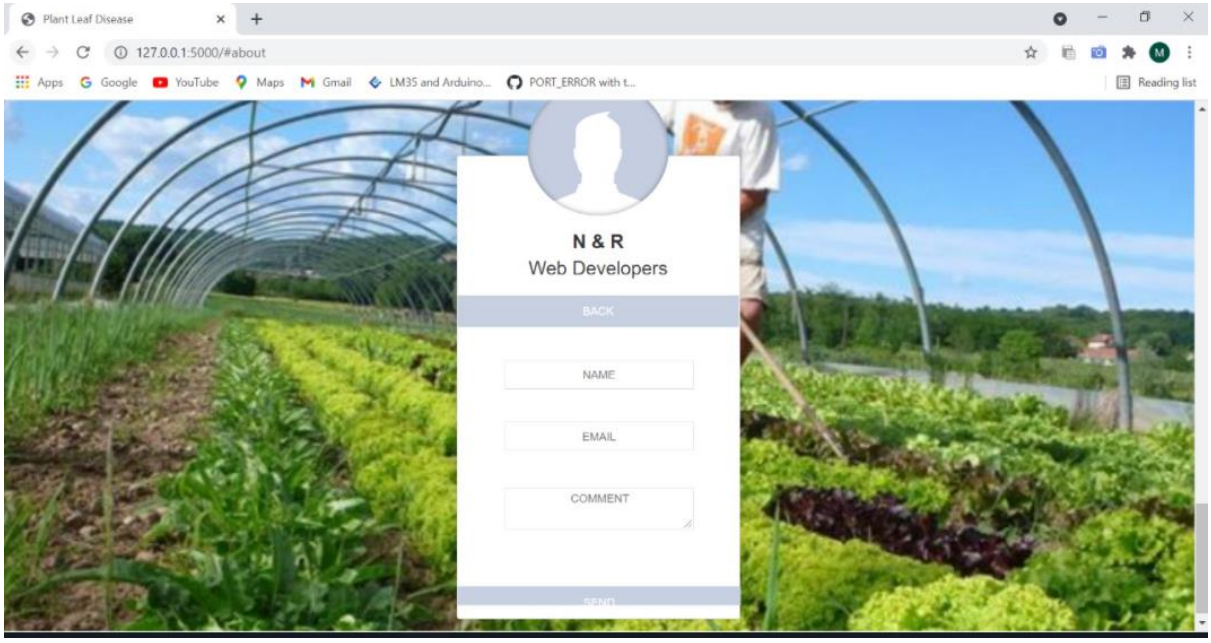


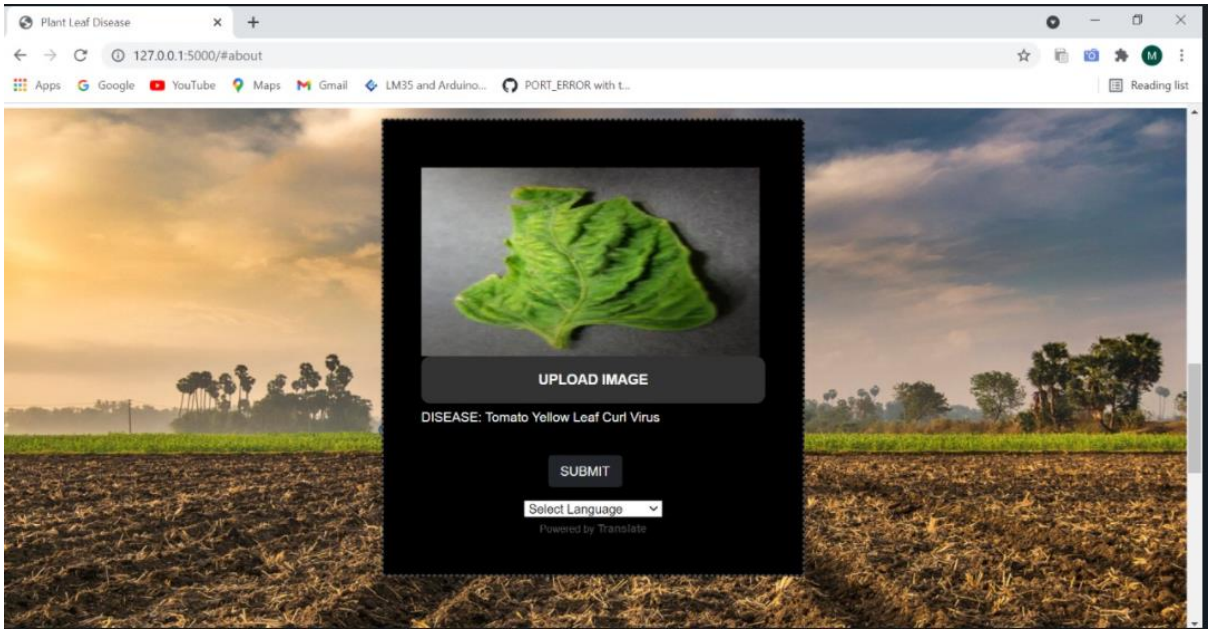
IMAGE UPLOAD OPTION:

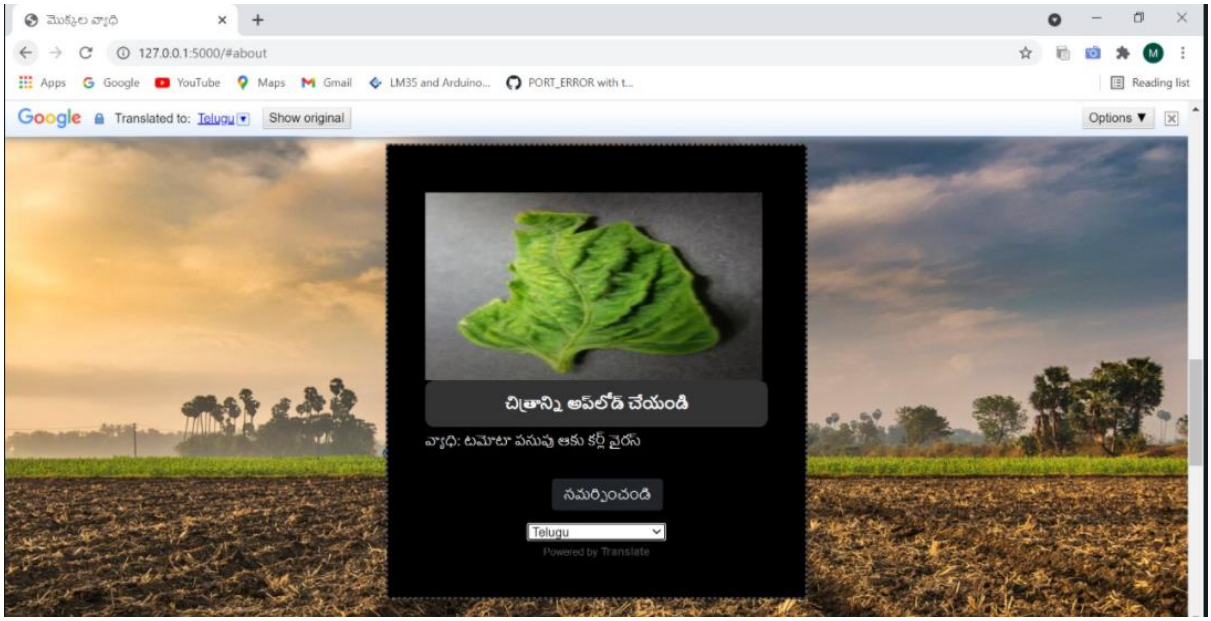


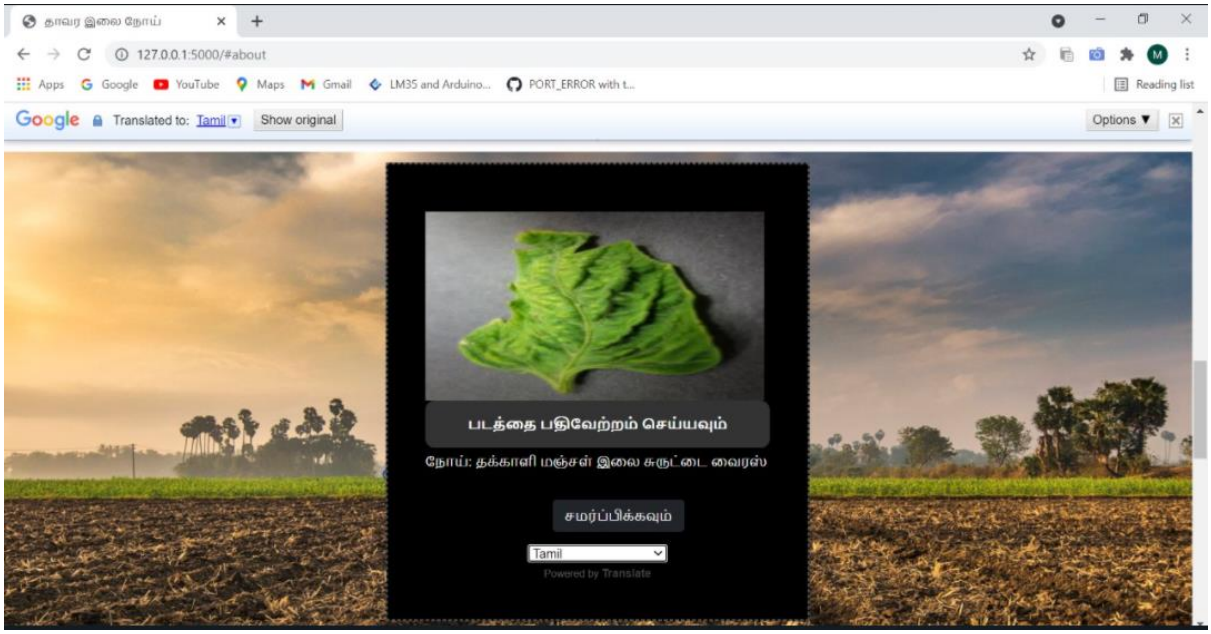
CONTACT PAGE:



OUTPUT (DISPLAYED DISEASE CAUSED TO LEAF):







13. CONTRIBUTIONS

Maddukuri Nivas :- Alexnet, connections to the website

Kovvuri Uday Surya Deveswar Reddy: - Mobilenet, vgg16 algorithms

14. CONCLUSION

An application of detecting the plant diseases and providing the necessary suggestions for the disease has been implemented. Hence the proposed objective was implemented on three different types of crops namely Rice, Sugarcane and Cotton. The diseases specific to these plants were considered for testing of the algorithm. The experimental results indicate the proposed approach can recognize the diseases with a little computational effort. By this method, the plant diseases can be identified at the initial stage itself and the pest control tools can be used to solve pest problems while minimizing risks to people and the environment. In order to improve disease identification rate at various stages, the training samples can be increased with the optimal features given as input condition for disease identification and fertilization management of the crops. As a part of Future Enhancement, the complete process described in this project can be automated so that the result can be delivered in a very short time.

15. REFERENCES

[1] S. Arivazhagan, R. Newlin Shebiah*, S. Ananthi, S. Vishnu Varthini March 2013. Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. Agric Eng Int: CIGR Journal Vol. 15, No.1 211

[2] Al-Bashish, D., M. Braik, and S. Bani-Ahmad. 2011. Detection and classification of leaf diseases using K-means-based segmentation and neural networks-based classification. Information Technology Journal, 10(2): 267-275

[3] Al-Hiary, H., S. Bani-Ahmad, M. Reyalat, M. Braik, and Z. AlRahamneh. 2011. Fast and accurate detection and classification of plant diseases. International Journal of Computer Applications, 17(1): 31-38