

**PLANT DISEASE DETECTION USING MACHINE
LEARNING**
MAJOR PROJECT-II REPORT
SUBMITTED IN PARTIAL FULFILLMENT OF REQUIREMENT FOR
THE AWARD OF THE DEGREE OF
MASTER OF TECHNOLOGY
IN
SOFTWARE ENGINEERING

Submitted By -

HARKISHAN KAUSHIK

(2K20/SWE/09)

Under the supervision of

Dr. MANOJ KUMAR

(Professor & Head (Computer Center))



DEPARTMENT OF SOFTWARE ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering) Bawana Road, Delhi-110042

May 2022

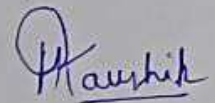
CANDIDATE'S DECLARATION

I hereby declare that the Major Project-II Report entitled "Plant Disease Detection using Machine learning" which is being submitted in partial fulfillment of the requirement for the award of a degree Master of Technology (Software Engineering) is a bonafide report. I have not submitted the matter embodied in this dissertation for the award of any other degree or diploma.

Place: Delhi

Date: 30/05/2022

(2k20/SWE/09)



Harkishan Kaushik

CERTIFICATE

This is to certify that Project Report entitled "Plant Disease Detection using Machine Learning" submitted by Harkishan Kaushik (2K20/SWE/09) in partial fulfillment of the requirement for the award of degree Master of Technology (Software Engineering) is a record of the original work carried out by him under my supervision.

Place: Delhi

Date: 30/07/2022



Dr. Manoj Kumar

(Professor & Head (Computer Center))

Department of Computer Science & Engineering

Delhi Technical University

(Formerly Delhi College of Engineering),

Shahbad Daulatpur,

Main Bawana Road, Delhi-110042.

ABSTRACT

With the ever-growing population of human beings and animals, the demand for the crops to sustain them has also been increasing. We have tried many methods to increase the yields of the crops such as the introduction of HYV (High Yielding Varieties), increasing the area under cultivation and many more. One such problem to the cultivation is the plant-based diseases caused by various pathogens. In general, cultivated plants are much more susceptible to plant diseases because of the proximity of similar plants (sometimes hundreds of kms) which causes the pathogens to spread more freely. Generally, plant diseases are classified into two categories- Infectious and Noninfectious. Infectious plant diseases are caused by organisms like viruses, bacteria, mycoplasma, fungus, etc. Non-Infectious plant diseases are caused mainly due to environmental factors such as unfavorable moisture, sunlight, temperature, toxic elements in the environment, insufficient minerals in the soil, etc. Proper diagnosis of such diseases is very important to stop the spread of the diseases and save the cultivation. A common way to diagnose such diseases is based on analyzing the appearance of the leaves, stem, and roots for any abnormalities like discoloration, overdevelopment or underdevelopment, misshape, holes, swollen, etc. As estimated by the UN about 80% of farmers in developing countries are smallholder farmers so the protection of plant diseases becomes very crucial in saving their livelihoods and achieving food security. Various organizations are engaged in providing information to the farmers online and through various programs. As the internet is widely available and the computation power of modern technology is capable enough, the diagnosis of plant disease with the help of machine learning and computer vision will be very helpful to the farmers. As more than half of the world population has access to smartphones containing cameras and internet capabilities along with good processing powers, so the development of image-based plant disease detection will be very helpful and beneficial for the smallholder farmers and society. In this project, we will look at the feasibility of developing such a platform by looking at the detection of plant diseases through machine learning techniques.

ACKNOWLEDGMENT

I take this opportunity to express my sincere gratitude to my project supervisor, **Dr. Manoj Kumar**, for providing the opportunity of carrying out this project and for being the guiding force behind this work. I am grateful for their constant supervision and support throughout the course of this project. Without his support, the timely completion of this project wouldn't have been possible. I would also like to acknowledge Delhi Technological University library and staff for providing the right academic resources and environment for this work to be carried out.

Finally, I express my sincere gratitude and thank my parents for their consistent care and support day in and day out during the entire course of the project. I further thank all my friends and fellow seniors who have helped me directly or indirectly for the successful completion of this project work.

Harkishan Kaushik

(2k20/SWE/09)

TABLE OF CONTENTS

	Page No.
Declaration	II
Certificate	III
Abstract	IV
Acknowledgement	V
Table of Contents	VI
List of Figures	VIII
List of Tables	IX
List of Abbreviations	X
1. Chapter 1: Introduction	1
1.1. Background	1
1.2. Current Scenario	1
1.3. ANN	2
1.4. Image Segmentation	3
1.4.1. Region-Based image segmentation	4
1.4.2. Edge Based Segmentation	5
1.4.3. Clustering-based image segmentation	6
1.4.4. CNN-based image segmentation	6
1.5. CNN	7
1.5.1. Convolution Layer	7
1.5.2. Pooling Layer	8
1.5.3. Activation Layer	8
1.5.4. Dropout Layer	9
1.5.5. Fully Connected Layer	9
1.6. Preprocessing by CNN	9
1.6.1. Architecture of a CNN	10

1.7.	Objective of work	11
1.8.	Scope of work	11
1.9.	Organization of Dissertation	11
2.	Chapter 2: Literature Survey	13
3.	Chapter 3: Methodology	20
3.1.	Our Approach	20
3.2.	Problem Statement	20
3.3.	Dataset Used	20
3.4.	Set Up	23
3.5.	Steps of Implementation	24
3.5.1.	Importing Libraries	24
3.5.2.	Loading Image Dataset	25
3.5.3.	Converting image to array	26
3.5.4.	Converting Image labels into Binary levels	27
3.5.5.	Splitting of Input	27
3.5.6.	Model Building	28
3.5.7.	Model Compiling	29
3.6.	Plotting the graph	30
3.6.1.	Chart Results	31
3.6.2.	Loss Graph Plot	32
3.6.3.	Calculation of model accuracy	32
4.	Chapter 4: Results	34
4.1.	Results and Analysis	34
4.1.1.	Comparison	34
4.1.2.	Action policy	35
4.1.3.	Results and accuracy	35
5.	Chapter 5: Conclusion	36
5.1.	Conclusion	36
5.2.	Future Scope	36
6.	References	38

LIST OF FIGURES

Figure 1.1 Architecture of CNN	10
Figure 2.1 Heat Maps generated	17
Figure 3.1 Structure of dataset	21
Fig 3.2 Samples of images in dataset	22
Fig 3.3 Examples of different phenotypes of tomato plants	22
Figure 3.4 Training vs. Validation Accuracy vs. No. of Epochs Graph	31
Figure 3.5 Training and validation Loss vs. Number of Epochs	32

LIST OF TABLES

4.1 Comparison and Analysis of different Image Segmentation Techniques	34
--	----

LIST OF ABBREVIATIONS

CNN- Convolutional Neural Network

ANN- Artificial Neural Network

R –CNN- Region based Convolutional Neural Network

GPU- Graphics Processing Unit

ReLu- Rectified Linear Unit

UAV- Unmanned Aerial Vehicle

CC- Connected Component

MRI- Magnetic Resonance Images

MOCNN- Multiple Output Convolutional Neural Network

CHAPTER 1

INTRODUCTION

1.1 Background

In this new era, images have become the main source of information exchange raising the need for their automated interpretation from a natural point of view, which can be done conveniently and accurately by humans. In some scenarios, recent advancements in technology have even surpassed the capabilities of the human brain. It is now possible to interpret images in a completely automated environment and even use Artificial Intelligence for decision making. For instance, CNNs (Convolutional Neural Networks) are widely used for image segmentation and object identification

1.2 Current Scenario

Machine learning is a typical feature of this digital age. Following the process of how machines learn to classify we come to know that prerequisites play an important role in image processing and object identification. There may be many techniques available to accomplish a particular task but with different accuracies and algorithms. We will look at various possible methods which can be used in our task. In this digital world, images are used extensively for communications, visualizing and analyzing. They are generally stored as a 3D matrix representing red, green and blue color channels. However, they can be easily transformed into other color channels to extract luminance, saturation, etc. Additionally, an image may contain multiple objects and different backgrounds, all of which might not be essential for the classification task. Hence, preprocessing step for feature selection and feature extraction helps us to filter useful information from among the datasets.

1.3 ANN

Multilayer perceptron (dense neural network) is a feed-forward artificial neural network resembling biological neural networks i.e., the structure and functioning of the human brain. Similarly, to the brain nerve cells, ANN comprises neurons to pass the information from one layer to another in a sequential manner. As we know, biological neurons are made up of three parts.

1. Dendrites: They receive input from the layer in front of the dendrite.
2. Axon: Neurons are connected to each other by axons.
3. Synapses: It transfers synaptic outputs to neurons in the next layer.

Basically, the input is received by the dendrites and sent to the nucleus, where the nucleus determines whether to produce an output signal or not. When the output is launched from the nucleus, it passes through the axons and reaches the synapses where it passes to the next layer of neurons. The information received by the dendrites is passed to the cell nucleus, which determines whether the stimulus is to be passed to the next layer or rejected depending on a certain threshold.

Similarly, ANN is made up of neurons called nodes which are interconnected by links and each link is assigned a specific weight. Node receives input from many nodes and either perform an action or not based on the activation function used. Output is produced when the sum of the input values reaches the threshold and is then passed to the input for the next shift. Otherwise, the input will be rejected, and no action is taken.

ANN are generally used because -

- Once ANN begins self-learning, it can establish complex non-linear relationships as actual relationships between inputs and outputs are non-linear relationships in real life.
- ANN addresses human-like intellectual abilities after learning from large datasets. As the dataset grows, the power of artificial intelligence grows. It can recognize relation between input and output data never seen before.
- No constraints are applied to ANN inputs. ANN can give better results on heterogeneous data having very high volatility and non-constant variance.

Because of the above advantages ANN are generally used in -

- Image processing and character recognition because images are usually non-linear.
- Recognizing signs based on some mathematical calculations and predictions as the image needs to be processed. ANN finds the class with which the object is most compatible.
- ANN can try to find and predict the unpredictable behavior of certain events before the event has occurred. This can be very helpful in areas such as weather forecast, stock market forecast, score forecast, etc.

But using ANN has certain disadvantages and challenges such as -

- As ANN is used to predict unseen data so training a model requires a long training time. This is especially true if we use the CPU for training instead of the GPU.
- Large amounts of data are required by the model having many layers to fully build the model as there are many connections with different weights, so it needs to adjust the weights based on the given input.
- Having a good dataset does not guarantee the best results. In addition, network weights should be optimally adjusted so that later predictions may be more accurate.

1.4 Image Segmentation

Feature selection is the method of choosing the minimum set of features to deal with problems more compactly, effectively and computationally efficient. It includes practices such as creating new features, creating existing features, and removing unnecessary and non-essential features, combining multiple functions to a minimum number, and dividing one function into multiple features.

With the sudden increase in computer vision applications over the past few years, image segmentation has proved to be a crucial preliminary step for achieving the desired results. Semantic segmentation divides the image into meaningful parts, with each part as a predefined class. There are countless tasks that can be solved with the help of photo segmentation like object classification, disease diagnosis, image restoration, morphing,

etc. In this process, each pixel is assigned with a label corresponding to a class and later same class pixels are grouped to determine the boundary of objects. Suppose we have different objects in our image like cars, trees, traffic lights, and animals. Therefore, image segmentation classifies all trees as one single class tree, all animals in a single class animal, and all signals in a single class. As observed, multiple objects of the same type are assigned a common label which might not be desired in all scenarios. To overcome this instance segmentation to distinguish between objects of the same type.

To further understand this, let's consider a scenario where we want to cross the street by looking at various objects in the area, such as cars, traffic lights, sidewalks, pedestrian crossings, and pedestrians. Hence, our eyes immediately process each object and determine its location ensuring our safety. Now the question arising is can a computer do this job? Till a few years back, it was not possible for computers to accurately perform this task until researchers developed Image segmentation and object identification solutions which can be further analyzed using ML/ DL techniques. Also, Image segmentation plays an important role in identifying cancer cells by monitoring cell growth and shape in the blood and alarming for any abnormality. This makes it possible to detect cancer in the blood at an early stage which can be healed in time.

Among all machine learning techniques, CNN (Convolutional Neural Network,) is the most popular neural network because today most of the things work with artificial vision and automation tasks like self-driving cars, or robots that can move on their own. CNN turned out to be the best way to process an image and identify objects, detect scenes, human faces and other details in the image. There are many ways to do image segmentation such as Region-Based Image Segmentation, Edge-detection based Image Segmentation, and Clustering-based Image Segmentation. Here in this project, we have looked at various image segmentation processes and compared the results among them.

1.4.1 Region-Based image segmentation

It separates the objects into different regions based on some threshold value. This technique makes use of the image pixel intensities. It segments similar intensity regions out of higher intensity regions. A threshold is selected for differentiating so it is also called threshold segmentation. Pixel intensities less than the threshold value come under one

region and above the threshold value pixel area come under another region. Similarly, more than one threshold can be selected, and an image can be segmented into more than two regions based on pixel intensities. But if we have multiple objects, multiple thresholds may be required if there is an overlap among the objects.

Some of the advantages of region-based segmentation are fast operation speed, helpful in situations where there is high contrast, and the calculations are simple

But this method does not perform well when objects have their edge pixels overlapping with each other as it is unable to identify the boundaries of both objects and may consider both objects as the same.

1.4.2 Edge Based Segmentation

In an image it is the edges that separate an object from another object in the background. So, this technique takes advantage of this fact every time it encounters a new edge while scanning it to detect a new object. This edge detection is done with the help of a filter matrix that needs to be run on the image to detect the specified edge type. It is one of the most used and simple to implement image segmentation technology because most image processing tasks involve objects. Identification that can be achieved by detecting the edges of the image. It's using discontinuous local features of the image for detecting edges and defining the boundaries of objects. This technique works well in the image having only one or two objects and neither is present in the overlap between them. In other cases, this technique will not be able to correctly identify the object. They are also called the kernels. There are many types of these like:

- Sharpening filter – used to sharpen an image
- Blurring filter - used to blur out an image.
- Sobel Filter (vertical) -used to detect vertical edges in the image.
- Sobel Filter (horizontal) - used to detect horizontal edges in the image
- Laplace Filter - used to detect horizontal edges as well as vertical edges in the image.

As discussed above these filters are just a weight matrix called kernels and in order to use them, these kernels convolved over images. Convolution comprises element-wise matrix

multiplication to a new value for a pixel. Then kernel is moved to the next pixel as per the stride and similar operations are performed. This step is continued for every pixel in a sequential manner to enhance the contrast between different objects

1.4.3 Clustering-based image segmentation

This technique works by splitting the data points in the image into several clusters based on similar pixel values. Based on the number of objects that can represent the value of clusters can be selected. This can be a time-consuming process, but you can get accurate results with small datasets. Image segmentation based on the use of clustering is based on the principle of dividing the pixels of an image into a near-uniform pixels cluster. It selects a random cluster center throughout the image and then continues to grow the size of the cluster based on the similarity parameter. In this way, the image is divided into areas corresponding to the number of clusters selected. Then during the clustering process-like types of data points are grouped into a single cluster.

The advantages of using clustering-based segmentation are that it works really well on small datasets and generates excellent clusters, but computational time is too large and selecting the optimal value of clusters can be tricky sometimes.

1.4.4 CNN-based image segmentation

This method is currently state-of-the-art in the field of image segmentation research. It works on the 3 dimensions i.e., Height, width and number of channels. The first two dimensions represent the resolution of the image, and the third dimension represents the number of channels (RGB) or the intensity value of the colors red, green, and blue colors. Images are normally fed to neurons which are reduced in dimensions to reduce the size of the network, and processing time, and to avoid underfitting. Also, if an image of size $224 * 224 * 3$, it will be converted to 1D, creating an input vector for 150528 which is still too large to be fed into the network. It is useful in scenarios such as self-driving cars to help identify different objects such as label trees, cars, signals, people, footpaths, trucks, cycles, animals, etc. which can be used to train the CNN and make driving decisions according to the objects identified in the scene. In recent times deep learning is becoming a very important part of image segmentation techniques and algorithms continuously

contributing to accomplishing computer vision tasks and building well-trained and intelligent automatic monitoring systems.

Some advantages of using CNN Based Segmentation are that it outperforms the traditional image segmentation methods and can operate on any size of the input to make pixel-wise predictions, but it needs a large dataset and if the network is deep then each step takes a long time.

Some of the areas in which image segmentation is used are autonomous driving, automatic braking, parking sensors, robotic path detection, scene recognition, motion detection and video surveillance.

1.5 CNN

1.5.1 Convolution Layer

Filters or kernels are used to step through an image at fixed gap intervals called strides. Size selection. The length of the stride is important to achieve the desired result. Dot product when the filter sweeps the image is calculated on the part of the image where the filter is placed. Next is the sum of all the values in the product matrix. It is copied to the corresponding location in the convolution feature map matrix. Therefore, we get a reduced image dimension feature map. There are different types of filters, and each filter is used for the extraction of a different kind of function from the photo. For example, a filter serves to extract a filter that can be extracted using feature types from images based on shape and edges and different filters based on color intensities.

Some of the parameters which help to adjust CNN's performance are

- Stride - It defines the number of pixels with the help of which we should pass our clear out over the picture in order that we will recognize a brand-new set of pixels at the same time as doing convolution. Stride's price ranges from 1 to a few relying upon the quantity of loss which we may be accommodated during convolution. The quantity of loss in the picture will increase with the growing price of stride.
- Padding - It is a system of including zeros across the border of a unique picture symmetrically. This facilitates in acquiring the characteristic map output to be of

length as in step with our requirement. Commonly its miles used to maintain measurement of the picture after convolution.

- Filters -. These can be of many types. Each filter will increase the intensity of the output generated after convolution. So, if we have three filters then the intensity of the output can be three. There are three parameters on which the output of convolution relies i.e., Stride, Depth, and Padding. Then these parameters should be finely tracked to attain the preferred output.

During the convolution operation, the dimensions of the input image are reduced after the filter matrix is folded over the input image. It also increases the depth of the image to help identify the area of the photo. Example: $9 * 9$ RGB image with $3, 3 * 3$ kernel in increments 1 (per channel) generates $7 * 7 * 3$ RGB functions. Where 3D represents the depth of the convolved image.

1.5.2 Pooling Layer

It handles a small kernel on the image and is used to select pixels having the highest intensity while discarding others. The resulting matrix is a reduced dimension matrix of the feature image thus helping to reduce unnecessary sparse image cells that are useless in classification. Max pooling helps in reducing the dimension of the network (or image), which may lead to information loss. It works on the idea that the pixels adjacent can be approximated by the maximum information-carrying pixel.

1.5.3 Activation Layer

It mainly uses ReLu as an activation function which sets every negative value as zero while retaining positive values. This step is usually followed by convolution and the pooling layer. They are becoming cutting edge in many areas such as computer vision achieving human-like or better performance. Sounds intriguing, but CNN's design is a tough job in itself. There are no fixed specifications for CNN design. Many researchers Have given a general suggestion but they don't always catch up to the task as it depends not only on the algorithm but also on the data. CNN leverages spatial hierarchy, which is useful for characterizing data, extracting characteristics, and classifying them into different classes. This is evolved into a stream of data expansion and preprocessing to

increase data as more data becomes available allowing for better training and avoiding overfitting. This will help you create more robust models for new samples as it tries to generalize noise during the training phase.

1.5.4 Dropout Layer

Dropout layers are usually applied after layers that contain neurons in a fully connected network. It is a regularization layer. It randomly omits the weights of some input neurons at each iteration to let the other neurons in that iteration get more weight. Usually, people have 20-50% dropouts. Using the dropout layer improves network performance by 1-2 %.

1.5.5 Fully Connected Layer

The number of parameters in the layer continues to converge to reach the desired number of classes. A few hidden units in the layer can enhance the learning ability of the network, but saturation enhances the network accuracy. There is no formulation of the unit to select but usually hits and trial work. These accumulate with the depth of the fully connected subset network. Most networks tend to work well when the number of units is a multiple of 64. Generally, 2 to 3 layers of networks are good if after flattening the outputs of the Convolutional layers, there are enough patterns being passed to the network.

1.6 Preprocessing by CNN

CNN's capture hierarchical spatial features, but algorithm performance is also limited by patterns that exist in the data. One of the features required for a good algorithm is generalization. Training data tends to overfit and provides very accurate results for training data, but with poor performance on test data. One way to improve this is to use data enhancement and preprocessing. Techniques such as inversion, rotation, translation, channel drop, Gaussian noise, etc. Trimming this creates more synthetic data and makes it more robust against invisible real-world data helping to classify better. The other benefit it provides is more data than is usually possible to better optimize CNN parameters. On the other hand, there are many techniques to try to do it. It provides a clearer image than

intentional noise such as Gaussian noise. See in the dark, super-resolution and blurry image reconstruction are just a few of them.

1.6.1 Architecture of a CNN

CNN design structure based on data and optimization techniques is useful for most CNNs that are less dependent on the structure. The CNN architecture contains a flow of all the steps taken to create a Convolutional neural network. This includes CNN transmissions that collect and stores the data in the database, convolution layer, activation layer including ReLu layer, dropout layer followed by stacking norm layers. This will create a fully connected layer and give output. Following Flow chart follows CNN's architecture.

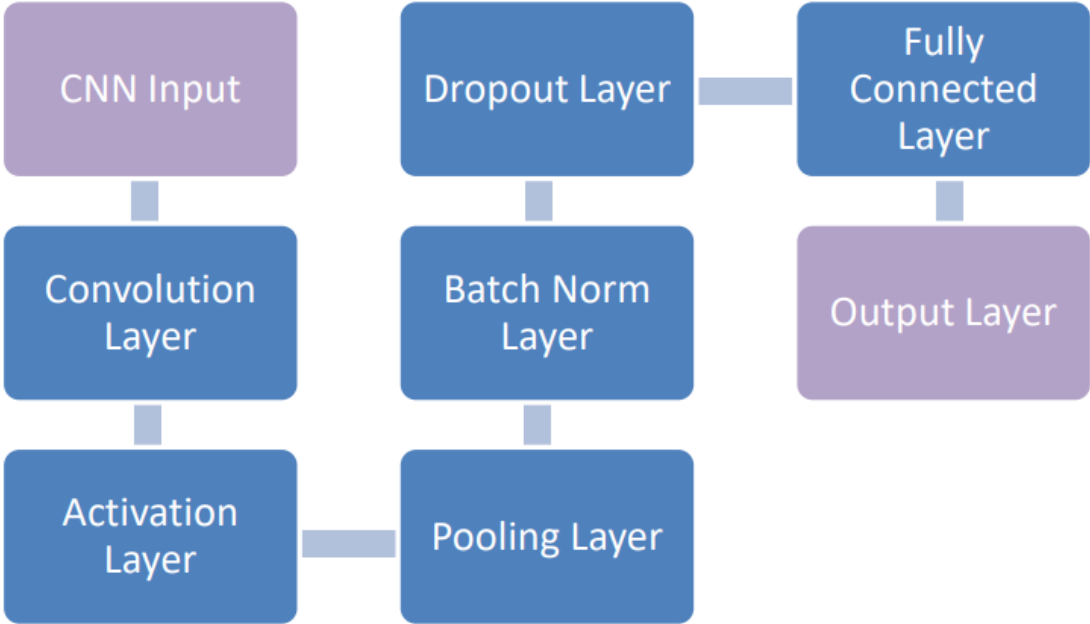


Fig 1.1 Architecture of CNN

The output of a CNN for tasks such as classification is usually a different probability of output. For example, if the final output of a CNN has four units (usually called classes), they will be normalized. In the probability of occurrence of these classes, the unit with the highest probability is marked as a starting class. This can actually be different in task-

specific cases such as facial recognition. The output is an encoded format of the input, not a representation of the output of a particular class. Backpropagation helps minimize errors by reducing the loss / cost function.

1.7 Objective of work

Here, this task uses image segmentation using CNN to detect plant diseases. As we know that our farmers are at stake due to depletion of crops and one of the main reasons for their situation is deterioration of crops due to illness. This is because most farmers in our country are illiterate, making it challenging to learn new things easily to detect such diseases of crops early. So, by the time they come about their plant diseases, it has already been infested. Harvests if mainly intersected timely, they may be healed. Therefore, through this approach we want to develop an automated method for identifying plant diseases. This is found on the leaves of all kinds of plants by taking use of the leaves of different types of plants recorded. It then tests each leaf with a model to automatically detect such diseases. This reduces the pressure of making crop health expertise available and reduces the risk of harvest loss. Here techniques such as CNN, edge detection, and clustering are used to analyze and discover useful results for such classified images.

1.8 Scope of work

As we know, India is one of the largest crop-producing countries, but still, the crops it produces are not enough for the people residing here. Therefore, it is necessary to produce hygienically and disease-free plants for the people of the country. The country develops with people when there are people who can make a healthy and nutritious contribution to the well-being of the country. Therefore, the work we are about to do is automated and covers a wide range within the industry. Because of detection of plant diseases based solely on images, farmers can act quickly for the analysis of the type of crop he / she grows. This ultimately contributes to good crop production for a better country.

1.9 Organization of Dissertation

Chapter 1 provides a background to this task. This includes all the underlying technologies; the necessity and importance of this work is explained in detail with the help of a workflow chart to deepen the reader's understanding.

Chapter 2 has a detailed literature search. It is done to explain all the important work already done in this work area and approaches to solving the problem.

Chapter 3, Describes the implementation. It also describes the architecture used for CNN.

Chapter 4 provides a detailed analysis and review of the implemented methods. All of the results were discussed, and areas for improvement were identified. The result is displayed.

It also describes the use and accuracy of charts and other factors.

In Chapter 5, the conclusions and future scope of this work were discussed. Detailed discussion on how to do that and the expansion of the use of this technique has occurred.

Last section, contains references to all the resources used in this thesis.

CHAPTER 2

LITERATURE REVIEW

2.1 Literature review

Olaf Ronneberger et al. [1] performed image segmentation on a biomedical image dataset. They performed the segmentation of the neural structure found in microscopic images of human blood samples. Cell proliferation tracking in the human body is performed automatically using this approach. Typically, the task of CNN is classification, and the output of a single image is a class label. But in many other applications must include localization information in the output. They applied a segmentation task to a raw image categorizing objects of different colors produces a black and white segmentation mask. The foreground is white, and the background is black. Then the image is mapped with a pixel-by-pixel loss of weight to identify an edge pixel. The Prediction class label provides a local area for pixels in the image. Sliding window setup for training the network. The accuracy of the model is 77.5% and the training time is about 10 hours.

Benugopal K.R. et al. [2] Combining CNN and CC (Connected Component) algorithms Segment the SEM image. As we know, CNN is used to extract features directly from raw images with minimal pretreatment. In addition, CNN can recognize patterns in images that do not have this previously as it was provided as training data. Suppose it resembles one of the training data images. The accuracy of the model (Fscore) is 78% [2]. Performed neuronal tissue detection is done with a SEM (scanning electron microscope) dataset. SEM provides a dataset of images. The resolution is almost perfect for identifying dense neuropil components. This is an automated one which describes an approach that uses CNNs to identify affinity graphs. Furthermore, this affinity graph can be combined with any partitioning algorithm to accurately segment the image. Neurons are particularly difficult to segment because they are branched, intertwined, and very dense. Axons in neural structures are so thin that only electron microscopy has enough resolution to reveal

them. To recognize 2D CNN is used as a multi-layered pattern with very advanced image conversion techniques perceptron. Six feature maps are created for each layer of this CNN model. All filters of the size of the CNN are $5 * 5$. [2]. Random selection based on standard normal distribution; deviations are for calculating the edge weights of the model. Submission to CNN is raw material of EM image. CNN automatically recognizes features in the image. Subsequently CNN learns by adjusting the weights with the help of a stochastic gradient descent learning algorithm.

Shan E. Ahmed Razaetal. [3] Uses a CNN-based deep architecture for that cell, nucleus and gland segmentation in fluorescence microscopy and histological images. She made it possible to work with variable input strength and object size using an additional convolution layer to make the model robust to noisy data. This model helps to create a molecular profile for an individual cell. The input dataset usually uses a noisy fluorescent image and performs segmentation. The task is difficult. In addition, the variable cell size complicates the segmentation algorithm. This technique gives better results with such noisy data. The result of this model is useful for building molecular profiles in multiplex fluorescence images. It learns some features of image resolution to better understand why models are more robust and flexible.

Zhuoling Li et al. [4] has developed a domain customization method called CLUCNN. This method is designed specifically for medical imaging. This contrasts with the normal object identification task. Developed to detect pedestrians, cars, or trees, the task here is to detect objects in medical images like deformation of corneal images and objects of lung X-ray images. The things to keep in mind are that the medical imaging is that deep neural networks are plagued by very large amounts of data parameters, and this slows down the convergence speed. Sparse methodology for overcoming this problem and a new learning rate scheme was introduced. This leads to the fact that the size of the training data is not as large as general image training data. The size of the training data can be less than 400, which makes it impossible to cover all possible cases when training a model. Images can also be taken in different hospitals resulting in variations in conditions, images related to

the same issue may vary significantly. As due to these shortcomings in medical datasets, deep neural network models cannot be used.

Because there are many parameters to train and a larger dataset is needed to work. So, they developed a framework called clustering convolutional neural. Network (CLU CNN). It is specifically targeted at the segmentation of medical images. The model is relatively small and faster than RCNN consisting of CNN and Agglomerates Nesting cluttering filtering (ANSF) used to target single-targeted medical image objects recognition. The two main contributions of their work are one is the domain matching method. It was developed without any additional training to work efficiently, the others were BNIN nets designed by them to improve the stability of the model [5]. As the number of microscopic images increased, automatic nanoparticle detection was developed, it will be an indispensable task. AyseBetul Oktayetal. [5] discussed detection methods of nanoparticles in microscopic images by segmentation. It suggested a way to do that detection of nanoparticles using deep learning and recognition of their shape and size algorithm. This method uses multiple output CNNs and has two outputs. Recognition comes first in an output that shows the position of nanoparticles in the input. The other is segmentation output that outputs the boundaries of segmented nanoparticles. One of them shows the output and the position of the nanoparticles in the image and the second output indicates the distance of the object placing a border in the center of the window.

Ahmed Bassio unyetal. [6] uses CNN to organize images into different categories and input datasets for different scenes. Here, an image segmentation map of the image is created. The purpose was to represent the image itself through a segmentation map. Leading to a one-to-one mapping to a set of labels for each image pixel. This approach also identifies the shape, location, and size of the object. They first extracted the image based on the given suggestions, then created the feature vector of the image to encode the image accordingly. Then image vector classification is performed. Feature extraction is of three types. H. High, medium and low-level feature extraction techniques are available. Attributes or characteristics such as color, texture, contrast, hue that describe the physical characteristics. It is considered a low-level feature extraction in the image. Intermediate level features may include to extract shapes and spatial relationships without knowing the

semantics of the shape. that's all, they try to find them by comparing the various shapes available in the image with the already labeled image in similarity. Next comes high-level image capabilities. This is the place to focus most of the time. It mainly includes mapping from pictorial expression to meaning.

Image segmentation of remote sensing data has been proposed by Xiaomeng Fu et al. [7]. He implemented it using a fully convolutional neural network. FCNs have been shown to outperform CNNs in segmentation of remote sensing images. With this model, the accuracy achieved with FCN of over 85% is achieved. As, the number of remote sensing images is increasing rapidly as the number of satellites increases. Also, it is difficult to manually segment these images to find useful results. Now that deep learning technology has advanced, it has become possible to automatically extract high level features from input images. FCN can accept input images of any size and expand the data to get a feature map of the last convolution layer. The first level of the model is an image of dimension $h * w$ and the number of color channels [7]. The upper layer of the CNN corresponds to the coordinates of the connection in the image. What CNN does is distort and transform the basic components of an image. (That is, convolution, pooling, and ReLu). A typical pattern recognition network uses a fixed size input on a CNN to produce a non-spatial output H. The output contains only classification information. However, we can use FCN to get an input of any size and generate a classified image map as an output that classifies each pixel of the image. Thus, semantic segmentation of high-resolution remote sensing images uses the FCN network in combination with the Matrix-27 extension technology for efficiency [7]. Roads, plantations, buildings, and bodies of water are accurately segmented in the output.

Ji Shunping et al. [8] Automatically classified distant crops using 3DCNN to capture the image captured by the satellite. To structure multispectral remote sensing data, a kernel is designed. Fine-tuning the parameters of the 3DCNN is also performed in sequence, training the model with crop samples to help your model learn spatially and temporally expressions. A sample of the complete plant growth cycle is retained to retrain the model of mature plants. Plant classification from remote sensing images taken by satellite or

UAV (Unmanned aerial vehicles) is becoming a basic task for income estimation and financial support and grain transport [8]. In addition to remote sensing data, SAR (Synthetic Aperture Radar) image data can also be used as input to the CNN model. Today, deep learning is increasing more and more algorithms for classification tasks. This method can automatically identify the characteristics of multi-level representation (low to high level), low level-edges and shapes, etc. having higher levels of structure, color, and other detailed features.

Guotai Wang et al. [9] analyzed the uncertainty of various types of CNN-based 2D and 3D image segmentation. She analyzed extension-based test time uncertainty to analyze different types of impact image conversion in segmentation output. They suggested extending the trial period uncertainty for monitoring the effects of various transformations on the input image. Task consists of image and noise conversion. Then it compared and combined the suggestions of tracheal uncertainty with model uncertainty. They concluded that the test time extended uncertainty gives better results than any other type of uncertainty calculation.



Fig 2.1 Heat Maps generated by Mohommed[10]

Mohammed Brahim [10] proposed a two tier architecture for better visualization of the leaves. Visualization algorithms are used to explain CNN decisions using a heatmap similar to segmentation architectures like U-Net where the supervision masks are replaced with a second classifier. This heat map tells the importance of each pixel for the classifier's decision. It is then back propagated to the input image used by the network for classification. These algorithms can produce different heat maps based on the chosen heuristics and propagation rules, which helps the understanding of the classifier.

Hao Wu et al. [10] categorizes images into different categories according to the list of the category specified in the input set. Give the model a set of images and a set of categories assigned to a category to each image from the list of available categories. The approach is an image represented by a semantic segmentation map, which is a mapping to one of each pixel of a predefined label set. It also identifies the shape, location, and size of objects. First extracted the image based on the suggestions given and created a feature vector for the image by coding the image accordingly. Next, image vector classification is performed. There are three types of feature extraction. High level, medium level and low-level feature extraction techniques are available. Attributes or characteristics such as color, texture and intermediate level features may include extracting shape and spatial relationships without the real thing. Knowledge of form semantics to compare the different shapes available in the image. Attempts to find similarities with images that have already been labeled [10]. Then they worked at high Planar image capabilities that image segmentation technology focuses primarily on. It includes the assignment of image representations to meaning.

Fengping An et al. [11] proposed a computational model of feedback. The mechanism of deep convolutional neural networks. They have developed two algorithms for this. The task, the first, is to learn and extract the deep features of the image and build the image using CNN feedback mechanism. Second, they need to segment medical images used the model to classify the pixel block samples. As medical images segmentation has received a lot of attention from researchers and clinicians. However, many methods for segmenting medical images are already available, but they are focused on machine learning, not on

image segmentation method in the first place. Because there were already many kinds of semantic images Segmentation available as region-based, threshold-based, variant model-based or graphical. Although based on theory, none of these methods have proven to be as efficient as deep machine learning Segmentation methods. They also solved it using a feedback optimization problem Greedy technique. They suggested the optimization method used to update the feedback Neuronal state in each layer [11]. They did this without changing the post coefficient. The bottom-up method is used to optimize the objective function.

CHAPTER 3

METHODOLOGY

3.1 Our Approach

In this project, we investigated and compared various image segmentation techniques. By trying out and studying various techniques, we discovered that CNN is one of the most powerful tools in image segmentation. Next, we applied CNN to the crops dataset to identify possible diseases existing in them. There are already various techniques for identifying plant diseases, both of which are not automatic and have little accuracy to provide good quality. To achieve accurate results, we have developed a methodology that provides both high accuracy and real time. Findings that help early detection of plant diseases that can be cured in time. We have developed a CNN-based machine learning model that can segment images into different parts. It also helps reduce the number of tasks required to identify plant diseases.

3.2 Problem Statement

Here we are trying to identify plants suffering from diseases of plant tissues. The dataset displays sample images taken from various angles. The main purpose of this task is to minimize the loss caused by disease-affected plants and ultimately result in crop growth for the crop grower. To solve this problem, we use image segmentation techniques using CNN's deep neural network architecture.

3.3 Dataset Used

The information has been amassed from crowd AI from the Plant Village Disease Classification Challenge. Plant Village is an organization working mainly in African villages to help farmers in cultivation. It has been downloaded using the hyperlink

command. The dataset includes leaves of diverse classes of illnesses located in plants. In this dataset every class includes around 1000 photographs of identical type plant illnesses. There are 15 types of illnesses which can be used to categorize entered photographs. The information shape layout is proven withinside the discern(Fig 3.1).

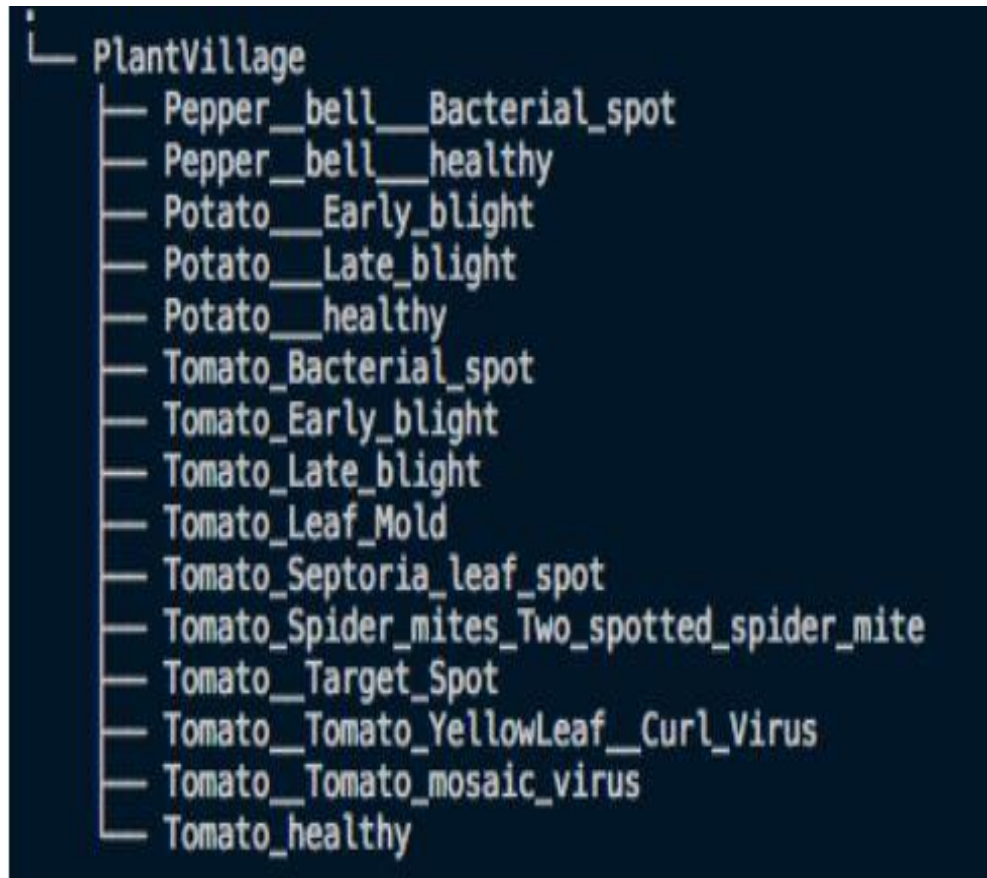


Fig 3.1 - Structure of dataset

The images collected in the dataset are all different types of plants. It is having photos of leaves of the same type of disease grouped into one folder. The new input data has the same properties as these data images and this image is also retained and added to an existing database. The model will be smarter with the new input image function thus helping to classify future input data. Some examples of input image data are shown in Figure 3.2.



Fig 3.2- Samples of images in dataset

In each category of plant there are different types of diseases and variations present. For ex the following figure shows the different type of tomato leaves

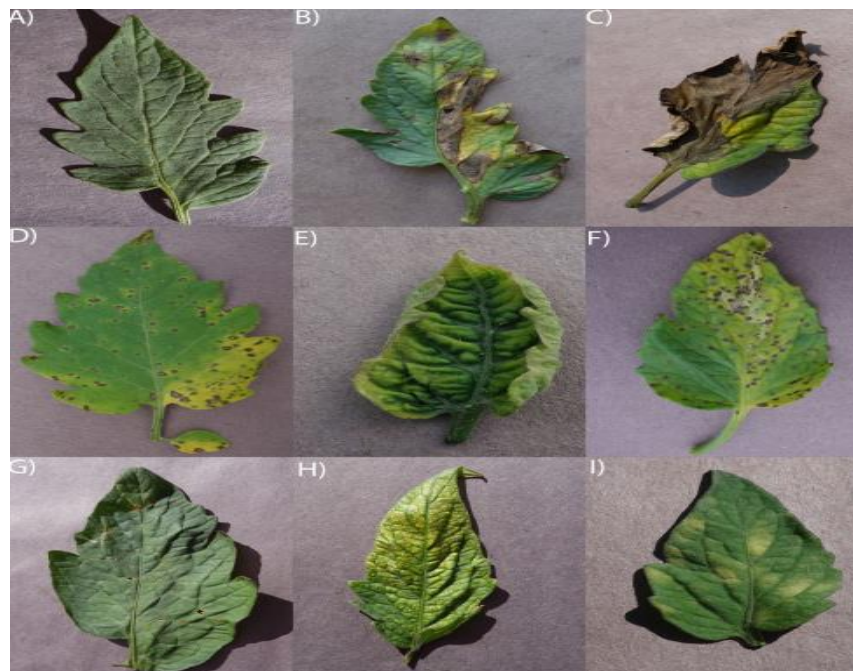


Fig 3.3 Examples of different phenotypes of tomato plants. (A) Healthy leaf B) Early Blight C) Late Blight D) Septoria Leaf Spot E) Yellow Leaf Curl Virus F) Bacterial Spot G) Target Spot H) Spider Mite Damage I) Yellow wilted Leaf)

3.4 Set Up

As we know, growing any plant or element in an unusual place calls for a few pieces of equipment to be used. Here while running in this challenge, we too wanted a few equipment for you to deliver our workout forward. Tools which have been used in the course of the improvement of this challenge are given below. To work with machine learning projects using Python, the following commands make it easier to write code IDE. To do this, the code is written in Python using the Jupyter Notebook. Jupyter provides a great interactive development platform for displaying output and results much easier and improved. Keras is a Python library with the functions needed to implement deep neural networks and more and visualization of the result method. We can use the model of the deep neural network that has already been built to train them on our dataset. In this way we can be sure that our model is efficient on execution time and training.

There are diverse technologies to be had in recent times for doing any unmarried task. But to pick the maximum premiere and appropriate approach consistent with our wants turns into a vital task. Here, we're using gadgets to get to know the deep neural community approach to do picture segmentation. Although there are many different image segmentation techniques to be had, after deep evaluation we've observed that CNN is the most appropriate approach to do the picture segmentation. I used CNN in the middle to force the answer to this question. That is, the detection of plant diseases using photo segmentation. This era has become very well known for the field of machine learning. Therefore, with the development of the times, our method fits the needs of the available industries.

3.5 Steps of Implementation

3.5.1 Importing Libraries

Firstly, all the python libraries which are required to do the project are imported for usage

```
import numpy as np
import pickle
import cv2
from os import listdir
from sklearn.preprocessing import LabelBinarizer
from keras.models import Sequential
from keras.layers.normalization import BatchNormalization
from keras.layers.convolutional import Conv2D
from keras.layers.convolutional import MaxPooling2D
from keras.layers.core import Activation, Flatten, Dropout, Dense
from keras import backend as K
from keras.preprocessing.image import ImageDataGenerator
from keras.optimizers import Adam
from keras.preprocessing import image
from keras.preprocessing.image import img_to_array
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
```

Using TensorFlow backend.

Here, the numpy library is used to perform array operations on image pixel arrays. Mostly in the Keras library, all of the functionality needed to implement the Deep Convolutional

Neural Network like sklearn library are also used to preprocess the input data. Preprocessing is just as important as training the model. Without proper preprocessing, the model can wrongly learn and start to produce incorrect results. Therefore, pretreatment is required to obtain accurate end results.

3.5.2 Loading Image Dataset

```
image_list, label_list = [], []
try:
    print("[INFO] Loading images ...")
    root_dir = listdir(directory_root)
    for directory in root_dir :
        # remove .DS_Store from list
        if directory == ".DS_Store" :
            root_dir.remove(directory)

    for plant_folder in root_dir :
        plant_disease_folder_list = listdir(f"{directory_root}/{plant_folder}")

        for disease_folder in plant_disease_folder_list :
            # remove .DS_Store from list
            if disease_folder == ".DS_Store" :
                plant_disease_folder_list.remove(disease_folder)

        for plant_disease_folder in plant_disease_folder_list:
            print(f"[INFO] Processing {plant_disease_folder} ...")
            plant_disease_image_list = listdir(f"{directory_root}/{plant_folder}/{plant_disease
_folder}")

            for single_plant_disease_image in plant_disease_image_list :
                if single_plant_disease_image == ".DS_Store" :
                    plant_disease_image_list.remove(single_plant_disease_image)

            for image in plant_disease_image_list[:200]:
                image_directory = f"{directory_root}/{plant_folder}/{plant_disease_folder}/{ima
ge}"

                if image_directory.endswith(".jpg") == True or image_directory.endswith(".JPG")
== True:
                    image_list.append(convert_image_to_array(image_directory))
                    label_list.append(plant_disease_folder)

            print("[INFO] Image loading completed")
except Exception as e:
    print(f"Error : {e}")
```

Then we will load the images dataset into memory for capturing. For training the model, the data remains in the main memory until training is complete. Once the model is trained, we can free the memory resources containing this dataset. It is possible that the process will take some time. The code below will get and train 200 images from each folder. Then the image is converted to an array.

3.5.3 Converting image to array

Here, the image is converted to a numeric array for calculation in the model. The CNN mathematical model only understands numbers. So, it updates the weight depending on the pixel value of the image obtained in the image matrix.

```
def convert_image_to_array(image_dir):
    try:
        image = cv2.imread(image_dir)
        if image is not None :
            image = cv2.resize(image, default_image_size)
            return img_to_array(image)
        else :
            return np.array([])
    except Exception as e:
        print(f"Error : {e}")
        return None
```

```
...
[149. 148. 164.]
[154. 153. 169.]
[164. 163. 179.]]
[[126. 123. 138.]
 [101.  98. 113.]
 [118. 115. 130.]
 ...
[149. 148. 164.]
[148. 147. 163.]
[151. 150. 166.]]
[[135. 132. 147.]
 [ 87.  84.  99.]
 [142. 139. 154.]
 ...
[153. 152. 168.]
[147. 146. 162.]
[145. 144. 160.]]
```

3.5.4 Converting Image labels into Binary levels

Image labels are converted to binary level so that the model can identify the image tags easily and quickly for each label. Therefore, the binary representation was encoded and assigned to each label class. This reduces the time it takes to label the image and makes it easier to interpret.

```
label_binarizer = LabelBinarizer()
image_labels = label_binarizer.fit_transform(label_list)
pickle.dump(label_binarizer, open('label_transform.pkl', 'wb'))
n_classes = len(label_binarizer.classes_)

Print the classes

print(label_binarizer.classes_)

['Pepper__bell___Bacterial_spot' 'Pepper__bell___healthy'
 'Potato___Early_blight' 'Potato___Late_blight' 'Potato___healthy'
 'Tomato_Bacterial_spot' 'Tomato_Early_blight' 'Tomato_Late_blight'
 'Tomato_Leaf_Mold' 'Tomato_Septoria_leaf_spot'
 'Tomato_Spider_mites_Two_spotted_spider_mite' 'Tomato__Target_Spot'
 'Tomato__Tomato_YellowLeaf__Curl_Virus' 'Tomato__Tomato_mosaic_virus'
 'Tomato_healthy']
```

3.5.5 Splitting of Input

When designing a machine learning model, we need to split input data into the training and the test set. Therefore, the input is split into a test and training set. The test set makes up 20% of the data and the train set makes up 80% of the data.

```
In [8]: np_image_list = np.array(image_list, dtype=np.float16) / 225.0

In [9]: print("[INFO] Splitting data to train, test")
x_train, x_test, y_train, y_test = train_test_split(np_image_list, image_labels, test_size=0.2,
random_state = 42)

[INFO] Splitting data to train, test

In [10]: aug = ImageDataGenerator(
rotation_range=25, width_shift_range=0.1,
height_shift_range=0.1, shear_range=0.2,
zoom_range=0.2, horizontal_flip=True,
fill_mode="nearest")
```

Here, after dividing the data into sets of training and validations, an expanded input array is created. The extended array is constructed by transforming the input image and rotating or flipping the image randomly. This is useful for learning images when the model needs to perfectly align the images, or the model can also read and classify rotated or mirrored images.

3.5.6 Model Building

In this step, the model is built, and the layers of the model are defined. It consists of repeating pairs of convolutions, pooling and ReLu layers. Here, in this model there are 5 layers of convolution, 6 layers of ReLu or activation, and 3 layers of maximum pooling.

```

model = Sequential()
inputShape = (height, width, depth)
chanDim = -1
if K.image_data_format() == "channels_first":
    inputShape = (depth, height, width)
    chanDim = 1
model.add(Conv2D(32, (3, 3), padding="same", input_shape=inputShape))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(64, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(Conv2D(128, (3, 3), padding="same"))
model.add(Activation("relu"))
model.add(BatchNormalization(axis=chanDim))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(1024))
model.add(Activation("relu"))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(n_classes))
model.add(Activation("softmax"))

```

3.5.7 Model Compiling

In this set, the defined model is compiled as a model with "model.fit_generator". Model follows this instruction to start the build. Full training and building will take some time. The variable "opt" is used to store the model's configuration according to the training performed. This includes the number of epochs that the model goes through to train itself.

```
In [13]:
opt = Adam(lr=INIT_LR, decay=INIT_LR / EPOCHS)
# distribution
model.compile(loss="binary_crossentropy", optimizer=opt, metrics=["accuracy"])
# train the network
print("[INFO] training network...")

[INFO] training network...

In [14]:
history = model.fit_generator(
    aug.flow(x_train, y_train, batch_size=BS),
    validation_data=(x_test, y_test),
    steps_per_epoch=len(x_train) // BS,
    epochs=EPOCHS, verbose=1
)
```

3.6 Plotting the graph

In this step, we create the model and to see the results in the form of a graph. Here first training and validation accuracy is calculated after each epoch. And the improvement of accuracy is plotted to show the improvement in model training over time.

```
In [15]:
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(1, len(acc) + 1)
#Train and validation accuracy
plt.plot(epochs, acc, 'b', label='Training accuracy')
plt.plot(epochs, val_acc, 'r', label='Validation accuracy')
plt.title('Training and Validation accuracy')
plt.legend()

plt.figure()
#Train and validation loss
plt.plot(epochs, loss, 'b', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and Validation loss')
plt.legend()
plt.show()
```


3.6.1 Chart results

Graphs are drawn for better visualization of model performance. Here is a graphic analysis of loss of accuracy and training is also visualized in connection with the increase in numbers of epoch..

3.6.2 Accuracy chart

This graph plots changes in the prediction accuracy of the model using a test set and a validation set. It increases the number of epochs performed during model training.

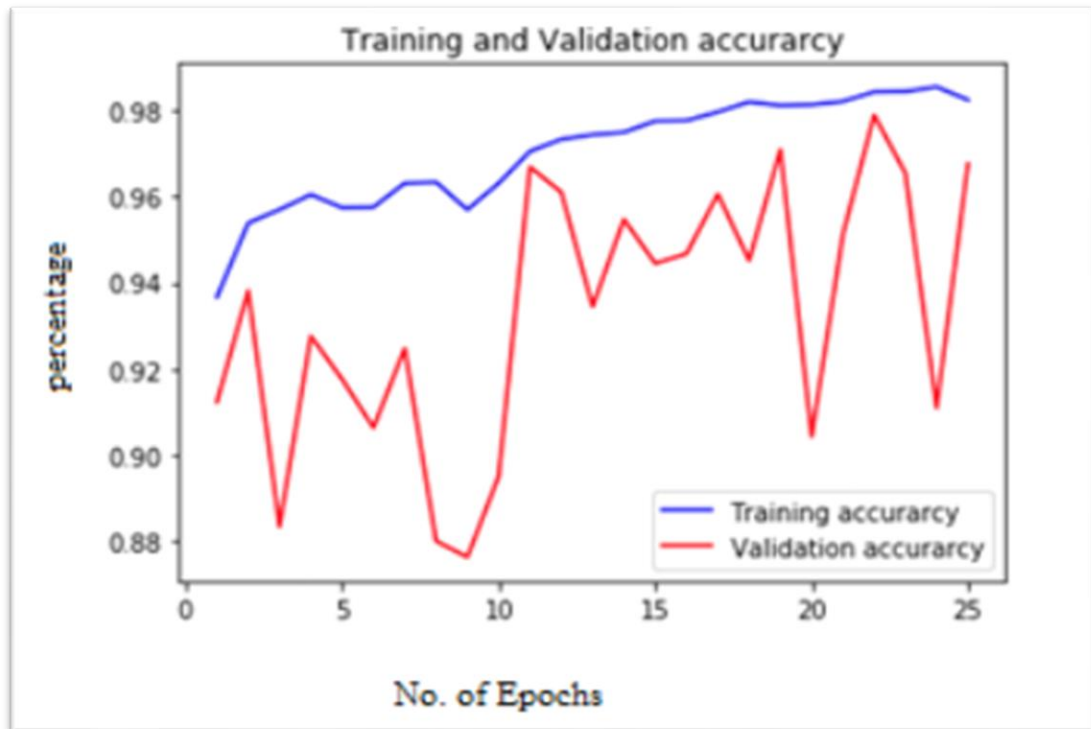


Figure 3.4 Training vs. Validation Accuracy vs. No. of Epochs Graph

Here the accuracy of training and validation improves as the number grows with the epochs. For each epoch, the accuracy is slightly improved.

3.6.3 Loss Graph Plot

This graph plots changes in training loss values using data from the test set and validation set with the increase in the number of epochs.

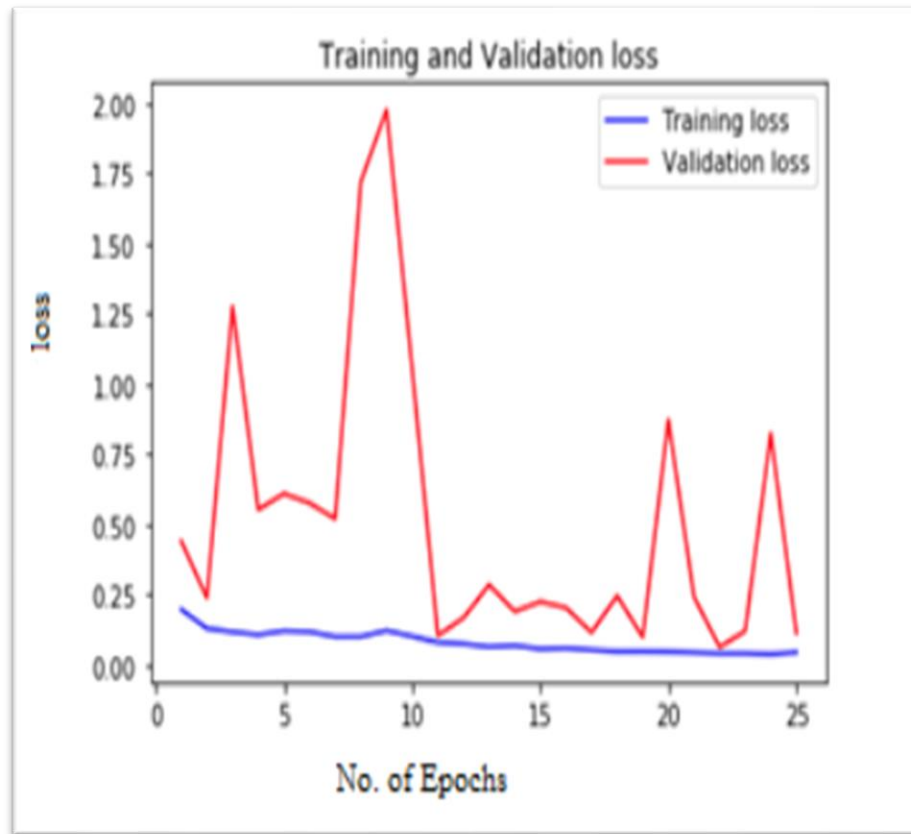


Figure 3.5 Training and validation Loss vs. Number of Epochs

Here the validation loss and training loss approaches zero with the increase in the value of the number of epochs.

3.6.4 Calculation of model accuracy

With the help of validation and test data, the accuracy of the model is calculated, and the model is saved and will be used again in the future. The accuracy of the test set after 25 epochs is 96.77%.

```
In [16]: print("[INFO] Calculating model accuracy")
scores = model.evaluate(x_test, y_test)
print(f"Test Accuracy: {scores[1]*100}")

[INFO] Calculating model accuracy
591/591 [=====] - 2s 3ms/step
Test Accuracy: 96.77383080755192
```

Save model using Pickle

```
In [17]: # save the model to disk
print("[INFO] Saving model...")
pickle.dump(model, open('cnn_model.pkl', 'wb'))

[INFO] Saving model...
```

CHAPTER 4

RESULTS AND ANALYSIS

4.1 Comparison

Here we compare the different methods available for image segmentation. By comparison we find that this applies to tasks where the image size is small and the objects are separated, i.e. region-based segmentation is appropriate if there is no overlap between objects. Also, if the objects overlap but the edges are clearly visible, we can use edge-based segmentation. Where there are many objects that spread throughout the region it is efficient to use cluster-based segmentation. And when we need the most accurate results possible, we use CNN-based segmentation when segmenting objects.

	Edge Based	CNN Based	Cluster Based	Region Based
Advantages	Works better for images having better contrast among the objects.	Provides highly accurate results.	Generates better clusters for small dataset	Simple calculations and fast operation speed.
Disadvantages	Unsuitable for images having many edges.	Takes comparably longer time to train the model.	Computation time is too large and expensive.	Difficult to get accurate segments for indistinguishable pixel boundaries.

TABLE 4.1 Comparison and Analysis of different Image Segmentation Techniques

4.2 Action policy

The "workflow" begins with learning about all available image segmentation techniques. A detailed study of existing methods was conducted to determine which method to pursue to achieve the desired result. Next is to solve the existing problem using the latest technology. Then collect the datasets needed to visualize the solution. Next, write efficient code to develop the model and optimize its parameters to reach the final model that can give the best results.

4.3 Results and accuracy

This work is highly accurate and gives very good results. The requirement of the problem was predicting whether a plant will show symptoms of any disease, making our work very important and somewhat difficult due to lack of prior knowledge of the environmental conditions in which these plants are located. Therefore, we cannot say anything particular about the virtual data as we know processing images of such complex tasks is itself a very big task and arrival at such a model is also a difficult one. Our model is very good at solving problems effectively. With accuracy of about 96% provided by the model for a particular set of inputs. The model may not provide such excellent accuracy with more datasets randomly obtained from all over the world. As there is always room for change and improvement in any job, it may be here as well that this model may not give accurate results for some unknown records. However, with proper sample training for the purpose of the classification we can learn from new datasets.

CHAPTER 5

CONCLUSION & FUTURE SCOPE

5.1. Conclusion

In this work, we compared different image segmentation techniques and explored different techniques and technologies from which we have discovered that CNN is one of the most powerful tools in image segmentation. A detailed analysis of CNN is also done here, explaining the different layers and how they work. We explained most of the existing potential benefits and uses of CNN. As you know, CNN technology has seen a surge in implementation in the manufacturing industry these days. Human life is becoming more convenient thus reducing less manual work. But there is still a lot to do to make these automated monitoring systems accurate and reliable. Some work also needs to be done in the field of making various implemented model to combine as a one such that they can be fed to a robot by which it can act and do the tasks more intelligently and accurately In this work we have compared various image segmentation techniques and after researching various techniques we have found that the CNN is one the most powerful tools in image segmentation techniques. As CNN technology is at a boost of implementation, making human life more and more convenient and reducing dependence on manual tasks. But there's still a lot to do in their production of a more accurate and reliable automatic monitoring system and also the need to improve the accuracy of such systems that are reliable enough to perform important tasks such as monitoring. There is still a lot to do in this area for creating different models to be implemented, combined into one, and feeding them to the machine allowing it to work more intelligently and accurately to complete tasks.

5.2 Future Scope

In any study, there is always room for improvement, innovation, or modification of existing techniques. Despite the availability of very high quality research in this area, there are still many improvements to strive for to make these automated monitoring systems

more accurate and reliable. If the image quality is poor, we need to take steps to deal with the uncertainty and overlapping boundary pixels for segmented objects. Need to improve accuracy for such systems so they become reliable enough to perform important tasks such as monitoring. There is still a lot to do in this area to create different models and combine them into one so that it can be used commercially at a large scale. Also, we can develop mobile applications for this purpose of monitoring. The camera of the smartphone can be used for identification and also at the same time adding more data into a global dataset. Other techniques such as transfer learning could also be explored further.

REFERENCES

- [1] Olaf Ronneberger, Philipp Fischer and Thomas Brox, “U-Net: Convolutional Networks for Biomedical Image Segmentation”, MICCAI 2015: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015 pp 234-241.
- [2] M. L.S., V.K. G. (2011) Convolutional Neural Network Based Segmentation. In: Venugopal K.R., Patnaik L.M. (eds) Computer Networks and Intelligent Computing. ICIP 2011. Communications in Computer and Information Science, vol 157. Springer, Berlin, Heidelberg
- [3] Shan E Ahmed Raza, Linda Cheung, Muhammad Shaban, Simon Graham, David Epstein, Stella Pelengaris, Michael Khan, Nasir M. Rajpoot, Micro-Net: A unified model for segmentation of various objects in microscopy images, Medical Image Analysis, Volume 52, 2019, Pages 160-173,ISSN 1361-8415,
- [4] Zhuoling Li, Minghui Dong, Shiping Wen, Xiang Hu, Pan Zhou, Zhigang Zeng, CLU-CNNs: Object detection for medical images, Neurocomputing, Volume 350, 2019, Pages 53-59, ISSN 0925-2312,
- [5] Ayse Betul Oktay, Anil Gurses, Automatic detection, localization and segmentation of nano-particles with deep learning in microscopy images, Micron, Volume 120, 2019, Pages 113-119, ISSN 0968-4328,
- [6] Ahmed Bassiouny and Motaz El-Saban, “Semantic segmentation as image representation for scene recognition”, 2014 IEEE International Conference on Image Processing (ICIP).
- [7] Xiaomeng Fu and Huiming Qu, “Semantic Segmentation of High-resolution Remote Sensing Image Based on Full Convolutional Neural Network”, 2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE) DOI: 10.1109/ISAPE.2018.8634106.
- [8] Ji Shunping & Chi, Zhang & Xu, Anjian & Shi, Yun & Duan, Yulin. (2018). 3D Convolutional Neural Networks for Crop Classification with Multi-Temporal Remote Sensing Images. Remote Sensing. 10. 75. 10.3390/rs10010075.
- [9] Guotai Wang, Wenqi Li, Michael Aertsen, Jan Deprest, Sébastien Ourselin, Tom Vercauteren, Aleatoric uncertainty estimation with test-time augmentation for medical

- image segmentation with convolutional neural networks, *Neurocomputing*, Volume 338, 2019, Pages 34-45,
- [10] Hao Wu, Rongfang Bie, Junqi Guo, Xin Meng, Chenyun Zhang, CNN refinement based object recognition through optimized segmentation, *Optik*, Volume 150, 2017, Pages 76-82, ISSN 0030-4026,
- [11] Feng-Ping, Liu Zhi-Wen, Medical image segmentation algorithm based on feedback mechanism convolutional neural network, *Biomedical Signal Processing and Control*, Volume 53, 2019, 101589, ISSN 1746-8094,
- [12] Hai Huang, Hao Zhou, Xu Yang, Lu Zhang, Lu Qi and Ai-Yun Zang, "Faster R-CNN for marine organisms detection and recognition using data augmentation", *Neurocomputing*, Volume 337, 14 April 2019, Pages 372-384
- [13] Dan C Cirean, Alessandro Giusti, Luca M Gambardella and Schmidhuber, "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images", *Advances in neural information processing systems* 25 · January 2012
- [14] S. Ji, W. Xu, M. Yang and K. Yu, "3D Convolutional Neural Networks for Human Action Recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221-231, Jan. 2013. doi: 10.1109/TPAMI.2012.59
- [15] Shan E Ahmed Raza, Linda Cheung, Muhammad Shaban, Simon Graham, David Epstein, Stella Pelengaris, Michael Khan, Nasir Rajpoot, Micro-Net: A unified model for segmentation of various objects in microscopy images, *Medical Image Analysis*, Volume 52, 2019, Pages 160-173, ISSN 1361-8415.
- [16] Sadegh Karimpouli, Pejman Tahmasebi, Segmentation of digital rock images using deep convolutional autoencoder networks, *Computers & Geosciences*, Volume 126, 2019, Pages 142-150, ISSN 0098- 3004, <https://doi.org/10.1016/j.cageo.2019.02.003>.
- [17] Moeskops P. et al. (2016) Deep Learning for Multi-task Medical Image Segmentation in Multiple Modalities. In: Ourselin S., Joskowicz L., Sabuncu M., Unal G., Wells W. (eds) *Medical Image Computing and Computer Assisted Intervention – MICCAI 2016*. MICCAI 2016. Lecture Notes in Computer Science, vol 9901. Springer, Cham
- [18] Deniz CM, Xiang S, Hallyburton RS, Welbeck A, Babb JS, Honig S, Cho K, Chang G. Segmentation of the Proximal Femur from MR Images using Deep Convolutional

Neural Networks. Sci Rep. 2018 Nov 7;8(1):16485. doi: 10.1038/s41598-018-34817-6. PubMed PMID: 30405145; PubMed Central PMCID: PMC6220200.

[19] AyseBetulOktay, Anil Gurses, Automatic detection, localization and segmentation of nano-particles with deep learning in microscopy images, Micron, Volume 120, 2019, Pages 113-119, ISSN 0968-4328,

[20] Pim Moeskops, Jelmer M. Wolterink, Bas, H. M. van der Velden, Kenneth G. A. Gilhuijs, Tim Leiner, Max A. Viergever and Ivana Išgum, "Deep Learning for Multi-task Medical Image Segmentation in Multiple Modalities" MICCAI 2016: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016 pp 478-486

[21] Mostefa Ben naceur, Rachida Saouli, Mohamed Akil, Rostom Kachouri, Fully Automatic Brain Tumor Segmentation using End-To-End Incremental Deep Neural Networks in MRI images, Computer Methods and Programs in Biomedicine, Volume 166, 2018, Pages 39-49, ISSN 0169-2607,

[22] Deniz, Cem & Hallyburton, Spencer & Welbeck, Arakua & Honig, Stephen & Cho, Kyunghyun & Chang, Gregory. (2018). Segmentation of the Proximal Femur from MR Images using Deep Convolutional Neural Networks. Scientific Reports. 8. 10.1038/s41598-018-34817-6.