

**Deep Neural Networks Based Detection of Plant Leaf Diseases**  
**by Image Classification**

**Project report submitted in partial fulfillment of the requirement for the degree of**  
**Bachelor of Technology**

**in**

**Computer Science and Engineering**

**by**

**Prashant Gautam (161379)**

**Under the supervision of**

**Mr. Surjeet Singh**

**to**



**Department of Computer Science & Engineering and Information Technology**  
**Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal**  
**Pradesh, India**

**May 2020**

# **CERTIFICATE**

## **Candidate's Declaration**

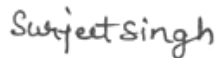
**I hereby declare that the work presented in this report entitled Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification in fulfillment of the requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering submitted in the department of Computer Science & Engineering and Information Technology, Wagnaghat is an authentic record of our own work carried out over a period from January 2020 to May 2020 under the supervision of Mr. Surjeet Singh, Associate Professor, Computer Science and Engineering.**

**The matter embodied in the report has not submitted for the award of any other degree or diploma.**



**Prashant Gautam 161379**

**This is to certify that the above statement made by the candidate is true to the best of my knowledge.**



**Mr. Surjeet Singh**

**Assistant Professor**

**Computer Science and Engineering**

**Dated: 28-05-2020**

## **ACKNOWLEDGEMENT**

**I have taken efforts in this project however; it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them. Foremost, we would like thank my supervisor for the project, Mr. Surjeet Singh, Department of Computer Science and Engineering, Jaypee University of Information Technology, Wagnaghat for his incalculable contribution to the project. He stimulated me to work on the topic and provided valuable information which helped in the completion of the project.**

**I would also like to acknowledge his exemplary guidance, monitoring and constant encouragement throughout the literatures on this project.**

**I am indebted to all the professors of the Department of Computer Science and Engineering, JUIT Wagnaghat for instilling in us the basic knowledge about the fields that greatly benefited us while working on the project.**

**I also thank my friends and peers who extended their help and support whenever needed. Their contribution has always been significant.**

# TABLE OF CONTENTS

## 1. INTRODUCTION

### 1.1 Introduction

#### 1.1.1 Motivation

#### 1.1.2 Technology and Agriculture

#### 1.1.3 Common Leaf Diseases

### 1.2 Problem Statement

### 1.3 Objectives

### 1.4 Methodology

### 1.5 Organization

#### 1.5.1 Convolutional Neural Network

#### 1.5.2 TensorFlow

#### 1.5.3 Tkinter

#### 1.5.4 Jupyter Notebook

## 2. LITERATURE SURVEY

2.1 Detection and Classification of Plant Leaf Diseases by using Deep Learning Algorithm

2.2 Soybean Plant Disease Identification Using Convolutional Neural Network.

2.3 A Deep Learning-based Approach for Banana Leaf Diseases Classification

2.4 Plant leaf disease detection using image processing techniques.

2.5 Damaged paddy leaf disease detection by using image processing.

2.6 Identifying Healthy and Infected Medicinal Plants Using Canny Edge Detection Algorithm and CBIR

2.7 Leaf Disease Detection of Cotton Plant Using Image Processing Techniques

2.8 Plant Disease Detection Using Leaf Pattern

2.9 Detection of Rice Leaf Diseases Using Chaos and Fractal Dimension in Image Processing

2.10 Detection of Disease Affected Region of Plant Leaf Using Image Processing Technique

### **3. SYSTEM DEVELOPMENT**

3.1 Dataset Collection

3.2 Building the CNN model

3.3 Making predictions with the model

### **4. PERFORMANCE ANALYSIS**

4.1 Running the saved model

4.2 Filters application

4.3 Prediction Output

4.4 Testing with different cases

### **5. CONCLUSIONS**

6.1 Conclusion

6.2 Future Scope

### **REFERENCES**

## LIST OF ABBREVIATIONS

<b>CNN</b>	Convolutional Neural Network
<b>ReLU</b>	Rectified Linear Unit
<b>R-CNN</b>	Region Based Convolutional Neural Network
<b>SSD</b>	Single Shot Detector
<b>GDP</b>	Gross Domestic Product
<b>UN</b>	United Nations
<b>GUI</b>	Graphical User Interface
<b>HYV</b>	High Yielding Variety
<b>MLP</b>	Multilayered Perceptron
<b>KNN</b>	K Nearest Neighbor
<b>BPNN</b>	Back Propagation Neural Network
<b>SVM</b>	Support Vector Machine
<b>AI</b>	Artificial Intelligence
<b>FCM</b>	Fuzzy C Means Clustering
<b>PNN</b>	Probabilistic Neural Network

## **LIST OF FIGURES**

- 1.1 Gray spot / Cercospora infected leaf
- 1.2 Common Rust infected leaf
- 1.3 Northern Leaf Blight infected leaf
- 1.4 Forward and Back Propagation from nodes in fully connected CNN
- 1.5 Architecture of the deep CNN used
- 1.6 ReLu Layer Function
- 1.7 Max pooling function
- 1.8 Workflow to be followed
- 2.1 System Design
- 2.2 Classification results from different models.
  - 2.2.1 Training vs Validation accuracy
- 2.3 Proposed framework architecture
- 2.4 Prediction Accuracy Comparison
- 2.5 Filters applied on the set of images
- 2.6 SS from the MATLAB application
- 2.7 Feature Extraction
- 2.8 Processing Techniques Workflow and System overview
- 2.9 Leaf Disease classification based on feature points
- 3.1 Infected leaves samples
- 3.2 Installing & Importing required dataset

- 3.3 Images loaded are displayed at random
  - 3.3.1 Converting the images to workable data
- 3.4 Applying transformations and filters
- 3.5 Rechecking for the transformed dimensions
- 3.6 Building the CNN model
- 3.7 Model In training phase
- 3.8 Model 0 Training vs Validation Accuracy
- 3.9 Model 1 Training
- 3.10 Model 1 Training vs Validation Accuracy
- 3.11 Model 2 Training
- 3.12 Model 2 Training vs Validation Accuracy
- 3.13 Model 3 Training
- 3.14 Training vs Validation Accuracy
- 3.15 Prediction values
- 4.1 Importing libraries
- 4.2 Prompt to select the model number
- 4.3 Dialog box to select image
- 4.4 Prediction made by the model with 99% probability
- 4.5 Result shown in the pop-up window
- 4.6 Healthy Leaf with Probability 89%



4.7 Common rust with 100% probability

4.8 Northern Leaf Blight with 96% probability

## **LIST OF TABLES**

**2.1** Tabular comparison of techniques used in the past recent years

**3.1** Model Comparison Table

## **ABSTRACT**

**It is often observed that when the crops and trees are affected by various diseases it affects the grain and agricultural production of the country as a whole. Organically farmers and cultivators observe the plants with naked eye without any accurate judgment for the detection and identification of a disease. This method proves be time consuming, expensive and even inaccurate at times. Artificial detection using image processing techniques offers fast and precise results. Advances in artificial intelligence and machine learning provide an opportunity to expand and improve the practice of precise crop protection and extend the market of deep learning applications in the field of professional agriculture. Text book way of training facilitates a system implementation in practice. Various steps required for implementing this plant disease recognition model are discussed throughout the report, starting from compiling images in order to create an integrated database, assured by agricultural experts present online, a deep learning framework to perform the deep CNN training. This is a new approach in detecting plant diseases using the deep convolutional neural network which are trained and finely tuned so as to fit accurately to the dataset of affected plant's leaves that was gathered for various plant diseases. The uniqueness of the developed model lies in its working; healthy leaves are in line with other classes, which enables the model to distinguish between diseased leaves and healthy ones or from the background noise using deep CNN.**

**Keywords: Deep Convolutional neural networks, Deep learning, Image Classification, Plant disease, Rectified Linear Units.**

# CHAPTER 1

## INTRODUCTION

### 1.1 Introduction to Agronomics

The art of Agronomics or crop cultivation is the methodical practice done primarily by the homo sapiens in which they use land to grow crops, fruits, vegetables rearing and management of livestock. The word agriculture is essentially a *latin* word agriculture which rounds up to just two words soil and culture.

According to recent studies, more or less 50% of the entire world's population and manifestly the 75% of India's population is directly or indirectly engaged in an agricultural activity of some sort. It is known that India's agriculture stratum reports for approximately eighteen percent of India's GDP *vis-à-vis* gross domestic product and in fact provides employment opportunities to half of the country's workforce. Substantially, India is the world's largest exporter and producer of grains, lentils, rice, wheat and numerous spice products. Miscellaneous industries like the forestry and pisciculture commercially also come under all of these prevalent agricultural practices. It also fulfils the basic needs of the vast human population like food, shelter and clothing.

Statistically, one fifth of the aggregate exports of the country are agricultural goods. Agricultural statistics and predictions play a major role and hold formidable significance to a larger population at hand. In an attempt to enhance the quality as well as surge the quantity of the produce, various new state of the art techniques have been proposed in the field of agricultural sphere. Better and fairly advanced high yielding seeds, fertilizers, customized irrigation techniques are some of the practices which are quite prevalent and have increased the harvest yield and thus increased the land available under cultivation. To run a successful agrobusiness one needs to focus on a few fundamentals like HYV Seeds, soil conditioners, manure, fertilizer, machinery and handful labor. The human populace is outgrowing itself persistently over the years which can be sustained only by looking for better cultivation practices and expanding the use of ultra-advance power-assisted technologies and machineries. India possesses a vivid agricultural palette which consists of diverse crops, the most prominent of which are rice, corn maize and wheat. Despite having a colossal and enormous potential

in the agricultural sector, the overall end yield of crops per hectare area in India is marginally lower than that of the widely accepted international standards.

The farming sector plays a climacteric role in the believed diversification of the economic sector to a larger dimension. On the ground level there might be some minute trackable progress in this dormant field, but there are certain problems which are still quite common and have been persistently affecting the Indian farmers for the past decades. It is also conjectured that one-fifth of comprehensive agricultural production is crumbled to dust due to lack of proficiency and incompetence in garnering, harvesting, transporting and stockpiling of the of the subsidized crops and end products.

### **1.1.1 Motivation behind the project**

It is well known fact that both the Central and State federal Governments churn significant amount of revenues from the agriculture sector. Roadways and Railway industry also derive a considerable part of their income from the dissemination of agriculturally produced goods. Thus, it can be safely said that agriculture is an indispensable understructure in the country's economy and can't be ignored that easily. As per the Economic Survey of the financial year 2019-20, agriculture sector puts more than 50 per cent of the total workforce in India to a gainful employment and contributes just around 17-18 per cent to the country's GDP at large. The survey also suggests that the farmers of the new building nation are now keener to adapt relatively new farm mechanization and smart techniques at a faster pace as compared to recent years. The crops and plants growing in abysmal conditions are very much susceptible to various kinds of complex diseases which are proportionately very difficult and baffling to espy through naked eyes without any prior knowledge in the field.

The 21<sup>st</sup> century contemporary farms and agricultural activities/operations show a major leap from activities few decades ago, predominantly because of technological evolution, with the introduction of sensors, tools, embedded devices, machinery, and largely the whole realm of information technology. With the current market demand in the new era agriculture frequently involves using sophisticated technologies such as humidity, temperature and moisture sensors, aerial overview from drones, robotic equipment and GPS technology industrial science has become a customary standard to some markets. These advanced devices generally associated with the precision agricultural

techniques/methods empower the agribusinesses to be more rewarding, profitable, efficient, non-vulnerable, and more ecofriendly to the nature.

### **1.1.2 Technology and agriculture**

With the disruption in the technological innovations and advancements in farming marshalled by robotic engineering evolution and bleeding edge sensor technologies have collectively reshaped the future of farming practices over the course of time. For the prolonged centuries, farmers have taken over more and more tricks up their sleeves in the forever pursuit of foremost greater yields, the popular credence that more quantity the better notwithstanding plays a preeminent part in this day and age of farming activities, which eventually leaves all the small-scale operations to be practically non-viable. Intelligent and intuitive robotic systems have a fascinating capacity to change the landscape of the current financially rewarding model of farming so that it becomes reasonably feasible and accessible to be a modest producer thus also deranging today's agribusiness panorama.

The modern twenty first century programmed and AI technologies have the potential to solve the archaic problems known in the traditional farming practices. An artificially automated agronomic system, can make crop production remarkably more efficient, long lasting, minimize costs and thereby boost substantial quality. Advanced devices to monitor vegetable growth, improve monitoring and nurturing soil quality are currently in development. According to a UN body, The Food and Agriculture Organization reportedly 20–40% of universal annual crop yields are squandered each year to pests and diseases, making tons of unaware and immature pesticide decisions. Intelligent self-regulating devices, such as robots and drones enabling the culturists to shed the excessive chemical use by perceiving a potential threat to the whole crop beforehand and thus taking a well thought decision and using a precise chemical application or technique to eliminate the pests. Crop monitoring drones well equipped with RGB high definition cameras can identify a potential threat or pest feeding up a crop.

These types of practices have given rise to a new branch called as the precision agriculture. Precision agriculture advancements are majorly revamping the old school cultivation as it introduces some valuable assets to the current demands. Precision agriculture is essentially an analytical data driven

*modus operandi* for supervising and increasing the yield of crops. In recent times cultivators are switching on to the advanced methods of precision agriculture methodologies for several reasons:

- Relatively more yield of crop productivity and better safety standards for workers.
- Uses less input materials by adding more value in a longer run.
- Substantially low disposal of chemicals into water sources, thus more environment friendly.
- Well-engineered and dependable supervision and monitoring of air and water quality. It puts the producers themselves in saddle giving a much broader control over plant processing, storage and dispersal.

### 1.1.3 Some of the common corn leaf diseases

#### ▪ “Cercospora” / Gray Leaf Spot

This type of disorder in a plant leaf incurs in the form of slate gray to brownish tan which is rectangle shaped abrasions on leaves, sheaths of leaf or may be present in the leaf husks. The fungus forms spores on the underside of leaves. This type of condition arises in extended periods of overly warm, cloudy days and consistently high ranges of humidity. With increased usage of reduced tillage and continuous corn more cases are being observed.



*Figure 1.1 Gray spot / Cercospora infected leaf*

- **Common Rust**

As the name suggests this can be identified when rust like tiny, cinnamon colored powdery spots surface on the upper and lower side of the corn leaf. When the infection is matured enough this powdery rust spores can be rubbed off. Regions with moderate to cool temperatures and high humidity give rise to this type of infection. The fungus arrives each season from crops grown in more southern regions.



*Figure 1.2 Common Rust infected leaf*

- **Northern and Southern Leaf Blight**

This type of disease is observed when elliptical to cigar-shaped, gray-green lesions turn



*Figure 1.3 Northern Leaf Blight infected leaf*



into tan-brown color. This infection starts from the lower leaves and with the progression of season the disease spreads to the upper canopy of the plant. High humidity and moderate temperatures are preferred conditions for this disease. Under humid conditions, lesions may appear dark, and fuzzy because the fungus forms spores on the dead tissues.

## **1.2 Problem Statement**

To develop a neural network system which recognizes a corn plant infected with bacterial and infectious diseases by identifying the early symptoms from its visual appearance. Using this method provides an edge in precise prediction over identifying manually through naked eyes which usually leads to human error. Using this intelligent automated technique potential hazards can be known beforehand and necessary actions can be made, thus avoiding crop failure in a larger domain.

## **1.3 Objectives**

- Creating a compilation of the dataset containing leaf images over diverse pictures.
- Training and developing a CNN model on collected dataset using TensorFlow and implementing the Keras.
- Building a classifier that can predict the plant disease on different labels, also improving on its prediction accuracy as compared to the previous models and approaches.

## **1.4 Methodology**

Deep learning is a modern technique prevalent in current times primarily used for processing of image and its analysis on the data extracted from it. Instead of using image segmentation and feature extraction we can take use of this approach in current problem at hand. A supervised learning model is developed for the detection of these corn leaf disorders. Fundamentally speaking this approach is based on Convolution Neural Network. To classify these images, we need to apply various filters thus identifying each pixel in a specific image which is made neural network ready. This labelled and well-arranged data is then used to extract some unique patterns and features.

The model first created essentially has three layers thus reducing the complexity. The first layer of the structure consists of strides of 3 x 3 filters followed by max pool layer and activation function as a relu function layer. The next layer will flatten out all the extracted features from the previous layer and then feed it forward to the SoftMax layer.

In the approach of building the proposed model, we opted for Rectified linear unit relu instead of sigmoid function because it is more efficient and returns faster results for training deep neural networks and improves the overall accuracy. The proposed CNN will be refined with continuous forward and backward propagation among the nodes.

**Forward propagation:** On first training the model at an instance model attempts to find certain patterns and features feeding them forward thus giving predicted result at the end flattened labels layer. This mechanism followed by nodes or neurons is defined as forward propagation. Every node in the network hold some value of features is then fed forward to the next nodes.

**Backward propagation:** Following the forward propagation mechanism in one iteration, the neural network propagates to the previous nodes in a backward fashion. This process helps in calculating the mean value error which defines new bias and weights selection to increase accuracy in the predicted outputs thus and reducing the error to its minimal value. The continuous flow of these values from one node to another makes the model find its inefficiencies and improve on them to give accurate predictions and minimize the error.

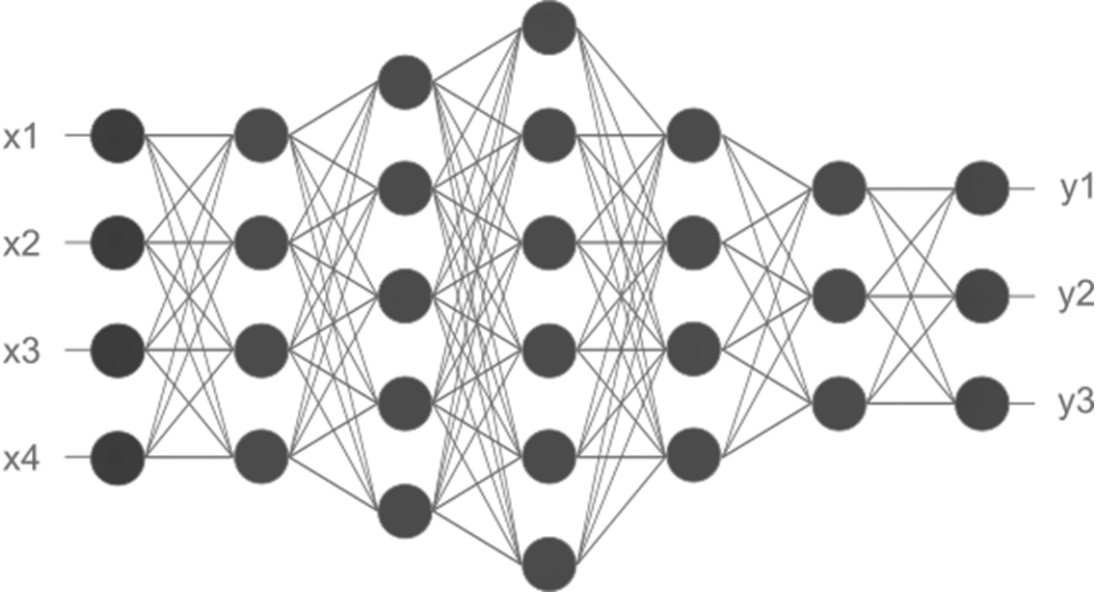


Figure 1.4 Forward and Back Propagation from nodes in fully connected CNN

The architecture of the Convolutional neural networks (CNNs) is constructed up of collective and diverse layers of fields having reception. Each node or even minute neuron collections are responsible for the processing fragments and chunks of the input image fed to one end of the network. Collaborating the outputs of all these collections are further tiled in order to the overlapping of input regions, thus resulting in the representation of the original input image in much higher resolution; this process is replicated intuitively for every layer in the network. The Tiling technique allows CNNs to evaluate translation of the input image. Convolutional neural networks can be composed of various local and global pooling layers. These layers contribute in combining the output received from these neuron clusters and combinations of the overall convolutional and fully connected layers.

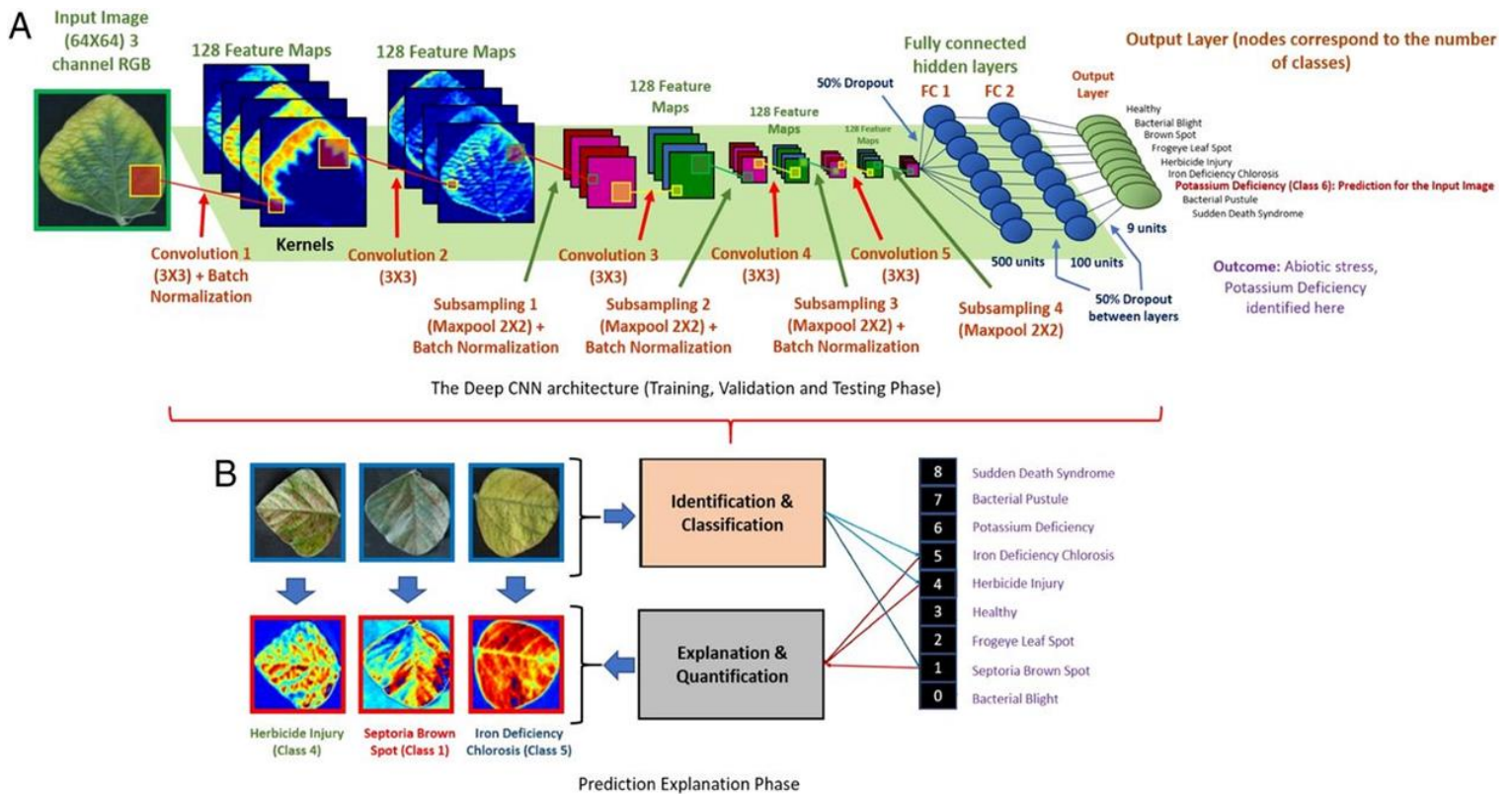
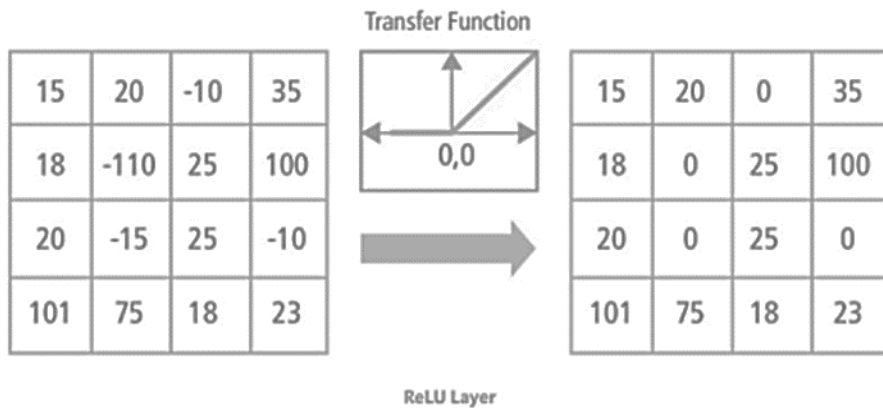


Figure 1.5 Architecture of the deep CNN used

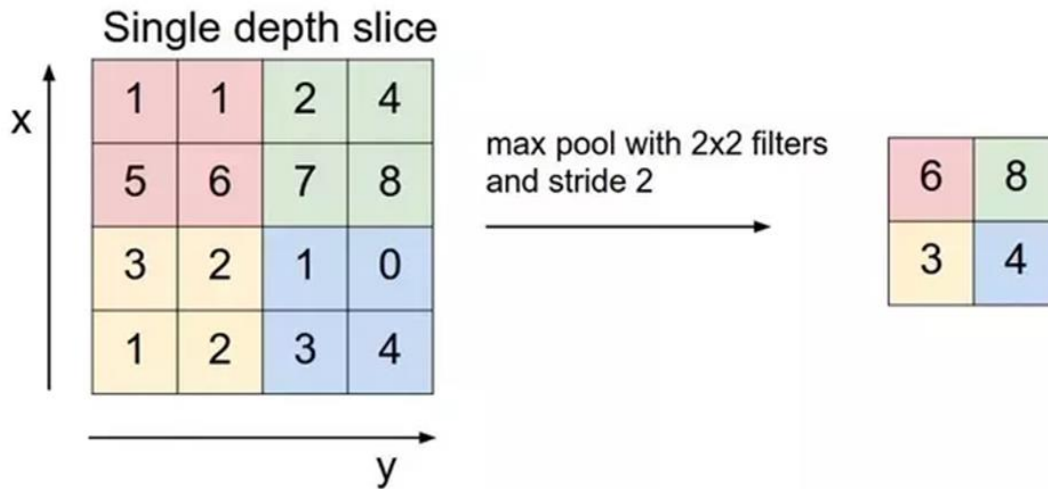
Convolutional Neural networks constitutes local and global pooling layers, which work on a collaborative level and finally the output of neuron clusters is forwarded ahead. The confusional matrices are integrated to form a fully connected layer having nonlinearity applied to each point after the end of each layer.

- Building up the neural network, it also we used rectified linear unit layer and a fully connected layer. This ReLU layer will perform the function of an activation function, and will ensure non-linearity all over as the data moves to each layer in the network. If the network is deprived of this layer the data which is fed into each layer will end up losing the required dimensionality that needs to be conserved across the whole computation.



*Figure 1.6 ReLu Layer Function*

- The convolutional layer runs by placing a filter over the matrix array of image pixels. This then creates a convolved feature map, in layman's language it's like looking at an image through a window, which allows you to see only specific features you might not otherwise be able to see.



*Figure 1.7 Max pooling function*

- Now in the next step the Pooling layer effectively downsamples or decreases the sample size of a particular feature map. This also makes processing much optimized as it reduces the number of parameters the network needs to process. The output of this process is a pooled feature map.
- The next prominent step is max pooling, which essentially extracts out the maximum input to a more simplified and optimized convolved feature, or sometimes concept of average pooling is taken to use which simply takes the average. These steps then collectively lead to feature extraction which includes the network building up a picture of the image data with respect to the mathematical rules.
- In the end of the neural network the features are then labelled out to a flattened layer to make certain accurate predictions into different categories.

## 1.5 Organization

The first step towards creating a model is to prepare the data set. After that we are going to write a function to encode the labels. Labels are nothing but the dependent variable that we are trying to predict. So, in our

training data and testing data we want to make sure that the labels are right so on that basis we can train our model so we are going to encode the labels first. After that we'll resize the image to 50 cross 50 pixel and we are going to read it as a grayscale image then we are going to split the data of approximately 24,000 images for training and 50 for testing once this is done we are going to reshape the data appropriately for TensorFlow. For the implementation of deep learning models, we are going to build the model and calculate the loss which is nothing but categorical cross entropy value. Then we are going to reduce the loss by using Adam optimizer with a learning rate set aside to point double zero 1 then we are going to train the deep neural network for about 10 epochs or even 15 epochs or iterations and finally we are going to make predictions from the built classifier.

### 1.5.1 CNN

The fundamental unit of this project is a Convolutional Neural Network (CNN) which is an optimized Deep Learning technique which can take image characteristics, appoint significance to certain features namely bias and weights at all the nodes to different viewpoints/prototypes in the picture and have the option to separate one from the other. These are less complex in comparison to other algorithms used for classification. These networks are more intuitive in making important decisions and do not require much of the manual work.

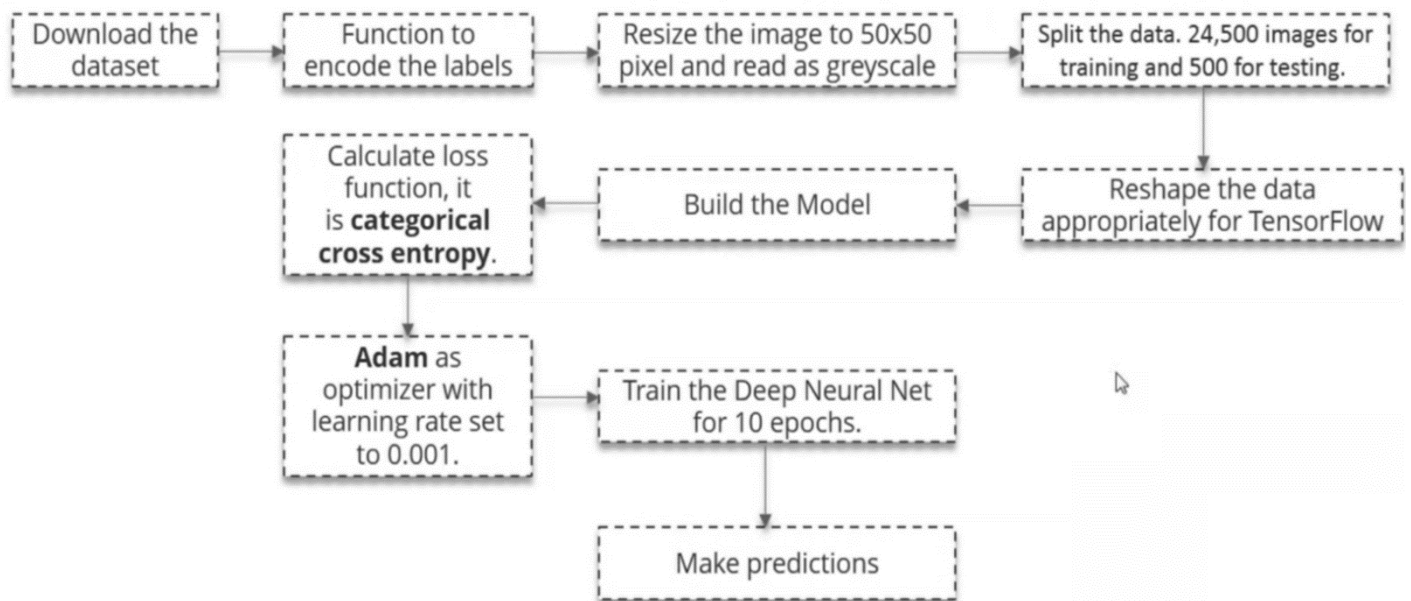


Figure 1.8 Workflow to be followed.

### **1.5.2 TensorFlow**

We use TensorFlow in our approach as it is open-source for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks.

Google's TensorFlow is currently most trending python library used for image recognition and classification purposes.

These are nothing but multidimensional arrays, two dimensional tables or multi-dimensional. Its main feature includes its ease of use in implementing complex deep learning models with few lines of code.

### **1.5.3 Tkinter**

We Tkinter in the later stages of development to create a usable GUI that that is more user friendly. The user is able to select an image for processing from the device itself from a dialog box prompt. This Python GUI toolkit proves to a great help for connecting with the operating system controls.

### **1.5.4 Jupyter Notebook**

Using the anaconda navigator, the Jupyter Notebook we were able create our CNN model in Python language. It is basically an open-source web application that can be used for data cleaning and image transformation, TensorFlow, statistical and analytical modeling, data visualization and machine learning.

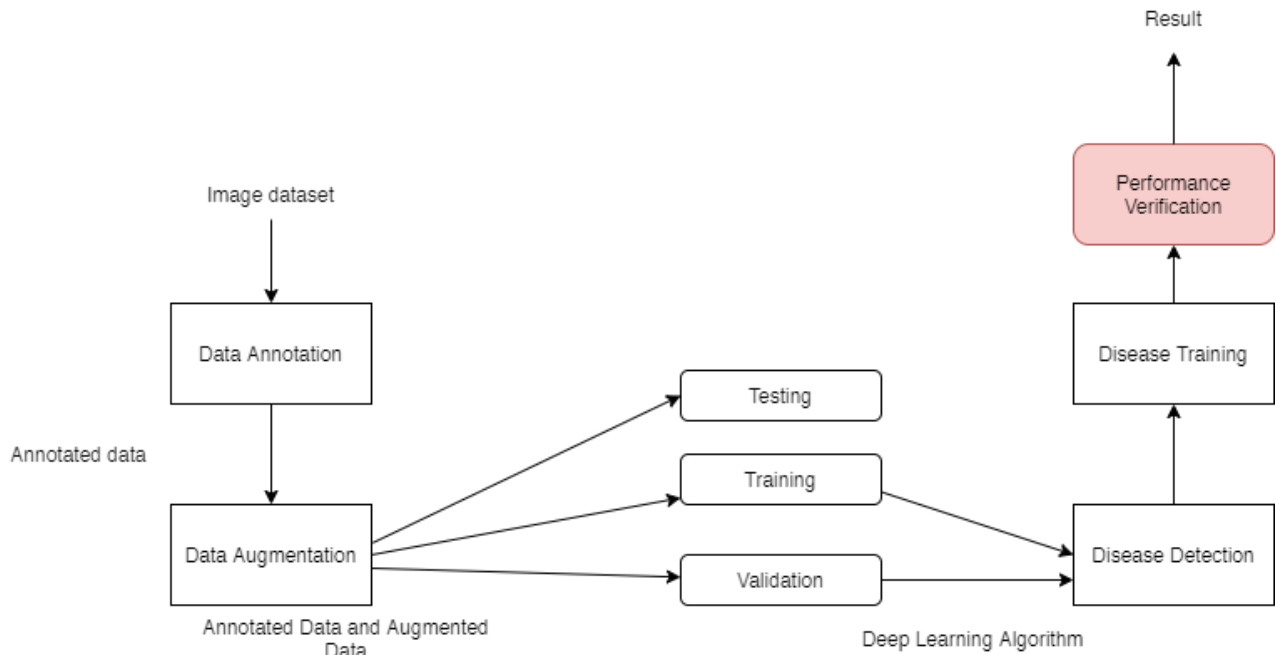
## CHAPTER 2

### LITERATURE SURVEY

#### [2.1] Detection and classification of plant leaf diseases through deep learning Algorithm

*M. Akila, Prabu Deepan [2018]*

The proposed framework had a particular automated learning model in this paper, this framework depends on a particular convolutionary neural system for the discovery of leaf illness. Different cameras and assets caught the informational collection images. Pests and illness are not an issue in farming, as indicated by look into, as solid plants and plants developing in rich soil can withstand bother/ailment. Indicators, for example, Faster Region-based Convolutionary Neural Network (Faster R-CNN), Fully Convolutionary Region-based Network(R-FCN), and Single Shot Detector (SSD) were utilized. The dataset was isolated into three sets to play out the test, for example Set of approval, set of preparing and set of tests. The principal appraisal is performed on the approval set, after the neural system preparing is performed on the preparation set and the last assessment is performed on the test set-to assess results on crude information, they use preparing and approval sets to execute the preparation procedure and test set.



*Figure 2.1 System Design*



## [2.2] Identification of soybean plant disease using the Convolutionary Neural Network

*Jiangsheng Gui, Mor Mbaye [2019]*

In this report, soybean plant contamination is recognized utilizing convolutional neural system. The arrangement of information contains pictures from normal sources. In the analysis of sickness, the procedure productivity of 99.32% is accomplished. The test likewise shows that the preparation set's information increment improves the system's exhibition. This present paper's proposed CNN model comprises of three convolutionary layers, trailed by a maxpooling layer for each layer Fully associated with MLP is the last layer. The enactment capacity of ReLu is applied to every convolution layer's output the yield layer is given to the softmax layer that gives the four yield classes of likelihood circulation. The arrangement of information was isolated into preparing (70%), approval (10%) and testing (20%)

	Architecture	Validation accuracy	Test accuracy
Grayscale	[3X3, 4X4]	77.60%	78.74%
	[5X5,5X5]	70.20%	70.07%
	[3X3, 4X4]	77.20%	78.67%
	[3X3, 2X2]	77.60%	77.87%
Color	[3X3,4X4, 1X1]	89.30%	88.20%
	[3X3,2X2,2X2]	89.50%	86.90%
	[3X3,4X4,3X3]	89.90%	88.00%
	[3X3,4X4]	88.00%	85.50%
Segmented	[5X5, 3X3]	87.30%	85.30%
	[3X3,4X4]	87.60%	85.90%
	[3X3,2X2]	87.00%	85.50%

Figure 2.2 Classification results from different models.

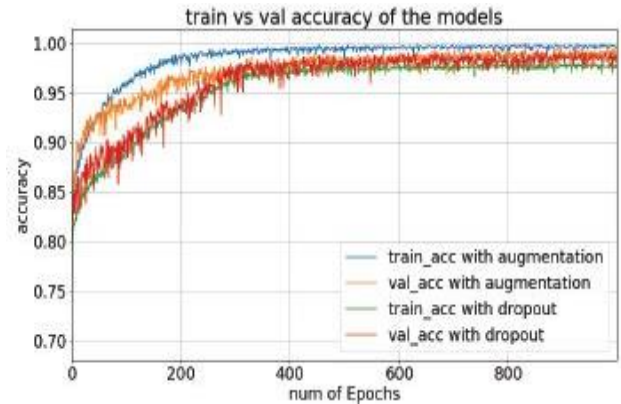


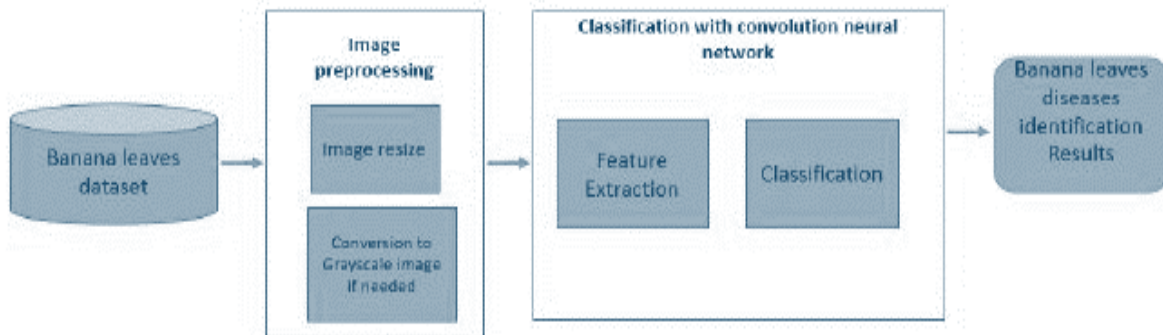
Figure 2.1.1 Training vs Validation accuracy

From the order investigation, it tends to be seen that shading pictures are more exact than dark images. Color pictures are in this way better for extraction of the component. The model result likewise demonstrates the model to be overfitting. Overfitting happens when the layout is fruitful in fitting the preparation cluster. It at that point turns into a speculation of new occurrences not in the preparation set.

### [2.3] A Deep Learning Approach Classification of Banana Leaf Diseases

*Bassem Bouaziz, Jihen Amara, Alsayed Algerawy [2017]*

This paper proposed to characterize and arrange banana ailment through the convolutionary neural system. The proposed model will help ranchers in banana plant illness detection. With the signs, the rancher will snap a photo of the leaf and the procedure can survey the sort of disease. The specialist utilized profound neural systems to discover two unmistakable banana ailments that join banana shimmering Sigatoka and banana in the genuine scene and under troublesome conditions, for example, lighting, muddled setting, totally extraordinary picture resolution, size, cause, and orientation. Once numerous analysts had the option to see the impacts of keen grouping. This has affirmed that with almost no framework exertion, the proposed approach would significantly help Associate in Nursing to accurately distinguish leaf infections. We utilized a profound learning approach in this paper to portray and depict the ailment of banana leaves. The structure of the technique comprises of two components, for example picture preparing and profound arrangement dependent on perusing.



*Figure 2.2: Proposed framework architecture.*

### [2.4] Using image processing techniques, plant leaf disease detection

*Richards E.Woods and Rafel C.Gonzalez [2016]*

This paper incorporates an overview of different methods for arrangement that can be utilized for distinguishing proof of plant leaf maladies. Choosing a characterization strategy is consistently a difficult assignment since various information will differ in the estimation of the outcome.

Arrangements of plant leaf malady have wide applications in various fields, for example, natural

research, farming, and so forth. This paper gives an outline of the different procedures used to characterize plant leaf malady. Plant infection distinguishing proof is the way to forestalling misfortunes in horticultural item yield. This paper investigated various procedures for fragmenting the plant's ailment partition. This paper utilizes picture preparing strategy to introduce an example of various techniques for plant leaf malady recognition. There are numerous techniques for the distinguishing proof and arrangement of ailments in programmed or PC vision, however this exploration zone is as yet absent. It is difficult to recognize all the sickness utilizing a solitary method We arrive at the accompanying resolution from the examination of the above characterization strategies. Maybe the easiest of all calculations to foresee the class of a test model is the k-closest neighbor strategy. The time multifaceted nature of making expectations is an undeniable downside of the k-NN technique. Neural systems are additionally lenient of loud information sources. In any case, it's difficult to comprehend calculation structure in the neural network's was seen as effective in characterizing high-dimensional informational collections with the best accessible AI calculations.

#### **[2.5] Detection of damaged paddy leaf disease by image processing**

***Priyanka B Raj, Hegde Soumya G, Dr. Neha Mangla, Pooja R [2018]***

A respectable approach is proposed in this paper, for example picture handling of the paddy leaf by histogram, to maintain a strategic distance from the impact of these maladies for an enormous scope. Through this strategy, one can recognize the infection at an exceptionally essential stage and accordingly find a way to diminish yield misfortune in time. The proposed strategy is planned for setting up a certified plant leaves sickness reviewing framework. Distinctive leaf tests are considered for exploratory purposes. The accompanying advances are followed for analysis:

- (A) Image Enhancement,
- (B) Pre-process picture
- (C) Image division,
- (D) Histogram change,

(E) Detection of paddy sickness. Histogram Equation:

$$M = A(i) = \dots = [I] Z_i(x) dx \dots (1) \quad m v = A(iv) = \dots = Z v i(ik) \quad k=1 \dots = 1 v k / 1 = \dots$$

—(3) For  $v=1,2, \dots, N$  where  $m k=1 v$  is a measure of strength.

For the grading of diseases, a histogram-

based system is developed by referring to the scale of the disease scoring in Table I.

$1 1 0 0 ( , ) \log ( , ) 4 n m$  Paddy  $i v i x y G A n m Z x y - - = = \sum \sum$  ----- Where Paddy  $G =$  grade value,  $( , ) Z x y i =$  Role of matrix image, value of  $n =$  row and value of  $m =$  column

### DISEASE GRADING VALUE BY PICKS VALUE

Table: 1

Disease Grade	Training Sample	Testing Sample	Classifier Accuracy
Blast	196	82	87%
Bacterial Leaf Blight	205	83	92%
Rice tungro	210	71	90%

Figure 2.4 Prediction Accuracy Comparison

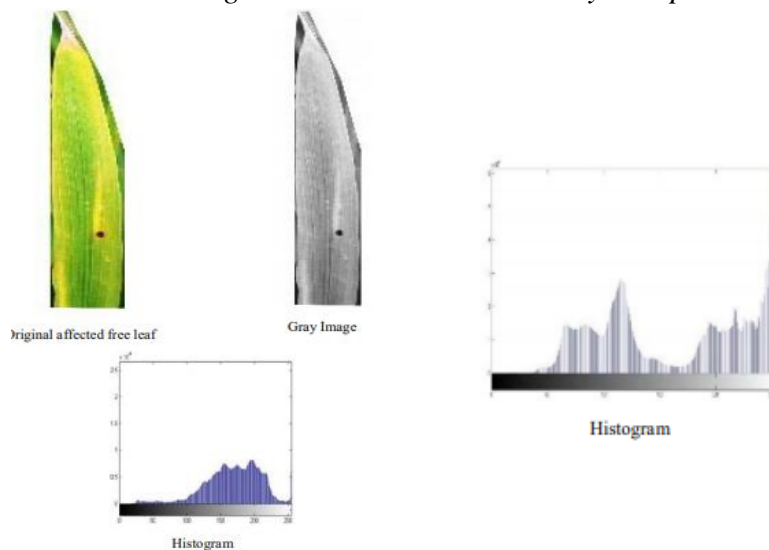


Figure 2.5 Filters applied on the set of images

## [2.6] Identifying medicinal plants that are safe and tainted with Canny Edge Detection Algorithm

*Vinita M. Tajane, N.J.Janwe professor [2019]*

This paper presents a method for the application of canny edge detection algorithm on medicinal plants ' healthy and disease samples Farmers suffer from the problem that arises from different types of plant characteristics / diseases.

The very first step is to apply canny edge detection algorithm to safe and diseased samples in order to identify the plant disease.

### **CANNY EDGE ALGORITHM DETECTION:**

The methodology here is that the images of healthy and infected plants were taken for the first time. Then apply the algorithm for canny edge detection to samples.

Canny edge detection algorithm which retains the structural properties for further processing of images.

The general purpose of edge detection is to reduce the amount of data in an image substantially The objective of this algorithm for the following criteria:

Detection:

It should increase the probability of detecting actual edge points while decreasing the likelihood of wrongly detecting non-edge points.

This is equivalent to maximizing the ratio of signal to noise.

ii) Localization: the edges found should be as close to the actual edges as possible.

iii) Number of responses: not more than one observed edge will result in one real edge. The algorithm for detection of the Canny Edge runs in 5 separate steps:

1. Smoothing: photo blur to eliminate noise. Implemented with Basic Kernel Size (N) and Gaussian Envelope Parameter Sigma by Gaussian Filtering.

2. Finding gradients: where the gradients of the image have large magnitudes, the edges should be marked.

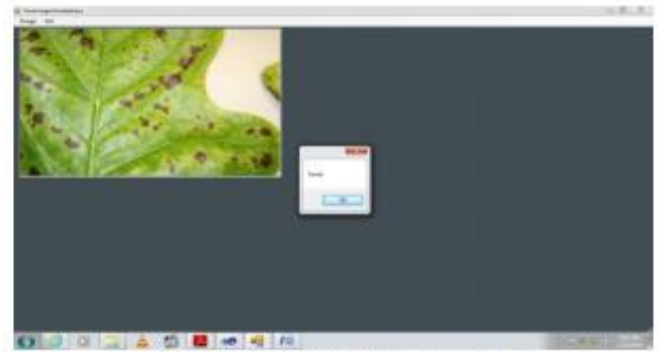
3. Nonmaximum suppression: edges should be labeled only as local maxima. Find gradient direction and perform non-maxima suppression using these directions.

4. Double thresholding: thresholding is used to evaluate possible edges.

5. Hysteresis edge tracking: Final edges are determined by suppressing all edges not connected to a very certain (strong) edge.



Healthy Image Save into Database



Infected Image Save into Database

Figure 2.6 SS from the MATLAB application

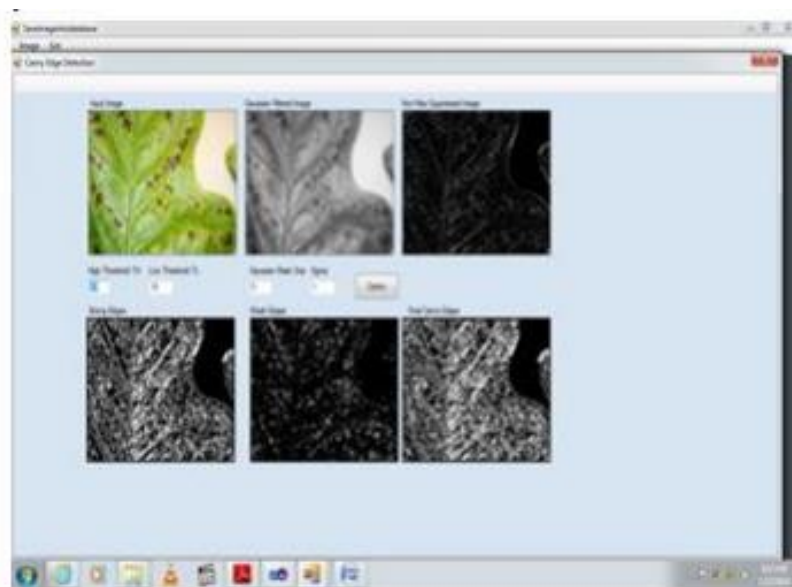


Figure 2.7 Feature Extraction

## [2.7] Detection of cotton plants with leaf disease using image processing techniques

*P. Gulve, Tambe Sharayu S., Ms. S.S. Kanse Pranita [2016]*

Important case for providing better care to protect the crop is proper identification of the disease. Spatial FCM & PNN classifier image processing gives the best result in identifying the type of disease in cotton plants.

In which the acquisition of images is carried out. A median filter is used to obtain pre-processing later. The preprocessed images of the leaf are then segmented using the clustering approach of Spatial FCM. Then the color features texture features such as power, entropy, similarity,

contrast, edges are extracted from the diseased image of the leaf using repeated texture configuration and then compared to regular image of cotton leaf.

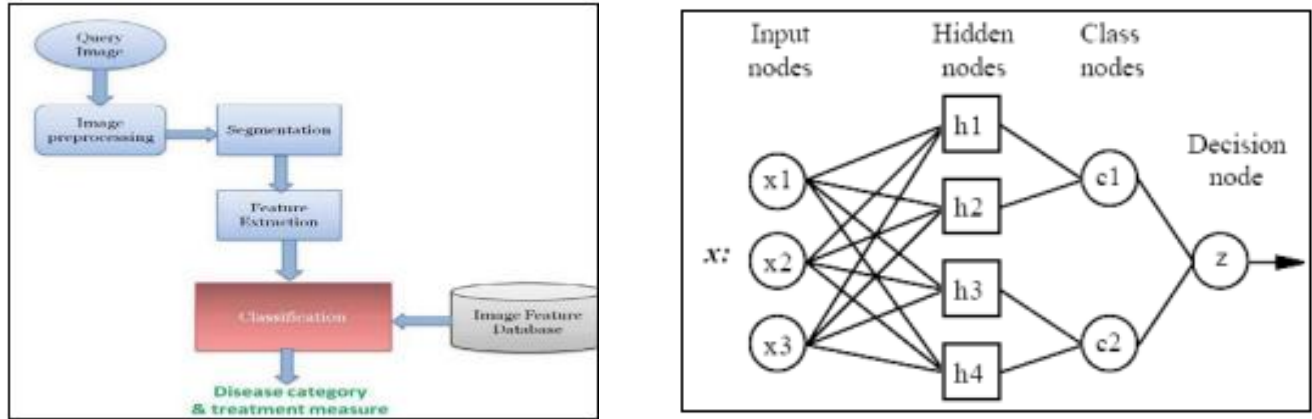


Figure 2.8 Processing Techniques Workflow and system overview

This research paper was used to analyze some technique for the identification of cotton plant leaf disease. Normal human eye on cotton leaf can not discern differences in color and texture. The software can be used to capture color and texture characteristics. Another software used will compare with the trained database and can be identified according to this disease.

Technology for image processing used to deploy the classification system. Thus, image processing with spatial fcm and pnn used to automatically detect and classify the diseases in the system.

## [2.8] Detection of plant diseases using Leaf patterns.

**A. Ranjith Ram, Vishnu S. [ 2015]**

We discuss the different methodologies for the detection of plant disease in this review paper. Studies show that relying on experts' pure nakedeye observation to diagnose and identify diseases can be time-consuming and expensive, particularly in rural areas and developing countries.

So, we present a solution based on quick, automated, cheap and precise image processing.

This paper discusses and describes the methods of image processing used to classify plant diseases. The primary plant disease identification strategies are: BPNN, SVM

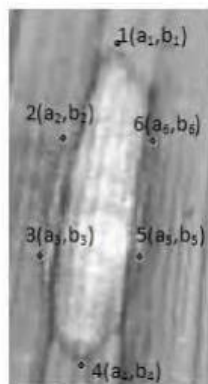
The analysis indicates that this technique of disease detection shows a good potential with the ability to detect diseases of the plant leaf and certain limitations. Therefore, in existing research, there is room for change.

## [2.9] Rice Leaf Disease Detection Using Chaos and Fractal Image Processing

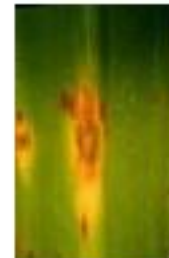
*V.Surendrababu, Dr.C.P.Sumathi, E.Umapathe [2017]*

In this paper, the specialists saw some significant research issues in bedlam and fractal measurements in picture handling for rice leaf illness location. Due to self-likeness, the fractal for the leaf example of the sick leaf at the last stage will have a similar example for each type of rice leaf ailment during the underlying stage. The reproduction of the fractal would in this way permit the malady to be distinguished early. The procedures of reproducing the fractal from impact malady influenced leaf designs through fractal measurement and tumult hypothesis to be done similarly as different sorts of leaf infections.

Vertices	Generated points
5	$(x_1, y_1)$
3	$(x_2, y_2)$
2	$(x_3, y_3)$
3	$(x_4, y_4)$
5	$(x_5, y_5)$
4	$(x_6, y_6)$
2	$(x_7, y_7)$



Normal Leaf



Diseased Leaves



Figure 2.9 Leaf Disease classification based on feature points

Along these lines for each kind of infection that has been distinguished, there is characterization. Besides, the proposed strategies and results can be of incredible intrigue and give setting to future



research in the field of mayhem and fractal measurement of picture preparing for different applications.

## **[2.10] Detection of the affected area of the plant leaf using image processing technology**

*Vijai Singh, A K Misra, Varsha [2014]*

The image processing approach is used in this paper to achieve accuracy in plant leaf disease detection. In this article, the method of image processing is used to detect plant leaf disease accuracy. Citrus leaf disease detection is conducted using the clustering algorithm k-mean.

The percentage calculated using region and image properties for the affected area of the leaf. That's the advantage to the farmer, reducing the exact amount of pesticides needed to cover the area being infected. There are three critical stages in the experimental work; in the initial stage, the image acquisition through which digital camera records the actual plant sample image.

The image is subjected to image preprocessing tools in the next stage, reducing the size and complexity of the image.

For the segmentation process, the precise digital information is used that separates the diseased portion of the leaf samples. Eventually, the region of the segmented portion of the leaf is determined using algorithms for image processing.

**Photo processing:** The acquisition of photographs is the first stage involving a digital camera to capture a photo. The images are taken in the garden or fields from various lighting conditions. For current work, a total of 100 images of citrus-diseased leaf were collected.

The captured citrus diseased leaf image is shown in Fig.1(a) Preprocessing: eliminate the undesirable distortion of these images in the preprocessing of the image.

It is reshaped to the size of 256 pixels and enhances the contrast of some images.

Fig. shows a resized citrus leaf image and an enhanced contrast image, resulting in the image being used for further operations such as creating clusters in the segmentation process.

The best way to overcome growing plant diseases through digital images combined with image processing, pattern recognition, classification techniques.

### **Image Processing**

The most ideal approach to defeat developing plant illnesses through computerized pictures joined with picture handling, design acknowledgment, order techniques. The picture handling programming in agribusiness bolsters the cultivating area and assists ranchers with expanding horticulture's proficiency

and efficiency. The significant kinds of contagious, viral, and bacterial-based plant illnesses. Seen some broad conditions are yellow and earthy colored spots, early and late burning, and so on. Investigating and identifying the various ailments of plant leaf through accessible catching gadgets and computerized picture preparing and getting an upgraded picture or giving the data.

The database contains an enormous array of pictures of safe and polluted leaves in either indigenous or foreign repositories. Photographs are in love with a traditional camera.

An image generally constitutes of three channels RGB which are red green and blue.

Up to the current finish, we tend to perform a preprocessing step wherever each image in our dataset is resized to 60 or 60 pixels and regenerates to a gray scale.

### **Deep-Learning-Based Classification**

The neural network is made up of several layered neurons. The neurons are interconnected in adjacent layers. These neurons learn to convert inputs to corresponding outputs (labels) (pre-extracted and pre-processed features).

The Conventional Neural Network comprises of different layers working together on a set of data to find some patterns and thus create a learning map of some unique features in the process. This type of neural network is trained multiple times over thousands of piles of data. CNN consists of three main parts that are layers to be defined, pooled and totally linked. The convolution and pooling layers' act as feature extractors from the images of the input while the fully connected layer acts as the classifier.

Determination is the primary objective of removing features from each source image automatically. The pooling layer reduces the mobility of these features.

Fully attached to the SoftMax activation mechanism at the end of the template, the trained high-level features are used to group the input objects into predefined classes.

**2.11 Tabular comparison of techniques used in the past recent years [Table 2.1]**

<b>Year</b>	<b>Contributors / Authors</b>	<b>Technique Used</b>	<b>Accuracy</b>
<b>2019</b>	Raj Priyanka, Dr. Neha Mangla,	Canny edge detection algorithm and CBIR	<b>92.3%</b>
<b>2019</b>	Jiangsheng Gui , Mor Mbaye	CNN Network	<b>90.5%</b>
<b>2018</b>	Jihen Amara, Bassem Bouaziz, Alsayed Algerawy	Deep learning	<b>83%</b>
<b>2018</b>	Rafel C.Gonzalez and Richards E.Woods	Classification of image processing techniques	<b>89%</b>
<b>2017</b>	M. Akila, P. Deepan(Assistant Professor)	Histogram approach	<b>71.2%</b>
<b>2016</b>	Dr. Neha Mangla, Priyanka B Raj, Soumya G Hegde, Pooja R	Canny edge detection algorithm and CBIR	<b>92.3%</b>
<b>2016</b>	Sharayu S. Tambe , Gulve Pranita	PNN and extract color and texture feature	<b>86%</b>
<b>2015</b>	Vishnu S, A. Ranjith Ram	BPNN, SVM to detect plant leaf disease	<b>73%</b>
<b>2015</b>	V.Surendrababu, Dr.C.P.Sumathi, E.Umapathy	Chaos and fractal dimension	<b>81.2%</b>
<b>2014</b>	Vijai Singh, Varsha, A K Misra	Capture, resize and enhancing contrast	<b>71.6%</b>

# CHAPTER 3

## SYSTEM DEVELOPMENT

### 3.1 Dataset Collection

The first priority when training a model is to create an accurate dataset. Our compilation consists of various images of diverse flora. In the development of this, we prioritized some of the commercial/cash grain crops, cereal crops, vegetable crops and fruit plants such as corn maize and potato. Infected bacterial or viral leaves, healthy leaves all of them were collected from diverse range of sources like images download from the Internet, or simply taking pictures using any camera device.

This specific model is optimized for corn(maize) plant.

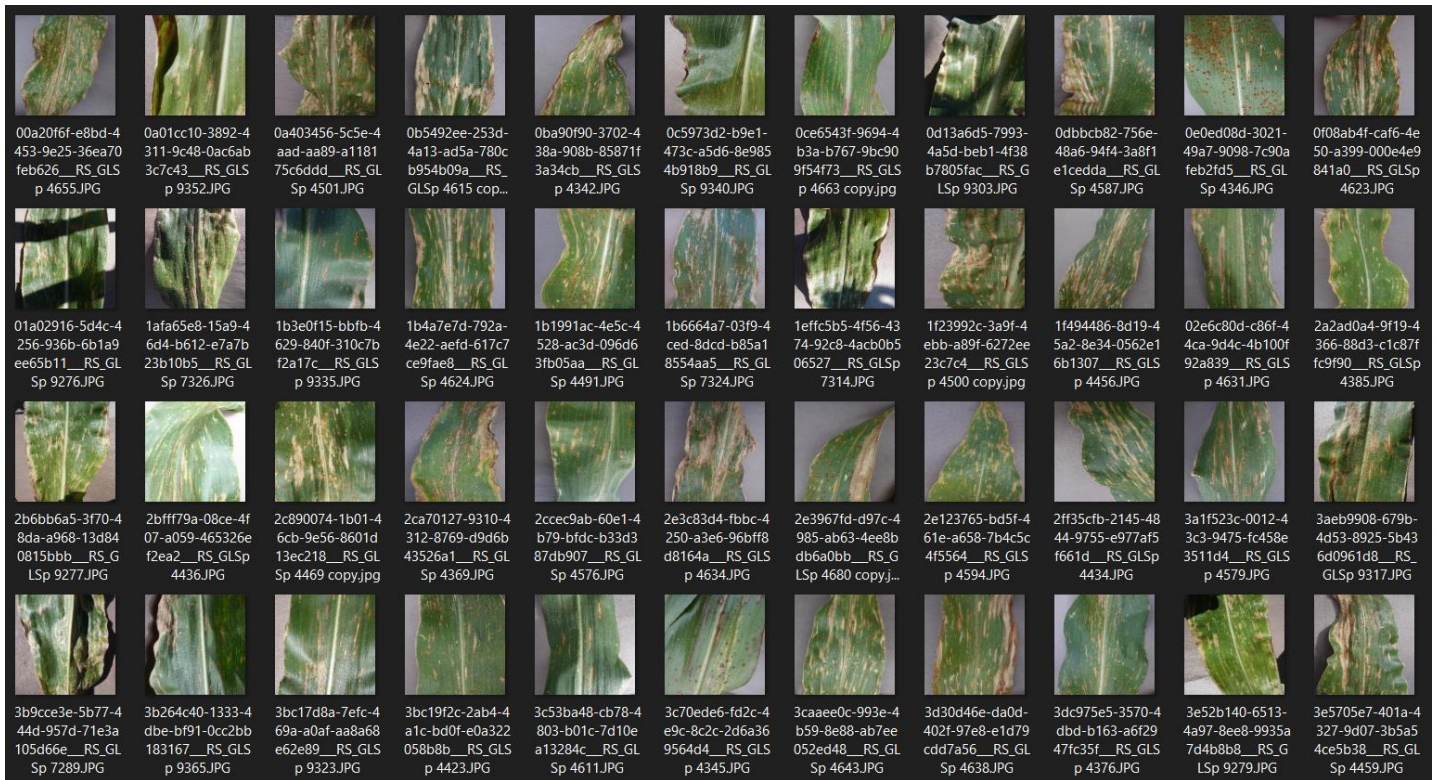


Figure 3.1 Infected leaves samples

### 3.1.1 Creating the dataset:

Here we are going to use the collected Plant leaf Dataset, which contains 80,000 RGB images in 8 categories. We will use 65000 for training and the rest 15000 for testing purposes. At first, we will import and then load the data to feed it into the CNN network.

- Importing the libraries and loading the data:

```
In [1]: import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
import pickle

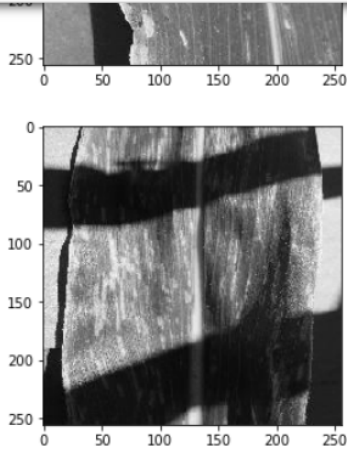
In [2]: DATADIR="C:/Users/pgmsc/Desktop/ds/Data"
CATAGORIES = ["Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot", "Corn_(maize)___Common_rust_", "Corn_(maize)___healthy", "Corn_
```

*Figure 3.2 Installing & Importing required dataset*

Here we have loaded the data and mapped the images into various classes for the classification.

- Pre-processing the data for computation:  
We scaled and resized the images thus creating a workable training dataset. The loaded images are then displayed with applied transformations on them.

```
In [4]: for categories in CATAGORIES:
        path = os.path.join(DATADIR , categories) # path for corn leaves
        count = 0
        for img in os.listdir(path):
            img_array = cv2.imread(os.path.join(path,img),cv2.IMREAD_GRAYSCALE)
            plt.imshow(img_array , cmap="gray")
            plt.show()
            count = count +1
            if count == 10:
                break
        break
```



```
In [5]: img_array.shape
```

```
Out[5]: (256, 256)
```

*Figure 3.3 Images loaded are displayed at random*

To make the computation more discernible we convert the three channel RGB pictures into Grayscale. Here we observe that the images loaded from the training set have size 256\*256. It shows that the pixel values will fall in the range of 0 to 255.

```
In [8]: def create_data():
        for category in CATAGORIES:
            path = os.path.join(DATADIR , category) # path for corn leaves
            class_num = CATAGORIES.index(category)
            for img in os.listdir(path):
                try:
                    img_array = cv2.imread(os.path.join(path,img),cv2.IMREAD_GRAYSCALE)
                    new_array = cv2.resize(img_array , (IMG_SIZE,IMG_SIZE))
                    training_data.append([new_array,class_num])
                except Exception as e:
                    print(str(e))
        create_data()
```

```
In [9]: len(training_data)
```

```
Out[9]: 3852
```

*Figure 3.3.1 Converting the images to workable data*

```
In [16]: for features , label in training_data:
          X.append(features)
          y.append(label)

X = np.array(X).reshape(-1,IMG_SIZE,IMG_SIZE,1)
```

```
In [17]: np.shape(X)
```

```
Out[17]: (3852, 200, 200, 1)
```

```
In [18]: np.shape(y)
```

```
Out[18]: (3852,)
```

```
In [19]: #dataset ready to be worked upon
```

```
In [ ]:
```

```
In [20]: X= X/255.0
```

```
In [21]: pickle_out = open("X_corn.pickle","wb")
          pickle.dump(X,pickle_out)
          pickle_out.close()

          pickle_out = open("y_corn.pickle","wb")
          pickle.dump(y,pickle_out)
          pickle_out.close()
```

*Figure 3.4 Applying transformations and filters*

Here we displayed the images with the applied transformations for the computation. Now we scale the images into the range 0-1 for further feeding into the neural network.

### 3.2 Building the Neural Network Architecture:

In order to create the model, we will be feeding our system with the data set optimized for this computation. This dataset is split into two distinguishable parts; feature set and label set. The feature set consists of the array of images and the label tags consists of the class for the respective image. Building the neural network requires configuring the layers of the model, then compiling the model. In this deep learning model, we will chain together these layers multiple times for implanting the prediction.

Now we import the required libraries for building the model as the input image is reshaped into a matrix of 200\*200\*1

```
In [1]: import pickle

In [2]: import tensorflow as tf

In [3]: from tensorflow.keras.models import Sequential
        from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPooling2D

In [4]: pickle_in = open("X_corn.pickle", "rb")
        X = pickle.load(pickle_in)
        pickle_in.close()

        pickle_in = open("y_corn.pickle", "rb")
        y = pickle.load(pickle_in)
        pickle_in.close()

In [5]: import numpy as np

In [6]: np.shape(X)

Out[6]: (3852, 200, 200, 1)

In [7]: np.shape(y)

Out[7]: (3852,)

In [8]: X.shape[1:]

Out[8]: (200, 200, 1)

In [9]: y1 = np.array(y)
```

*Figure 3.5 Rechecking for the transformed dimensions*

Now we defined the convolution layer with 64 filters and a stride of 3 and activation function set as "relu" followed by a have pooling layer i.e. "max pool" layer. The same process is repeated with different parameters. After that we are using an activation layer as 'sigmoid'. We are using the "adam" optimizer to optimize the model and thus, reduced the overall loss.



```

In [10]: model= Sequential()
model.add(Conv2D(64,(3,3),input_shape = X.shape[1:]))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(64,(3,3)))
model.add(Activation("relu"))
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dense(64))

model.add(Dense(4))
model.add(Activation("sigmoid"))

model.compile(loss="sparse_categorical_crossentropy",
              optimizer = "adam",
              metrics = ["accuracy"])

history = model.fit(X, y1, batch_size=32 , epochs=7, validation_split= 0.1)

```

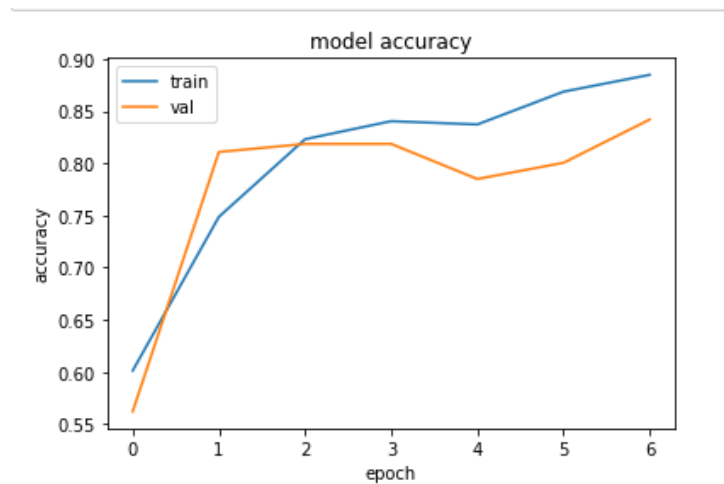
*Figure 3.6 Building the CNN model*

Consequently, in order to start the training of our model, model.fit function method is used; as the name suggests this method literally "fits" the model onto the given training data. Here we developed three models thereby increasing prediction accuracy with each iteration.

The first model will run for 7 iterations:

```
Train on 3466 samples, validate on 386 samples
Epoch 1/7
3466/3466 [=====] - 150s 43ms/sample - loss: 0.8157 - accuracy: 0.6013 - val_loss: 0.7057 - val_accuracy: 0.5622
Epoch 2/7
3466/3466 [=====] - 145s 42ms/sample - loss: 0.5656 - accuracy: 0.7487 - val_loss: 0.4664 - val_accuracy: 0.8109
Epoch 3/7
3466/3466 [=====] - 144s 41ms/sample - loss: 0.4106 - accuracy: 0.8231 - val_loss: 0.4017 - val_accuracy: 0.8187
Epoch 4/7
3466/3466 [=====] - 146s 42ms/sample - loss: 0.3764 - accuracy: 0.8405 - val_loss: 0.3838 - val_accuracy: 0.8187
Epoch 5/7
3466/3466 [=====] - 142s 41ms/sample - loss: 0.3649 - accuracy: 0.8373 - val_loss: 0.4613 - val_accuracy: 0.7850
Epoch 6/7
3466/3466 [=====] - 144s 42ms/sample - loss: 0.3070 - accuracy: 0.8687 - val_loss: 0.4500 - val_accuracy: 0.8005
Epoch 7/7
3466/3466 [=====] - 143s 41ms/sample - loss: 0.2624 - accuracy: 0.8849 - val_loss: 0.3379 - val_accuracy: 0.8420
```

*Figure 3.7 Model In training phase*



*Figure 3.8 Model 0 Training vs Validation Accuracy*

After the model is done training the loss and prediction accuracy metrics are also displayed. This model achieves an accuracy of 88% and 84% on validation data. All this data is then visualized into a graphical from thus making it easy to compare different models. Now, if you train your neural network for more epochs or change the activation function, you might get a different result that might have better accuracy.

Now to improve the accuracy of these models we will adjust their hyperparameters and tweak with the overall structure the neural network.

## Model 1

```
Train on 3466 samples, validate on 386 samples
Epoch 1/7
3466/3466 [=====] - 176s 51ms/sample - loss: 0.6228 - accuracy: 0.7294 - val_loss: 0.3635 - val_accuracy: 0.8290
Epoch 2/7
3466/3466 [=====] - 147s 42ms/sample - loss: 0.4053 - accuracy: 0.8266 - val_loss: 0.3136 - val_accuracy: 0.8420
Epoch 3/7
3466/3466 [=====] - 145s 42ms/sample - loss: 0.3444 - accuracy: 0.8439 - val_loss: 0.3058 - val_accuracy: 0.8394
Epoch 4/7
3466/3466 [=====] - 145s 42ms/sample - loss: 0.3111 - accuracy: 0.8569 - val_loss: 0.3676 - val_accuracy: 0.8290
Epoch 5/7
3466/3466 [=====] - 146s 42ms/sample - loss: 0.2819 - accuracy: 0.8774 - val_loss: 0.4689 - val_accuracy: 0.8394
Epoch 6/7
3466/3466 [=====] - 146s 42ms/sample - loss: 0.2150 - accuracy: 0.9013 - val_loss: 0.3256 - val_accuracy: 0.8575
Epoch 7/7
3466/3466 [=====] - 144s 42ms/sample - loss: 0.2031 - accuracy: 0.9140 - val_loss: 0.2970 - val_accuracy: 0.8756
```

```
: model.save("model1.h5")
```

Figure 3.9 Model 1 Training

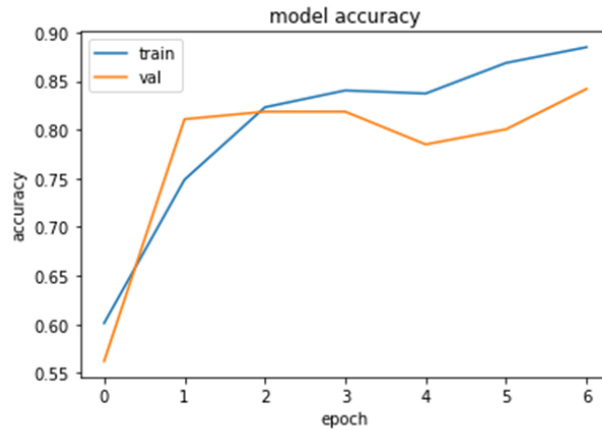
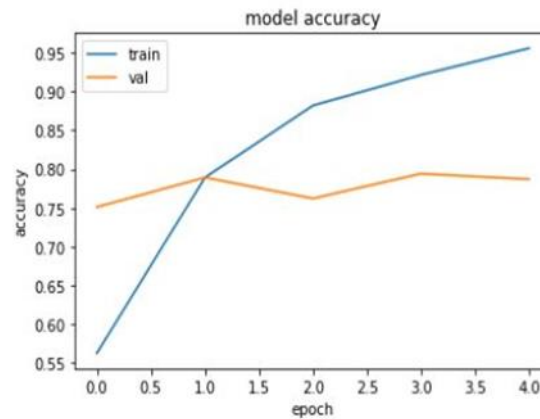


Figure 3.10 Model 1 Training vs Validation Accuracy

## Model 2

```
Epoch 1/5  
16344/16344 [=====] - 1577s 97ms/sample - loss: 1.2322 - acc: 0.5624 - val_loss: 0.7  
560 - val_acc: 0.7511  
Epoch 2/5  
16344/16344 [=====] - 932s 57ms/sample - loss: 0.6051 - acc: 0.7891 - val_loss: 0.63  
97 - val_acc: 0.7891  
Epoch 3/5  
16344/16344 [=====] - 870s 53ms/sample - loss: 0.3398 - acc: 0.8820 - val_loss: 0.80  
50 - val_acc: 0.7621  
Epoch 4/5  
16344/16344 [=====] - 866s 53ms/sample - loss: 0.2155 - acc: 0.9216 - val_loss: 0.74  
15 - val_acc: 0.7941  
Epoch 5/5  
16344/16344 [=====] - 817s 50ms/sample - loss: 0.1188 - acc: 0.9560 - val_loss: 0.90  
17 - val_acc: 0.7869
```

*Figure 3.11 Model 2 Training*



*Figure 3.12 Model 2 Training vs Validation Accuracy*

It turns out that the accuracy on the test dataset is a little less than the accuracy on the training dataset. This gap between training accuracy and test accuracy represents overfitting. Overfitting happens when a machine learning model performs worse on new, previously unseen inputs than it does on the training data. An overfitted model "memorizes" the noise and details in the training dataset to a point where it negatively impacts the performance of the model on the new data.

## Model 3

```
Train on 3466 samples, validate on 386 samples
Epoch 1/15
3466/3466 [=====] - 181s 52ms/sample - loss: 0.7544 - accuracy: 0.6428 - val_loss: 0.4968 - val_accuracy: 0.7513
Epoch 2/15
3466/3466 [=====] - 148s 43ms/sample - loss: 0.4329 - accuracy: 0.8113 - val_loss: 0.3479 - val_accuracy: 0.8290
Epoch 3/15
3466/3466 [=====] - 148s 43ms/sample - loss: 0.3958 - accuracy: 0.8099 - val_loss: 0.3605 - val_accuracy: 0.8316
Epoch 4/15
3466/3466 [=====] - 146s 42ms/sample - loss: 0.3785 - accuracy: 0.8257 - val_loss: 0.4049 - val_accuracy: 0.7772
Epoch 5/15
3466/3466 [=====] - 145s 42ms/sample - loss: 0.3576 - accuracy: 0.8329 - val_loss: 0.3145 - val_accuracy: 0.8497
Epoch 6/15
3466/3466 [=====] - 145s 42ms/sample - loss: 0.3268 - accuracy: 0.8451 - val_loss: 0.4045 - val_accuracy: 0.8238
Epoch 7/15
3466/3466 [=====] - 145s 42ms/sample - loss: 0.3263 - accuracy: 0.8430 - val_loss: 0.3270 - val_accuracy: 0.8420
Epoch 8/15
3466/3466 [=====] - 148s 43ms/sample - loss: 0.2585 - accuracy: 0.8635 - val_loss: 0.3425 - val_accuracy: 0.8342
Epoch 9/15
3466/3466 [=====] - 150s 43ms/sample - loss: 0.2365 - accuracy: 0.8736 - val_loss: 0.3266 - val_accuracy: 0.8446
Epoch 10/15
3466/3466 [=====] - 150s 43ms/sample - loss: 0.2068 - accuracy: 0.8898 - val_loss: 0.3130 - val_accuracy: 0.8420
Epoch 11/15
3466/3466 [=====] - 150s 43ms/sample - loss: 0.1948 - accuracy: 0.8964 - val_loss: 0.3308 - val_accuracy: 0.8472
Epoch 12/15
3466/3466 [=====] - 149s 43ms/sample - loss: 0.2096 - accuracy: 0.8857 - val_loss: 0.4561 - val_accuracy: 0.8523
Epoch 13/15
3466/3466 [=====] - 147s 43ms/sample - loss: 0.1777 - accuracy: 0.9155 - val_loss: 0.4021 - val_accuracy: 0.8549
Epoch 14/15
3466/3466 [=====] - 148s 43ms/sample - loss: 0.1438 - accuracy: 0.9331 - val_loss: 0.4643 - val_accuracy: 0.8601
Epoch 15/15
3466/3466 [=====] - 162s 47ms/sample - loss: 0.1230 - accuracy: 0.9383 - val_loss: 0.4607 - val_accuracy: 0.8653
```

*Figure 3.13 Model 3 Training*

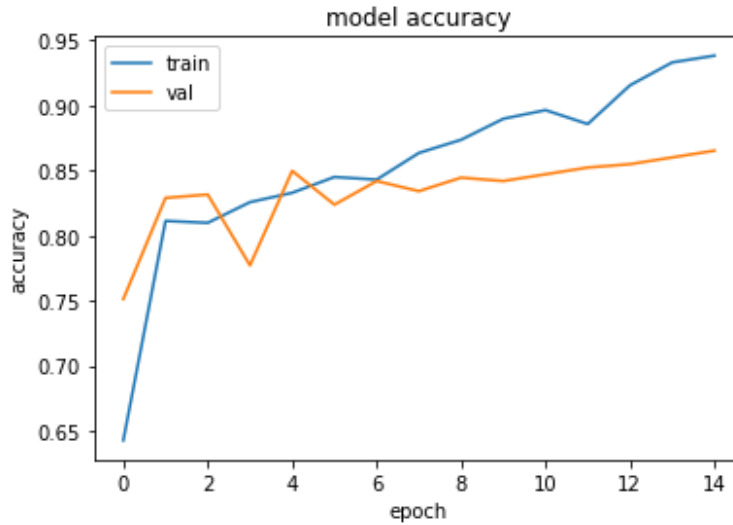


Figure Model 3.14 Training vs Validation Accuracy

**Comparison Table:** Models developed can be compared one to another with their metrics of final results in training in validation.

	Training Precision	Validation Accuracy	Overfitting/ Underfitting
Model 1	53%	50%	---
Model 2	95.2%	77%	Found
Model 3	93.8%	86.5%	Resolved

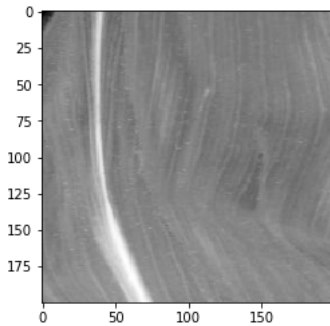
Table 3.1 Model Comparison Table

### 3.3 Making predictions with the trained model

Now that our model is trained, we can start making random predictions by feeding it some raw images. Generally, the CNN network gives out linear outputs known as logits which can't be interpreted. We have added a softmax layer at the end which converts the unfathomable logits to more specific probabilities values. Here, the model has predicted some probability score for each of the categorical label from various features.

```
In [15]: plt.imshow(X[1].reshape(200,200), cmap="gray")
```

```
Out[15]: <matplotlib.image.AxesImage at 0x2b9249b69b0>
```



```
In [16]: y[1]
```

```
Out[16]: 2
```

```
In [17]: model.predict(X[1].reshape(-1,200,200,1))
```

```
Out[17]: array([[1.6391277e-06, 5.9604645e-08, 6.8719000e-01, 1.3947487e-04]],  
              dtype=float32)
```

```
In [18]: #works like a charm
```

*Figure 3.15 Prediction values*

The result is displayed in terms of arrays with each element showing a probabilistic score. This score signifies the certainty in the prediction made into the labels. You can see which label has the highest confidence value. Here the trained model is 87% sure that it is a healthy leaf. It can be observed that the classification done by the model into the label is absolutely correct. This means that its working as expected.

# CHAPTER 4

## PERFORMANCE ANALYSIS

### 4.1 Running the saved model:

Here we have developed a mini user interface using Tkinter which lets the user select a custom specific image for the computation and thus making predictions through it.

```
In [1]: import cv2

In [2]: import numpy as np
import matplotlib.pyplot as plt

In [3]: from tensorflow.keras.models import load_model

In [4]: model_number = input("Enter Model number : ")
Enter Model number : 1

In [5]: model_name = "model" + model_number+".h5"

In [6]: model_name
Out[6]: 'model1.h5'

In [7]: test_model = load_model(model_name)

In [8]: import tkinter as tk

In [9]: from tkinter import filedialog

In [10]: root = tk.Tk()
#root.withdraw()

In [11]: path = filedialog.askopenfilename(title = "Choose image")
```

*Figure 4.1 Importing libraries*



Here we may select the model number to be used for prediction.

Enter Model number :

Figure 4.2 Prompt to select the model number

For taking the input image to classify the disease we make use of tkinter. Here we can select an image for further processing.

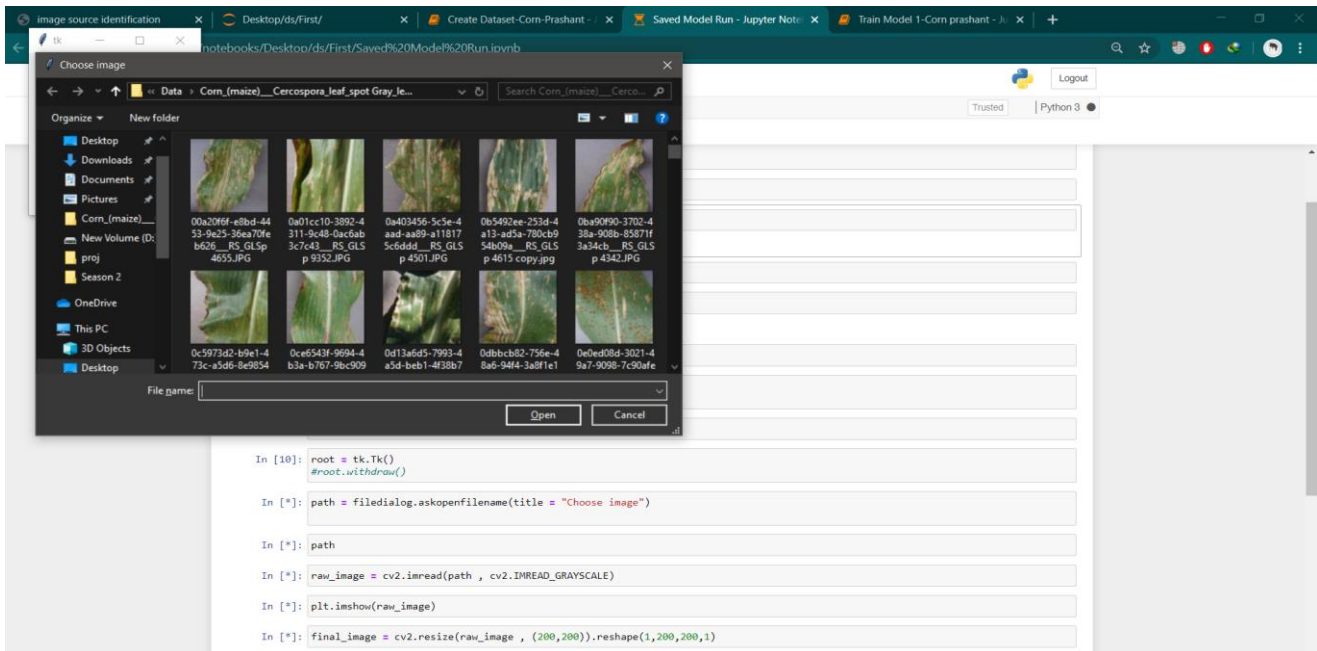
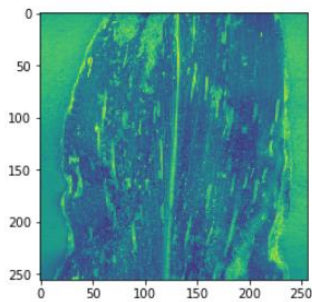


Figure 4.3 Dialog box to select image

## 4.2 Filters applied on the image:

Out[14]: <matplotlib.image.AxesImage at 0x253abd21470>



```
In [15]: final_image = cv2.resize(raw_image , (200,200)).reshape(1,200,200,1)
```

```
In [16]: final_image = final_image/255.0
```

```
In [17]: final_image.shape
```

```
Out[17]: (1, 200, 200, 1)
```

```
In [18]: t=test_model.predict_proba(final_image)
print(t)
number=['Cercospora leaf spot/ Gray leaf spot','Common rust','Healthy','Northern Leaf Blight']
index=np.argsort(t[0,:])
print('Likely Plant Disease is :',number[index[3]],"--Probability ",t[0,index[3]])
```

```
[[9.9784696e-01 5.9604645e-08 1.1920929e-07 9.7488844e-01]]
Likely Plant Disease is : Cercospora leaf spot/ Gray leaf spot --Probability 0.99784696
```

*Figure 4.4 Prediction made by the model with 99% probability*

It can be observed that the likely disease for the plant is specified i.e. Cercospora leaf spot with 0.99784696 probability.

### 4.3 Output dialog box:

```
image = cv2.resize(raw_image , (200,200)).reshape(1,200,200,1)

image = final_image/255.0

image.shape
(1, 200, 200, 1)

model.predict_proba(final_image)
[0.98499999 0.01499999 0.00099999 0.00099999]

print('The most likely Plant Disease is :', number[index[3]], "--Probability ", t[0, index[3]])

[0.98499999 0.01499999 0.00099999 0.00099999]
The most likely Plant Disease is : Cercospora leaf spot/ Gray leaf spot --Probability 0.15953916
```



Figure 4.5 Result shown in the pop-up window

Here we can see that the model and the interface are working as expected. The classified label disease is shown in the window popup along with the prediction accuracy.

## 4.4 Testing with different cases

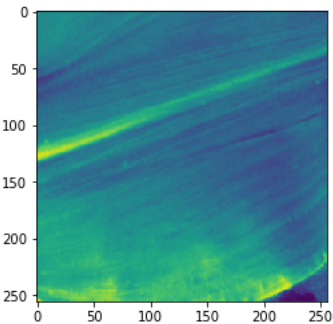
Now we can feed diverse random inputs into the model to check its reliability. We can tweak around with different categories for testing the model in detail.

```
Out[12]: 'C:/Users/pgmsc/Desktop/ds/Data/Corn_(maize)___healthy/0d27e784-5661-4474-9806-3453c7ef3bf5__R.S_HL_5511_copy.jpg'
```

```
In [13]: raw_image = cv2.imread(path , cv2.IMREAD_GRAYSCALE)
```

```
In [14]: plt.imshow(raw_image)
```

```
Out[14]: <matplotlib.image.AxesImage at 0x1d784501470>
```



```
In [15]: final_image = cv2.resize(raw_image , (200,200)).reshape(1,200,200,1)
```

```
In [16]: final_image = final_image/255.0
```

```
In [17]: final_image.shape
```

```
Out[17]: (1, 200, 200, 1)
```

```
In [18]: t=test_model.predict_proba(final_image)
print(t)
number=['Cercospora leaf spot/ Gray leaf spot','Common rust','Healthy','Northern Leaf Blight']
index=np.argsort(t[0,:])
print('Likely Plant Disease is :',number[index[3]],"--Probability ",t[0,index[3]])
```

```
[[1.4246106e-03 1.4996529e-04 8.9727485e-01 5.1371753e-03]]
Likely Plant Disease is : Healthy --Probability 0.89727485
```

*Figure 4.6 Healthy Leaf with Probability 89%*

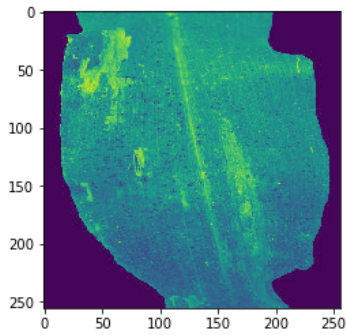
We can see that the prediction is accurate with significant probability for reliability.

```
Out[12]: 'C:/Users/pgmsc/Desktop/ds/Data/Corn_(maize)___Common_rust_/RS_Rust_1565.JPG'
```

```
In [13]: raw_image = cv2.imread(path , cv2.IMREAD_GRAYSCALE)
```

```
In [14]: plt.imshow(raw_image)
```

```
Out[14]: <matplotlib.image.AxesImage at 0x228003104a8>
```



```
In [15]: final_image = cv2.resize(raw_image , (200,200)).reshape(1,200,200,1)
```

```
In [16]: final_image = final_image/255.0
```

```
In [17]: final_image.shape
```

```
Out[17]: (1, 200, 200, 1)
```

```
In [18]: t=test_model.predict_proba(final_image)
print(t)
number=['Cercospora leaf spot/ Gray leaf spot','Common rust','Healthy','Northern Leaf Blight']
index=np.argsort(t[0,:])
print('Likely Plant Disease is :',number[index[3]],"--Probability ",t[0,index[3]])
```

```
[[0. 1. 0. 0.]]
Likely Plant Disease is : Common rust --Probability 1.0
```

*Figure 4.7 Common rust with 100% probability*

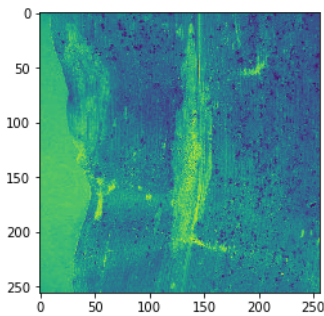
Works correctly.

```
Out[12]: 'C:/Users/pgmsc/Desktop/ds/Data/Corn_(maize)___Northern_Leaf_Blight/0abbec2f-123f-4ae1-a9b7-8babbe8d0e89___RS_NLB_3685.JPG'
```

```
In [13]: raw_image = cv2.imread(path , cv2.IMREAD_GRAYSCALE)
```

```
In [14]: plt.imshow(raw_image)
```

```
Out[14]: <matplotlib.image.AxesImage at 0x16f03b50470>
```



```
In [15]: final_image = cv2.resize(raw_image , (200,200)).reshape(1,200,200,1)
```

```
In [16]: final_image = final_image/255.0
```

```
In [17]: final_image.shape
```

```
Out[17]: (1, 200, 200, 1)
```

```
In [18]: t=test_model.predict_proba(final_image)
print(t)
number=['Cercospora leaf spot/ Gray leaf spot', 'Common rust', 'Healthy', 'Northern Leaf Blight']
index=np.argsort(t[0,:])
print('Likely Plant Disease is :',number[index[3]],"--Probability ",t[0,index[3]])
```

```
[[5.0333142e-03 0.0000000e+00 5.3524971e-05 9.6420246e-01]]
Likely Plant Disease is : Northern Leaf Blight --Probability 0.96420246
```

*Figure 4.8 Northern Leaf Blight with 96% probability*

# CHAPTER 5

## CONCLUSIONS

### 5.1 Conclusions of the work

In our attempt at this project we were able to develop various convolutional neural networks which give satisfactory results with upmost precision. There were various approaches in the recent years of image segmentation and classification to detect the infectious plant leaves. Here we took the less explored method of convolutional neural networks. The model was able to detect and classify plant leaves on the basis of diseases. Model was able to reach an accuracy of 94% giving precise results predominantly. The complete course of development was discussed from creating dataset of the images to model predictions and interface.

- Biologically leaf diseases sometimes exhibits similar symptoms which becomes difficult to fathom for artificial intelligence. This essentially means that there is a large scope for further research and development in this field.
- According to recent research complexity of plant diseases modifies drastically with the time, therefore thus we need a model which is adaptive to these types of conditions.
- The model shows a great potential backed up with accurate results.
- This type of automated system can prove to a great help to the agribusiness in terms of economic and environmental aspects.

## 5.2 Future Scope

- There is always a scope for improvement in the prediction accuracy by running the system through various versions and models of the CNN architecture. Moreover, it can also expand to a wider variety of crops giving it more fuzzy problems to work with.
- A standalone web application or mobile app can be developed which implements the same technique and provides the accessibility to the end user.
- As the field of research is ever growing with modern advancements in biotechnology and robotic engineering this type of system can be combined with monitoring robots and drones which may analyze the whole field crop with continuous and consistent analysis and hence notifying for any potential hazards and trigger warnings beforehand.



## REFERENCES

- [1] Hiroya Kondou, Hatuyoshi Kitamura, Yutaka Nishikawa, Yoshitaka Motonaga and Atsushi Hashimoto, Shape Evaluation by Digital Camera for Grape Leaf.
- [2] Prasad Babu M.S. and Srinivasa Rao B. (2007) Leaves Recognition Using Back Propagation Neural Network-Advice For Pest and Disease Control On Crops.
- [3] Maliappis M.T., Ferentinos K.P., Passam H.C. and Sideridis A.B. (2008) World Journal of Agricultural Sciences 4(5), 640-647.
- [4] Ryszard S. Choras (2007) International Journal Of Biology And Biomedical Engineering,1(1), 6-16.
- [5] Suhasini P.S., Sri Rama Krishna K., Murali Krishna I.V. (2009) Journal Of Theoretical And Applied Information Technology, 6(1), 116-122.
- [6] Zhi-Chun Huang, Patrick P. Chan, Wing W.Y. N.G., Daniel S. Yeung (2010) ninth international conference on machine learning and cybernetics, 11-14, 719-724
- [7] Huang J., Kumar S.R., Mitra M., Zhu W.J. and Zabih R. (1997) IEEE Int. Conf. Computer Vision and Pattern Recognition, 762-768.
- [8] Del Bimbo A., Mugnaini M., Pala P. and Turco F. Picasso, (1997) 2nd International Conference on Visual Information Systems, 125-131.
- [9] Jau-Ling Shih & Ling-Hwei Chen, color image retrieval based on primitives of color moments.
- [10] Priti Maheswary, Namita Srivastav (2008) international conference on computer and electrical engineering, 821-824.
- [11] Mona Sharma, Sameer Singh (2001) Seventh Australian and New Zealand Intelligent Information Systems Conference, 117-121

[12] Xinhong Zhang, Fan Zhang (2008) Congress on Image and Signal Processing, IEEE computer society, 773-776.

[13] Hui Yu, Mingjing Li, Hong-Jiang Zhang, Jufu Feng (2003) International Conference on Image Processing.

# Plagiarism Report

## Agronomics

### ORIGINALITY REPORT

<b>7</b> %	<b>4</b> %	<b>2</b> %	<b>6</b> %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

### PRIMARY SOURCES

<b>1</b>	<a href="http://www.tensorflow.org">www.tensorflow.org</a> Internet Source	<b>1</b> %
<b>2</b>	<a href="http://www.ijert.org">www.ijert.org</a> Internet Source	<b>1</b> %
<b>3</b>	Submitted to University of Northampton Student Paper	<b>&lt;1</b> %
<b>4</b>	Submitted to Chandigarh University Student Paper	<b>&lt;1</b> %
<b>5</b>	<a href="http://en.wikipedia.org">en.wikipedia.org</a> Internet Source	<b>&lt;1</b> %
<b>6</b>	Submitted to Cranfield University Student Paper	<b>&lt;1</b> %
<b>7</b>	Submitted to Visvesvaraya Technological University, Belagavi Student Paper	<b>&lt;1</b> %
<b>8</b>	Submitted to Kwame Nkrumah University of Science and Technology Student Paper	<b>&lt;1</b> %

# Agronomics

*by* Prashant G61

---

**Submission date:** 26-May-2020 10:14PM (UTC+0530)

**Submission ID:** 1332236451

**File name:** plag\_check\_g61.pdf (2.51M)

**Word count:** 7261

**Character count:** 38492

**JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT**

**PLAGIARISM VERIFICATION REPORT**

Date: 15/07/2020



Type of Document (Tick):  PhD Thesis  M.Tech Dissertation/ Report  B.Tech Project Report  Paper

Name: Prashant Gautam Department: CSE Enrolment No. 161379

Contact No. 9418976942 E-mail pgmscr7@gmail.com

Name of the Supervisor: Mr. Surjeet Singh

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): Deep Neural Networks

Based Detection of Plant Leaf Diseases By Image Classification

**UNDERTAKING**

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

**Complete Thesis/Report Pages Detail:**

- Total No. of Pages = 60
- Total No. of Preliminary pages = 11
- Total No. of pages accommodate bibliography/references = 2



**(Signature of Student)**

**FOR DEPARTMENT USE**

We have checked the thesis/report as per norms and found **Similarity Index** at 7 ..... (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.



**(Signature of Guide/Supervisor)**

**Signature of HOD**

**FOR LRC USE**

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	<ul style="list-style-type: none"><li>• All Preliminary Pages</li><li>• Bibliography/Images/Quotes</li><li>• 14 Words String</li></ul>		Word Counts	
<b>Report Generated on</b>		<b>Submission ID</b>	Total Pages Scanned	
			File Size	

Checked by  
Name & Signature

Librarian

**Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at [plagcheck.juit@gmail.com](mailto:plagcheck.juit@gmail.com)**