KLE Society's

KLE Technological University



**A Capstone Project Report**

**On**

**Detection and Classification of Plant Leaf Diseases using Deep Learning Algorithm**

*submitted in partial fulfillment of the requirement for the degree of*

**Bachelor of Engineering in**

**Computer Science and Engineering**

**Submitted by**

**Pooja G Kamble 01FE16BCS140**

**Praveen Kundargi 01FE17BCS418**

**Saurabh Anbarasan 01FE17BCS505**

**Under the guidance of**

**Prof. Manjunath Gonal and Prof. Vijay. S .Biradar**

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

HUBLI–580 031 (India).

Academic year 2019-20

**1.INTRODUCTION**

Disease detection in plants plays an important role in the agriculture field, since having diseases in plants are quite natural. If proper care is not taken in this area then it can cause serious effects on plants due to which respective product quality, quantity or productivity is also affected. Plant diseases cause a periodic outbreak of diseases which leads to large-scale death. These problems need to be solved at the initial stage, to save life and money of people. Automatic detection of plant diseases is an important research topic as it may prove benefits in monitoring large fields of crops, and at a very early stage itself it detects the symptoms of diseases means when they appear on plant leaves.

This enables machine vision, that is, to provide image-based automatic inspection, process control. Comparatively, visual identification is labor intensive, less accurate and can be done only in small areas. The project involves the use of self-designed image processing algorithms and techniques designed using python to segment the disease from the leaf while using the concepts of deep learning to categorize the plant leaves as healthy or infected. By this method, the plant diseases can be identified at the initial stage itself and the pest and infection control tools can be used to solve pest problems while minimizing risks to people and the environment.

Deep neural networks have recently been successfully applied in many diverse domains as examples of end to end learning. Neural networks provide a mapping between an input—such as an image of a diseased plant—to an output—such as a crop-disease pair. The nodes in a neural network are mathematical functions that take numerical inputs from the incoming edge and provide a numerical output as an outgoing edge. Deep neural networks are simply mapping the input layer to the output layer over a series of stacked layers of nodes.

* 1. **Overview of the project**

Plant phenotyping refers to a quantitative description of the plant’s anatomical, ontogenetical, physiological and biochemical properties. Today, rapid developments are taking place in the field of non-destructive, image-analysis -based phenotyping that allow for a characterization of plant traits in high-throughput.

This project develops computer vision-based methods for doing plant phenotyping on data collected. In particular, we consider the problem of extracting phenotype values from a sequence of images of a Tomato plant and then classifying them on the basis of their diseases and later on their stress.

* 1. **Motivation**
* The detection of disease in plants plays a key role in the field of agriculture, because having diseases in crops are quite natural.
* If this is neglected then it can cause serious effects on plants due to which quality, productivity and yield of the plant is affected.
* Diseases in plants cause a periodic outbreak of diseases which leads to large-scale death. These problems need to be solved at the initial stage, to save life and money of people.
  1. **Objectives of the project**
* To detect affected area of the leaf.
* To identify the disease of a plant and thus increasing yield of the plant.
* To obtain the phenotype of a plant without damaging it.
  1. **Challenges**
* To detect complex background that cannot be easily separated from the region of interest.
* Deep learning algorithms require large amount of labeled data.
* To identify the disease of plant at an early stage.

**1.5 Problem Statement**

To predict and identify the disease of a plant using deep learning methods.

**2. LITERATUE SURVEY**

**3.Proposed System**

We propose the plant disease detection technique that is used to predict and identify the disease of a plant. In Section 3.1, we discuss the detail description of the proposed system and stages through which the leaf image is processed and classified.

**3.1 Description of proposed system with simple block diagram**

A flow diagram is a symbolic representation of a process. Each step in the process is represented by a different symbol and contains a short description of the process step. The flow chart symbols are linked together with arrows showing the process flow direction.

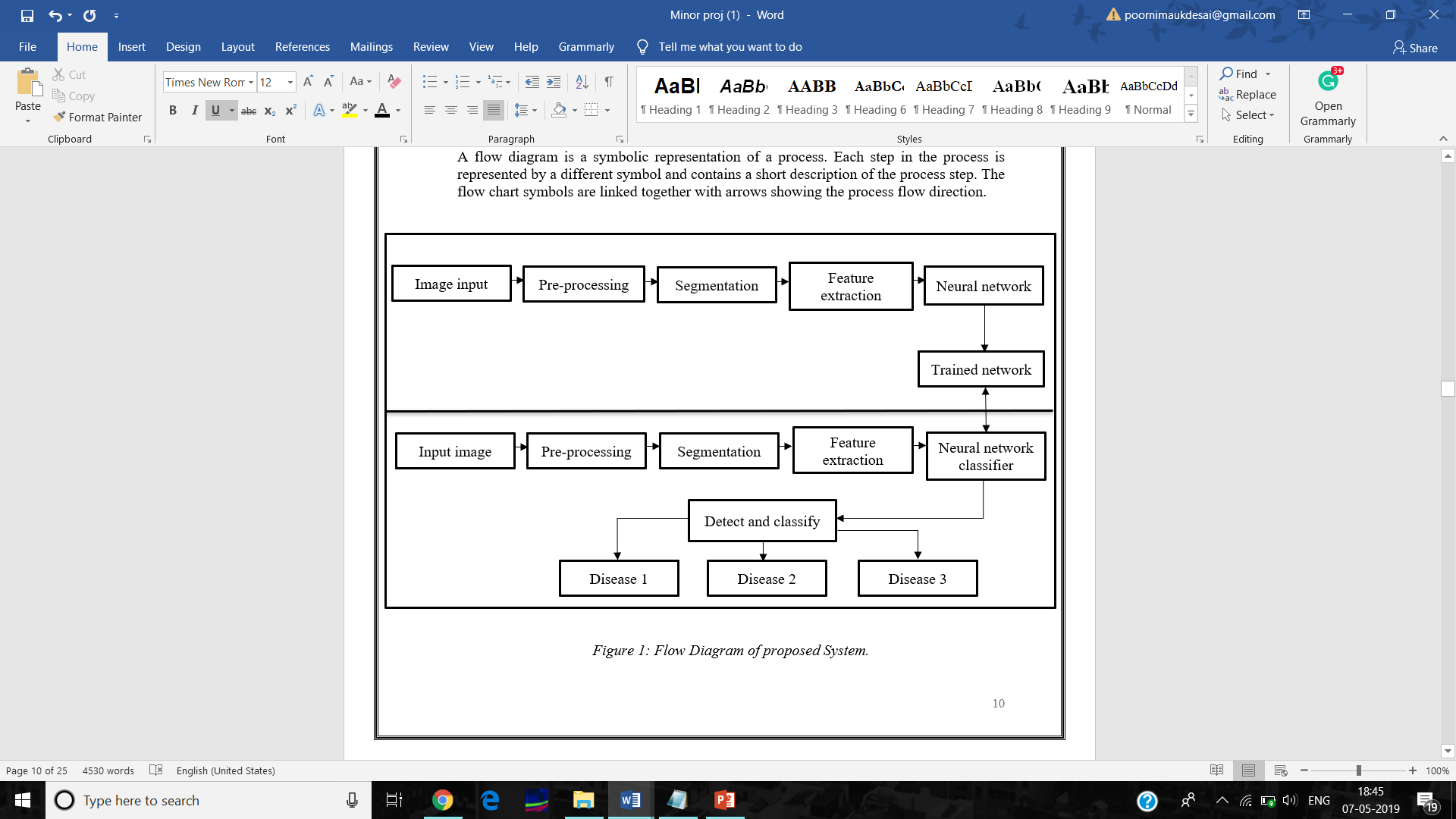


Figure 1: Flow Diagram of proposed System.

The above diagram gives the explanation of the design process of the proposed system.

1.Input Image:

The input is in the form of a image, which can be a healthy image or diseased image

2.Preprocessing:

Input Image (RGB) is converted into grayscale image. Filters are then applied to the image and then cropped to appropriate size.

3.Segmentation:

 The infected portion of the leaf is extracted by segmenting the diseased part with other similar coloured parts (say, a brown colour of a leaf that may look like the disease) which have been considered in the masked-out image, are filtered here. All further image processing is done over a region of interest (ROI) defined at this stage.

4.Feature Extraction:

Gray Level Co-Occurrence Matrix (GLCM) has proved to be a popular statistical method of extracting textural feature from images. Texture features extracted from this are namely homogeneity, correlation, energy, contrast.

In addition to this entropy of the image is also extracted. The entropy of an image is a measure of the degree of randomness in the image.

**3.2** **Target users**

* Farmers
* Agricultural Institutes

**3.3** **Advantages/applications of proposed system**

* Detection of disease at different stages of the plant.
* Plants can be relocated to a healthier environment if it is affected by abiotic stress.
* Treatment of plants can be done if the plant is affected by biotic stress.

**3.4 Scope of Work**

* Identification of disease of a single plant
* Plant used – solanum lycopersicum (tomato)
* Classification of biotic and abiotic stress

**4. Software Requirement Specification**

A **software requirements specification (SRS)** is a detailed description of a **software** system to be developed with its functional and non-functional **requirements**. Introduction of the Software Requirements Specification provides an overview of the entire SRS with purpose, scope, definitions, abbreviations, references and overview of the SRS. The aim of this document is to gather and analyse and give an in-depth insight of the complete Plant phenotyping by defining the problem statement in detail.

Software requirement specifications

• Operating system: Ubuntu server 16.04 or above.

• Python as programming language.

• Jupyter notebook used to write code.

**4.1 Requirement Specifications**

**4.1.1 Functional requirements:**

* System shall be able to detect the leaf from the captured image.
* System shall be able to eliminate image distortion.
* System shall be able to predict the disease.

**4.1.2 Non-functional requirements**

* Accuracy of the model should be greater than 70%.
* System shall require a minimum of 4 GB RAM.

**4.2 Acceptance test plan**

Acceptance testing is testing where a system is tested for acceptability. Images are fed to the system and different tests are performed to decide whether or not it satisfies all the requirements. Acceptance test pans are written based on the functional requirements defined for our project.

1. System shall be able to detect the leaf from image

|  |  |  |  |
| --- | --- | --- | --- |
| Test case id | Input description | Expected output | Actual output |
| 1 | System is able to detect leaf from the image | Leaf is detected |  |
| 2 | System is able to detect diseased portion of leaf | Diseased portion is detected |  |

2. System shall be able to eliminate image distortion.

|  |  |  |  |
| --- | --- | --- | --- |
| Test case id | Input description | Expected output | Actual output |
| 1 | System is able to take blurry image as input | Image has the noise removed |  |
| 2 | System is able to take dim image as input | Image is filtered |  |

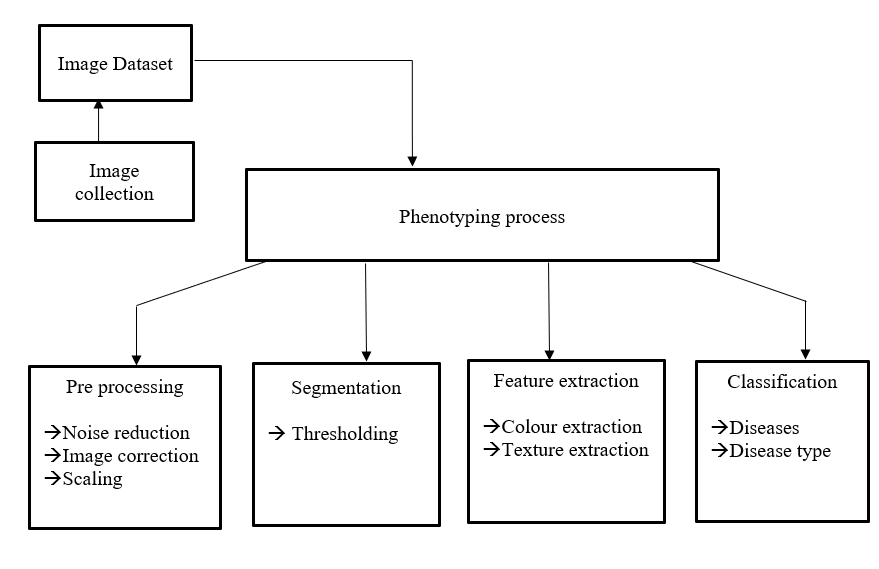
3. System shall be able to predict the type of disease.

|  |  |  |  |
| --- | --- | --- | --- |
| Test case id | Input description | Expected output | Actual output |
| 1 | System is able to identify a disease that is caused naturally | Disease detected is biotic |  |
| 2 | System is able to identify a disease that is caused unnaturally | Disease detected is abiotic |  |

**5. System Design**

**5.1 System Architecture**

A system architecture diagram shows the relationship between different components. Architecture description is a formal description and representation of a system.



*Figure 4: System Architecture.*

System architecture gives the overview of the project. The image is retrieved from storage area and is given to the system. It then undergoes pre-processing where image noise is removed and it is scaled. The it is segmented where the leaf is detected. The features of the image is collected and given to classifier which detects and predicts the disease.

**5.2 DFD level 0:**

Data Flow Diagram (DFD) represents the transfer or rather flow of data from one process to another to perform certain functionality. This describes how data is transferred from one process to another.

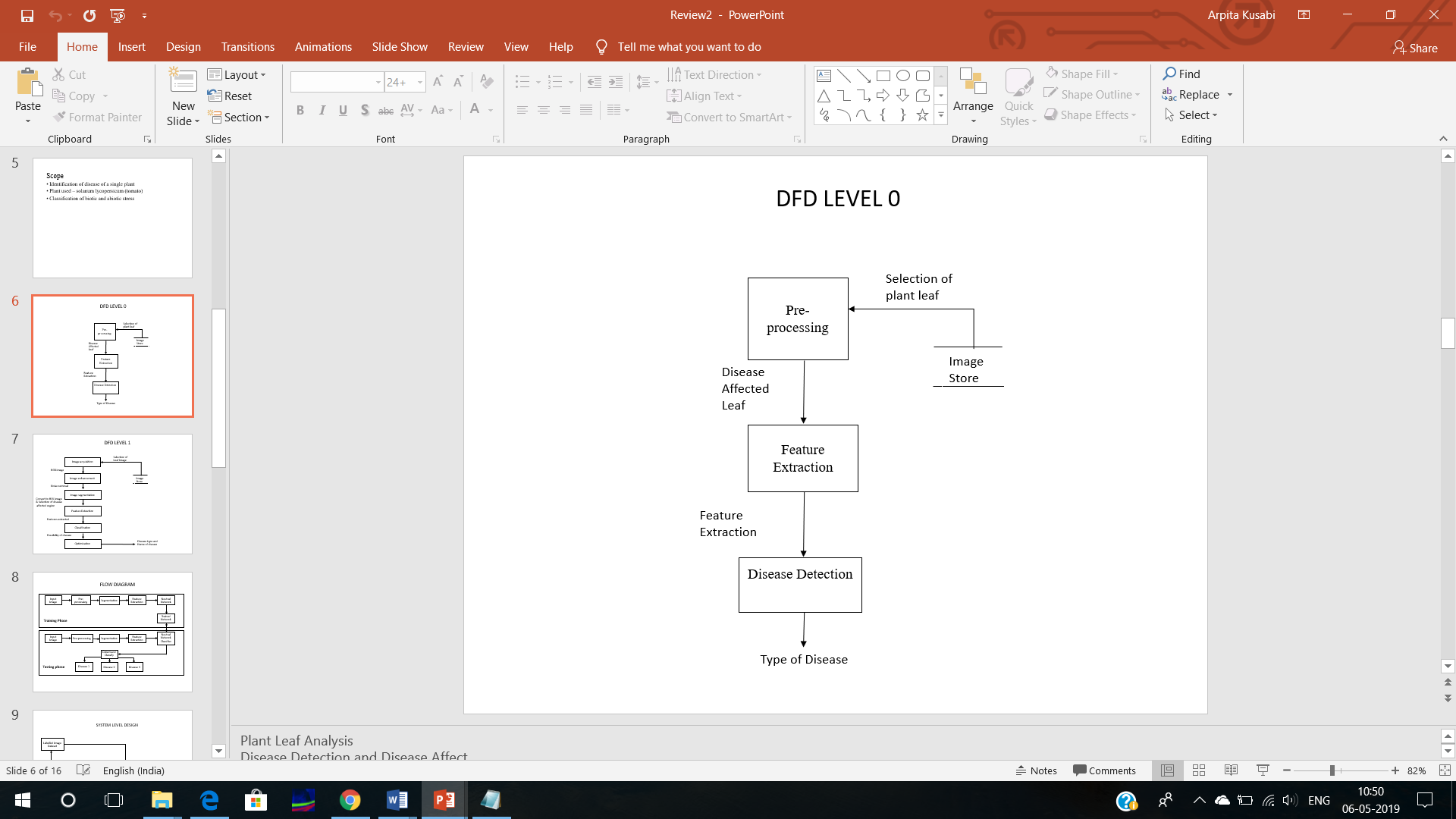


Figure 2:DFD level 0.

The above diagram represents DFD level 0.The image can be given to the system and the disease can be detected.

**5.3 Detailed DFD for proposed system**

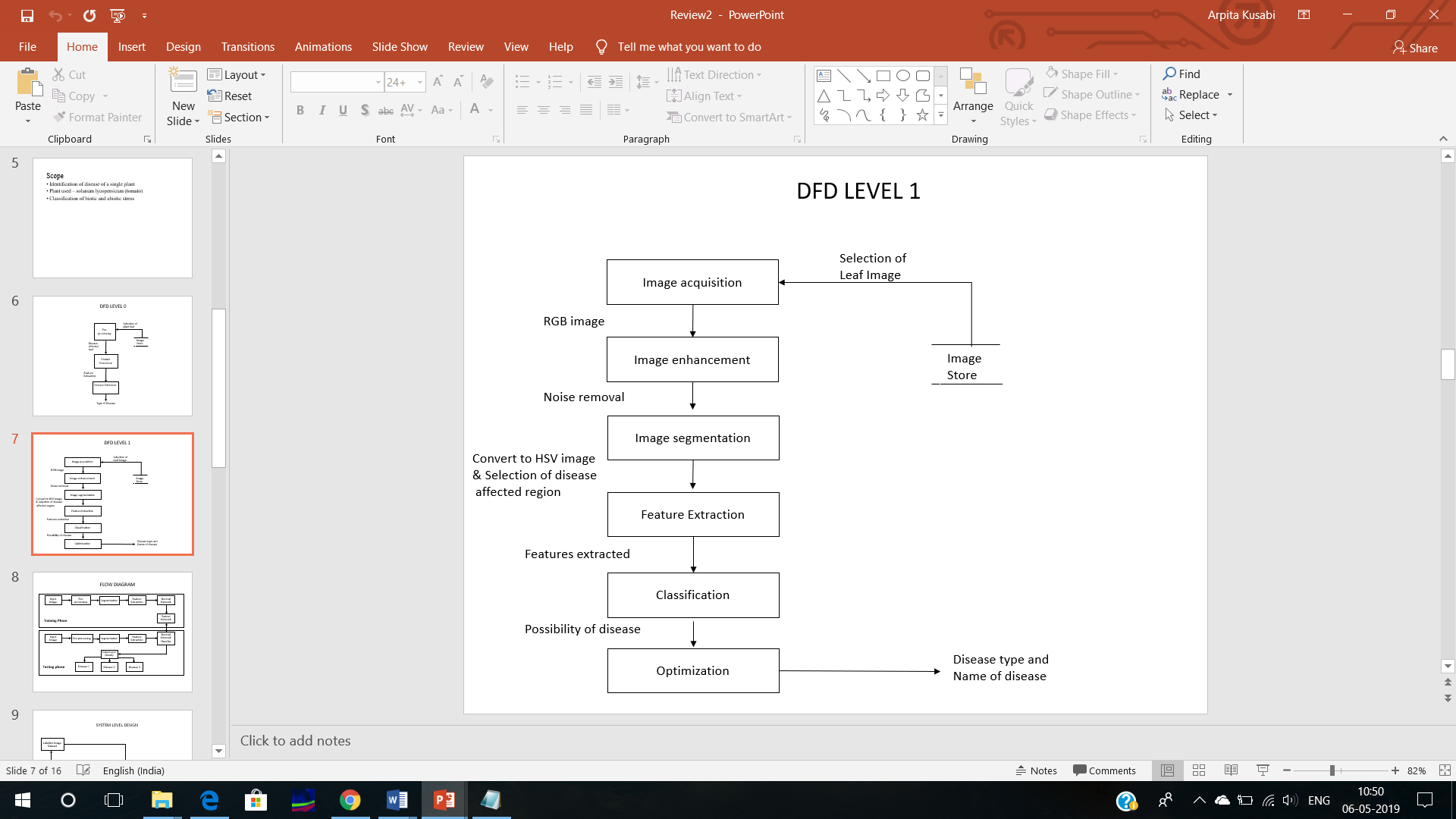


Figure 3: DFD level 1.

The level 1 DFD describes the detailed working of the project .The image will be fed to the system .The image will be segmented and have its features extracted .The features then will be sent to the classifier and disease will be predicted.

**6.** I**mplementation**

**6.1 Proposed Methodology**

The diseases detected are:

1. Bacterial Spot
2. Septoria
3. Late Blight
4. Tomato Mosaic
5. Yellow curve

The stresses classified are:

1. Biotic
2. Abiotic

Canny Edge Detector:

Among the several textual properties in an image, edge-based methods focus on the ‘high contrast between the text and the background’. The edges of the text boundary are identified and merged, and then several heuristics are used to filter out the non-text regions. Usually, an edge filters (e.g. canny operator) is used for the edge detection, and a smoothing operation. The Canny method finds edges by looking for local maxima of the gradient of I. The gradient is calculated using the derivative of a Gaussian filter . The method uses two thresholds, to detect strong and weak edges, and includes the weak edges in the output only if they are connected to strong edges for x and y direction.

Feature Extraction Methods

1. Gray Scale :

Grayscale is a range of monochromatic shades from black to white. Therefore, a grayscale image contains only shades of gray and no color.

While digital images can be saved as grayscale (or black and white) images, even color images contain grayscale information. This is because each pixel has a luminance value, regardless of its color. Luminance can also be described as brightness or intensity, which can be measured on a scale from black (zero intensity) to white (full intensity). Most image file formats support a minimum of 8-bit grayscale, which provides 2^8 or 256 levels of luminance per pixel. Some formats support 16-bit grayscale, which provides 2^16 or 65,536 levels of luminance.

2. Colour features:

* Mean: Average value of the pixels in the image.
* Standard deviation: It shows how much variation or "dispersion" exists from the average (mean, or expected value).

3. Choice of deep learning architecture:

* AlexNet
* GoogleNet

4. Choice of training mechanism:

* Transfer Learning
* Training From Scratch

5.Choice of training-testing set distribution:

* Train: 80%, Test: 20%
* Train: 60%, Test: 40%
* Train: 50%, Test: 50%
* Train: 40%, Test: 60%
* Train: 20%, Test: 80%

Classification Techniques:

KNN (K-nearest neighbour):

The simple version of the K-nearest neighbour classifier algorithms is to predict the target label by finding the nearest neighbour class. The closest class will be identified using the distance measures like Euclidean distance.

When KNN is used for classification, the output can be calculated as the class with the highest frequency from the K-most similar instances. Each instance in essence votes for their class and the class with the most votes is taken as the prediction. Class probabilities can be calculated as the normalized frequency of samples that belong to each class in the set of K most similar instances for a new data instance.

If you are using K and you have an even number of classes (e.g. 2) it is a good idea to choose a K value with an odd number to avoid a tie. And the inverse, use an even number for K when you have an odd number of classes. Ties can be broken consistently by expanding K by 1 and looking at the class of the next most similar instance in the training dataset.

ANN (Artificial Neuron Network):

An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output. ANNs are considered nonlinear statistical data modelling tools where the complex relationships between inputs and outputs are modelled or patterns are found.

  ANN takes data samples rather than entire data sets to arrive at solutions, which saves both time and money. ANNs are considered fairly simple mathematical models to enhance existing data analysis technologies. ANNs have three layers that are interconnected. The first layer consists of input neurons. Those neurons send data on to the second layer, which in turn sends the output neurons to the third layer.

**6.2 Modules**

1. Pre-processing

Image processing is a method to convert an image into digital form and perform some operations on it, in order to get an enhanced image or to extract some useful information from it.

The techniques we have used are:

* Conversion of RGB image to HSV image.
* Filtering – Median filtering, Smoothening, Edge enhancement.

2. Segmentation

Image segmentation is the process of partitioning a digital image into multiple segments. It is typically used to locate objects and boundaries (lines, curves, etc.) in images.

The techniques we have used are:

* Thresholding

3. Feature extraction

Feature extraction is a dimensionality reduction process, where an initial set of raw variables is reduced to more manageable groups (features) for processing, while still accurately and completely describing the original data set.

The features extracted are:

* Texture features using Gray Scale
* Color feature

4. Classification

Image classification is the process of taking an **input** and outputting a **class** or a **probability** that the input is a particular class

The techniques used are:

* Artificial Neural Network (ANN)
* K- Nearest Neighbor (KNN)

**7.Testing**

1. System shall be able to detect the leaf from image

|  |  |  |  |
| --- | --- | --- | --- |
| Test case id | Input description | Expected output | Actual output |
| 1 | System is able to detect leaf from the image | Leaf is detected | The system detects the leaf |
| 2 | System is able to detect diseased portion of leaf | Diseased portion is detected | Affected portion is detected |

2. System shall be able to eliminate image distortion.

|  |  |  |  |
| --- | --- | --- | --- |
| Test case id | Input description | Expected output | Actual output |
| 1 | System is able to take blurry image as input | Image has the noise removed | Noise is removed from image |
| 2 | System is able to take dim image as input | Image is filtered |  |

3. System shall be able to predict the type of disease.

|  |  |  |  |
| --- | --- | --- | --- |
| Test case id | Input description | Expected output | Actual output |
| 1 | System is able to identify a disease that is caused naturally | Disease detected is biotic |  |
| 2 | System is able to identify a disease that is caused unnaturally | Disease detected is abiotic |  |

**8.Result and Discussions**

In this project, we proposed a framework for automatic  
plant phenotyping based on texture and colour features of the  
plants.

Training and Testing Data:

AlexNet GoogleNet

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Transfer learning** | **Training from scratch** | **Transfer learning** | **Training from scratch** |
| **Train:20%, Test: 80%** | **Color** | 0.9736{ 0.9742, 0.9737, 0.9738} | 0.9118{ 0.9137, 0.9132, 0.9130} | **0.9820**{ 0.9824, 0.9821, 0.9821} | 0.9430{ 0.9440, 0.9431, 0.9429} |
|  | **Grayscale** | 0.9361{ 0.9368, 0.9369, 0.9371} | 0.8524{ 0.8539, 0.8555, 0.8553} | **0.9563**{ 0.9570, 0.9564, 0.9564} | 0.8828{ 0.8842, 0.8835, 0.8841} |
|  | **Segmented** | 0.9724{ 0.9727, 0.9727, 0.9726} | 0.8945{ 0.8956, 0.8963, 0.8969} | **0.9808**{ 0.9810, 0.9808, 0.9808} | 0.9377{ 0.9388, 0.9380, 0.9380} |
| **Train:40%, Test: 60%** | **Color** | 0.9860{ 0.9861, 0.9861, 0.9860} | 0.9555{ 0.9557, 0.95+  58, 0.9558} | **0.9914**{ 0.9914, 0.9914, 0.9914} | 0.9729{ 0.9731, 0.9729, 0.9729 |
|  | **Grayscale** | 0.9584{ 0.9588, 0.9589, 0.9588} | 0.9088{ 0.9090, 0.9101, 0.9100} | **0.9714**{ 0.9717, 0.9716, 0.9716} | 0.9361{ 0.9364, 0.9363, 0.9364} |
|  | **Segmented** | 0.9812{ 0.9814, 0.9813, 0.9813} | 0.9404{ 0.9409, 0.9408, 0.9408} | **0.9896**{ 0.9896, 0.9896, 0.9898} | 0.9643{ 0.9647, 0.9642, 0.9642} |
| **Train:50%, Test: 50%** | **Color** | 0.9896{ 0.9897, 0.9896, 0.9897} | 0.9644{ 0.9647, 0.9647, 0.9647} | **0.9916**{ 0.9916, 0.9916, 0.9916} | 0.9772{ 0.9774, 0.9773, 0.9773} |
|  | **Grayscale** | 0.9661{ 0.9663, 0.9663, 0.9663} | 0.9312{ 0.9315, 0.9318, 0.9319} | **0.9788**{ 0.9789, 0.9788, 0.9788} | 0.9507{ 0.9510, 0.9507, 0.9509} |
|  | **Segmented** | 0.9867{ 0.9868, 0.9868, 0.9869} | 0.9551{ 0.9552, 0.9555, 0.9556} | **0.9909**{ 0.9910, 0.9910, 0.9910} | 0.9720{ 0.9721, 0.9721, 0.9722} |

**INPUT/OUTPUT:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Input Image** | **HSV Image** | **Binarized Image** | **Affected area after masking** |
| **Bacterial Spot:** |  |  |  |  |
| **Septoria:** |  |  |  |  |

**9. References:**

* <https://www.ijert.org/research/detection-and-classification-of-plant-leaf-diseases-by-using-deep-learning-algorithm-IJERTCONV6IS07082.pdf>
* <https://www.researchgate.net/publication/321587357_Detection_and_Classification_of_Plant_Leaf_Diseases_in_Image_Processing_using_MATLAB>
* <https://link.springer.com/chapter/10.1007/978-981-15-1384-8_16>
* <https://www.ijarcce.com/upload/2016/november-16/IJARCCE%2044.pdf>