PLANT DISEASE DETECTION USING DEEP LEARNING

A PROJECT REPORT

Submitted in partial fulfillment of
the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by

Priyam paliwal	**(19EMCCS087)**
Vinit jain	(19EMCCS118)
Shivam kumar	(19EMCCS104)
Yash khandelwal	(19EMCCS121)
Prabhat rai	(19EMCCS082)

Under The Guidance Of
Dr. Anusuya
(Professor)

BIKANER TECHNICAL UNIVERSITY , BIKANER

MTRC

MODERN INSTITUTE OF TECHNOLOGY AND RESEARCH CENTRE
Alwar, Rajasthan, India – 673 601

Session 2022-2023

BIKANER TECHNICAL UNIVERSITY , RAJASTHAN

CERTIFICATE

Certified that this project report **"PLANT DISEASE DETECTION USING DEEP LEARNING"** is the original work of **"PRIYAM PALIWAL ,VINIT JAIN ,PRABHAT RAI ,YASH KHANDELWAL ,SHIVAM KUMAR"** students of B.Tech Final Year VIII Semester(Computer Science branch) who carried out the project work under my supervision.

SIGNATURE
Dr.Anusuya
SUPERVISOR
CSE Department
MITRC COLLEGE, ALWAR

SIGNATURE
Dr.J.R.Arun Kumar
HEAD OF THE DEPARTMENT
CSE Department
MITRC COLLEGE, ALWAR

ACKNOWLEDGEMENT

Firstly, I would like to express my gratitude to my advisor for the beneficial comments and remarks.

It gives me immense pleasure to express my sincere thanks towards **Dr.Arun Kumar** (Head of Department) and **Dr.Anusuya**(Project Supervisor),Professor, of Computer Engineering Department, Modern Institute of Technology Research Centre, Alwar (Rajasthan) for their constant support and guidance throughout the course of this work. Their sincerity, thoroughness and perseverance have been a constant source of inspiration for me. I would like to thank Dr. S.K Sharma, director of this institute, for providing all facilities and faculty members without whom this project would have been a distant reality, I also intend my heartiest thanks to my family and well-wishers.

PRIYAM PALIWAL(19EMCCS087)

PRABHAT RAI(19EMCCS082)

VINIT JAIN(19EMCCS118)

YASH KHANDELWAL(19EMCCS121)

SHIVAM KUMAR(19EMCCS104)

Contents

List of Figures

ABSTRACT

Agriculture is extremely important in human life.Almost 60% of the population is engaged in some kind of agriculture, either directly or indirectly.There are no technologies in the traditional system to detect diseases in various crops in an agricultural environment, which is why farmers are not interested in increasing their agricultural productivity day by day. Plant diseases can have a significant impact on crop yields, leading to global economic losses of 220 billion Dollar and food insecurity. So,rapid and accurate disease detection is essential for early intervention and prevention of crops. In this project, we propose plant disease detection system developed using the Py- Torch framework. The proposed system employs a convolutional neural network (CNN) architecture for detecting plant diseases from images of plant leaves.The dataset used for training and testing the model includes images of healthy and diseased leaves of several plants, such as tomato, potato, apple, etc.The CNN model consists of several convolutional layers, followed by max-pooling layers, and a fully connected layer for classification. The model achieved an accuracy of 96% on the test set, demonstrating the effectiveness of the proposed approach in detecting plant diseases. This system has potential applications in the field of precision agriculture and can aid farmers in detecting and managing plant diseases more efficiently and also suggest the appropriate pest used to decrease the effect of disease if any disease found. Overall, this project has the potential to revolutionize the agricultural industry by improving crop yields, reducing costs, and increasing efficiency.

Keywords: Convolutional neural network, Deep learning, Plant disease, Agriculture and plantvillage

Chapter - 1

INTRODUCTION

Crop growth and yield are essential aspects that influence the field of agriculture as well as farmer economically, socially, and in every possible way. So, it is necessary to have close monitoring at various stages of crop growth to identify the diseases at right time. But, humans naked may not be sufficient and sometimes it would be misleading scenarios arise. In this aspect, automatic recognition and classification of various diseases of a specific crop are necessary for accurate identification. In this chapter, I will provide a tour d'horizon for the proposed methodology of my research work. This chapter consists of the background, problem statement, objectives, and scope of the proposed methodology. One of the severe causes of increased microbial infection is illiteracy among farmers in India. Once a crop is infected by some disease it is difficult for farmers to find out the real cause of diseases. pathogens and pests are affecting the crops badly. The crop produces for 5 main food crops by 10 -40%, This data is in reference to a study report which is published by UC Agriculture and Natural Resources. In the Indian context where agriculture contributes to 16% of GDP and engages almost 60% of the population requires great measures to be taken to avoid plant diseases. According to the Ministry of Food Processing Industries in the year 2016 agricultural loss was 13 billion dollars. One of the helpful measures in plant disease detection methods can be done with the help of image processing and neural network. Neural network and deep learning in some recent research have proved its worth in doing such classification tasks efficiently. Agriculture is an essential sector in countries like India as those countries' economy directly or indirectly dependent

on agriculture. It indicates the necessity of taking care of plants from seedling until the expected crop obtains. Through this process, the crop needs to cross a lot of phases to obtain the expected crop such as weather conditions, the survival of the crop from various diseases, and the survival of the crop from various animals. Of these major phases, the crops can be protected from the various animals by providing proper protection for the field and this issue can be solvable. The next major issue is weather conditions which will not be in the control of humans, humans can only pray for better weather conditions to obtain a better crop. Finally, The major issue is very crucial to protect the crop from various diseases as these diseases can impact the complete growth and yield of the crop. If one can able to identify these diseases in time, then the crop can be protected using appropriate fertilizers. If this process of identification and classification of diseases able to digitalize which would be helpful for the agriculturists. It will decrease the time for the identification of diseases and precision in classifying the diseases.

1.1 Background

Agriculture is the oldest profession. Humans started cultivation even science there was no civilization. With the development of science, plants were identified as living-thing. It could also respire, reproduce, and even get prone to various diseases. These are different types of diseases by various microorganisms may it be bacteria, viruses, or fungi. Plant diseases can damage crops to a great extent. It can even be fatal to human beings. One such situation emerged in 1840 when a large amount of potato crop was destroyed due to a disease called the Late blight of potato. This is also known as the Irish famine and this was a darker phase of European history, where people were dying of hunger. We all know that plants are very important in our lives so we need to protect them not only from deforestation but also from various plant diseases. In India still, a large population is engaged in agriculture. It ranks 2nd in the world in terms of agriculture production This is again a major challenge. Crops

are damaged in India because of various factors one of the major causes is natural calamities and other is the microbial diseases. According to United Nations data 96-billion-dollar loss was in agriculture in a decade (starting from 2005). As humans do not have any control over natural disasters but we can control microbial infections in plants. Once a crop is infected by some disease it is difficult for farmers to find out the real cause of the disease. pathogens and pests are affecting the crops badly. The crop produces for 5 main food crops by 10 -40Natural Resources. In the Indian context where agriculture contributes to 16population requires great measures to be taken to avoid plant diseases. According to the Ministry of Food Processing Industries in the year 2016 agricultural loss was 13 billion dollars. One of the helpful measures in plant disease detection methods can be done with the help of image processing and neural network. Neural networks and deep learning in some recent research have proved their worth in doing such classification tasks efficiently.

1.2 Problem description

Plant diseases can cause significant damage to crops, resulting in reduced yield, lower quality, and economic losses for farmers. Some of the major problems associated with plant diseases include:

1. Reduction in yield: Plant diseases can reduce the productivity of crops, leading to lower yields and financial losses for farmers.

2. Economic losses: Plant diseases can have a significant economic impact on farmers, processors, and other stakeholders in the agricultural value chain. This can include lost income, increased production costs, and reduced market access due to quality and safety concerns.

3. Right disease prediction: It is very difficult for farmers to identified various diseases in plants. Many time farmers prediction are wrong in case of similar types of diseases.

4. Lower quality: Plant diseases can affect the quality of crops, resulting in lower market value and reduced consumer satisfaction.

5. Spread of diseases: Plant diseases can spread rapidly within and between crops, leading to widespread losses and increased costs for disease management. Spread of diseases: Plant diseases can spread rapidly within and between crops, leading to widespread losses and increased costs for disease management.

6. Spread of diseases: Plant diseases can spread rapidly within and between crops, leading to widespread losses and increased costs for disease management. Environmental impact: Some plant diseases can have a significant impact on the environment, such as the loss of biodiversity, soil degradation, and water pollution.

7. Food security: Plant diseases can pose a threat to food security by reducing the availability and affordability of food, particularly in developing countries where agriculture is a major source of livelihood.

1.3 Motivation

The study of plant diseases is important as they cause loss to the plant as well as plant produce. The various types of losses occur in the field, in storage or any time between sowing and consumption of produce. The diseases are responsible for direct monitory loss and material loss and for our country where most of the rural peoples generate their livelihood through agriculture so it is more important to stop losses in agriculture fields.

1.4 Objectives

There are some objectives of the proposed methodology:

1. To develop a prototype for a plant disease detection system that can efficiently identify plant diseases.

2. To apply image processing techniques to identify the disease pattern.

3. Use deep learning(ML) algorithms to predict disease.

1.5 Scope

Plant disease detection using deep learning has a wide scope in various fields, including agriculture, horticulture, forestry, and botany. Here are some of the areas where plant disease detection using deep learning can be useful:

1. Crop management: By detecting plant diseases early, farmers can take necessary measures to prevent the spread of diseases, such as using appropriate pesticides or pruning infected parts of the plant. This can increase crop yield and reduce losses.

2. Plant breeding: Plant disease detection can be used to identify disease-resistant plants, which can be used as breeding stock for developing new varieties with better resistance to diseases.

3. Disease surveillance: Plant disease detection can be used for monitoring and surveillance of plant diseases over large areas, which can help in predicting disease outbreaks and taking preventive measures. Environmental monitoring: Plant disease detection can be used to monitor the health of plants in natural ecosystems, which can help in studying the effects of environmental factors such as pollution, climate change, and habitat loss on plant health.

4. Education and research: Plant disease detection can be used as a tool for educating students and researchers about plant diseases and their causes, symptoms, and management.

5. Precision agriculture: Plant disease detection can be integrated with other technologies such as GPS, drones, and sensors to enable precision agriculture, which involves using data-driven techniques to optimize crop yield, reduce inputs, and minimize environmental impact.

Chapter - 2

LITERATURE REVIEW

Mondal in 2015[1] :Detection and classification technique of Yellow Vein Mosaic Virus disease in okra leaf images using leaf vein extraction and Naive Bayesian classifier Methodology used:K-mean clustering,Basic Morphological function,Naive Bayesian presents classification and detection techniques that can be used for plant leaf disease classification. Here, pre-processing is done before feature extraction. RGB images are converted into white and then converted into grey level image to extract the image of vein from each leaf. Then basic Morphological functions are applied on the image. Then the image is converted into binary image. After that if binary pixel value is 0 its converted to corresponding RGB image value. Finally, by using Pearson correlation and Dominating feature set and Naïve Bayesian classifier disease is detected.[3]

Padol in 2016[2] :SVM classifier based grape leaf disease detection. Methodology used: K-means clustering algorithm with SVM , Color co-occurrence method there are four steps. Out of them the first one is gathering image from several part of the country for training and testing. Second part is applying Gaussian filter is used to remove all the noise and thresholding is done to get the all green color component. K-means clustering is used for segmentation. All RGB images are converted into HSV for extracting feature.[5]

Reza in 2016[3] :detecting jute plant disease using image processing and machine learning Methodology used:Color co-occurrence method,Multi SVM Classifier.The paper presents the technique of detecting jute plant disease using image processing. Image is captured and then it is realized to match the size of the image to be stored in the database. Then the image is enhanced in quality and noises are removed. Hue based segmentation is applied on the image with customized thresholding formula. Then the image is converted into HSV from RGB as it helps extracting region of interest. This approach proposed can significantly support detecting stem-oriented diseases for jute plant.[8]

Tejonidhi in 2016[4] : Plant disease analysis using histogram matching based on Bhattacharya's distance calculation Methodology used:Bhattacharaya's similarity calculation they have proposed for a technique that can be used for detecting paddy plant disease by comparing it with 100 healthy images and 100 sample of disease1 and another 100 sample of disease2. It's not sufficient enough to detect disease or classify it training data is not linearly separable.[9]

Arivazhagan in 2018[5] : Mango leaf diseases identification using convolutional neural network Methodology used: Convolutional Neural Network(CNN) proposed a framework based on automated deep learning for the recognition and classification of various diseases in mango plants. The dataset utilized for this framework consists of 1200 images which include both diseased and healthy leaves of mango. The accuracy obtained from the proposed framework is 96.67%.[1]

Mehra in 2016[6] :Maturity and disease detection in tomato using computer vision Methodology used:Threshholding algorithm k mean clustering includes tomato disease detection using computer vision. A gray scale image is turned into binary image depending on threshold value. The threshold algorithm is used for image segmentation. The threshold values are given color indices like red, green, blue. But the thresholding is not a reliable method as this technique only distinguishes red tomatoes

from other colors. It becomes difficult to distinguish ripe and unripe tomatoes. For this K-means clustering algorithm is used to overcome the drawbacks. K-means create a particular number of non-hierarchical clusters. This method is numerical, unsupervised, nondeterministic and iterative. Then separating the infected parts from the leaf, the RGB image was converted into YcbCr to enhance the feature of the image. The final step is the calculation of the percentage of infection and distinguishing the ripe and unripe tomatoes.[2]

Tripathi in 2016[7] : Recent machine learning based approaches for disease detection and classification of agricultural products Methodology used: K-mean clustering,Gray-Level Co-Occurrence Matrix,Support Vector Machine,Artificial Neural Network. popular methods have utilized machine learning, image processing and classification-based approaches to identify and detect the disease of agricultural product.[10]

Pawar in 2016[8] : Cucumber disease detection using artificial neural network Methodology used:Artificial Neural Network, Gray level co-occurrence matrix. The methodology for cucumber disease detection is presented in this paper . The methodology includes image acquisition, image preprocessing, feature extraction with Gray level co-occurrence matrix (GLCM) and finally classified with two types: Unsupervised classification and supervised classification. [6]

Narmadha in 2017[9] : Detection and measurement of paddy leaf disease symptoms using image processing Methodology used: Support Vector Machine, Artificial Neural Network, FUZZY classification,k-means,color co-occurence method. Paddy plant is an important plant in continental region. RGB images are converted into gray scale image using color conversion. Various enhancement techniques like histogram equalization and contrast adjustment are used for image quality enhancement. Different types of classification features like SVM, ANN, FUZZY classification are used here. Feature extract6ion uses different types of feature values like texture feature, structure feature and geometric feature. By using ANN and FUZZY classification, it

can identify the disease of the paddy plant.[4]

Prakash in 2017[10] : Detection of leaf diseases and classification using digital image processing. Methodology used:K-mean clustering,Gray-Level Co-Occurrence Matrix,Support Vector Machine. image processing technique are used to detect the citrus leaf disease. This system includes: Image preprocessing, segmentation of the leaf using K-means clustering to determine the diseased areas, feature extraction and classification of disease. Uses Gray-Level Co-Occurrence matrix (GLCM) for feature extraction and classification is done using support vector machine (SVM)[7].

Chapter - 3

IMAGE PROCESSING USING DEEP LEARNING

IDP leverages a deep learning network known as CNN (Convolutional Neural Networks) to learn patterns that naturally occur in photos. IDP is then able to adapt as new data is processed, using Imagenet, one of the biggest databases of labeled images, which has been instrumental in advancing computer vision.

3.1 Image Processing

Digital image processing is the use of a digital computer to process digital images through an algorithm. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and distortion during processing. Since images are defined over two dimensions (perhaps more) digital image processing may be modeled in the form of multidimensional systems. The generation and development of digital image processing are mainly affected by three factors: first, the development of computers; second, the development of mathematics (especially the creation and improvement of discrete mathematics theory); third, the demand for a wide range of applications in environment, agriculture, military, industry and medical science has increased.

3.1.1 Image acquisition

The first step is to acquired a image from mobile camera . The picture is taken in such a way that any distortion avoided. The picture is used as the input for further processing. We've chosen the most common image domains so that we can accept any format as input to our method, including.bmp,.jpg etc

3.1.2 Image pre-processing

The use of computer algorithms to perform image processing on digital images is known as image pre-processing. We can detect the plant by analysing the image with a specific algorithm. We use a similar approach for image processing and detection with a specific algorithm. The image quality is critical in this process.

3.1.3 Image enhancement

The process of modifying digital images so that the effects are more appropriate for display or further image processing is known as image enhancement. These are the method to improve image that is histogram equalization ,noise removal using filter.

3.1.4 Image segmentation

The method of segmenting a digital image into multiple segments is known as image segmentation (sets of pixels, also known as image objects). Image segmentation is used to make image identification and analysis simpler by dividing the image into several segments and analysing each segment individually. Color, texture, and intensity are all common characteristics among the various segments.

3.1.5 Image analysis

This step, image segmentation is used to locate the region of interest. The technique used in segmentation is region-based segmentation, which uses the colour of the leaf to distinguish between healthy and diseased regions of the plant leaf.

3.1.6 Feature extraction

Feature extraction is a part of the dimensionally reduction method in machine learning, which divides and reduces a large collection of raw data into smaller classes. When we have a large amount of data and need to minimise the number of resources while avoiding errors, this step is critical. As a result, function extraction aids in the extraction of the best feature from large data sets by selecting and combining variables into functions.

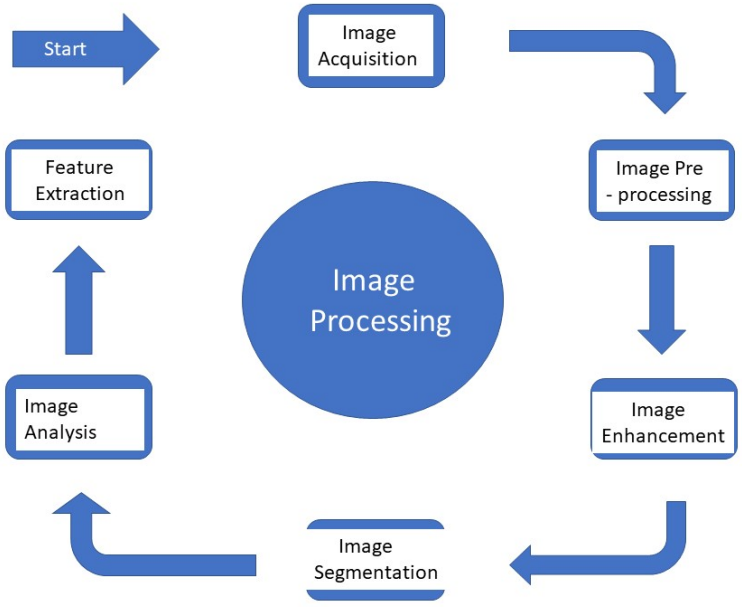

Figure 3.1: Flow Chart of Image Processing Steps

3.2 Disease classification

It is the method of using our qualified deep learning model to recognise plant disease. A digital camera or equivalent system should be used to take an image of the contaminated plant's leaf.Opencv was used to scan the image. Then it determines what kind of plant it is. It determines what kind of disease the plant has after finding it.

3.2.1 Convolutional Neural network

A CNN consists of an input layer and an output layer, as well as multiple hidden layers between them. The hidden layers basically consists of the convolution layer, pooling layer, Rectified Linear Unit, dropout Layer and normalization layers. In the case of image classification, the input is an image and the output is the class name also called label. Inspired by various pre-trained Convolutional Neural Network architectures such as VGG-16, VGG-19, Alexnet, a deep CNN architecture with three hidden layers has been proposed for this work. There is no precise rule in organizing the structure of the individual layer.

1. Input Layer The image can be given as input directly to the CNN model. The size of the image is denoted as [height width number of color channels]. For color image the number of color channels corresponds to 3 and for the grayscale image it is 1. Data augmentation can be done before passing the images to the CNN model. Since neural networks and deep learning models requires large amount of data, the dataset is increased by generation of artificial data through expansion of original dataset. The images are augmented by applying different transformations that include rotations, zooming, cropping, transpose and skewing while preserving the label of the image.

2. Convolutional Layer The prime operation of convolutional layer is convolution operation. The first layer in a CNN is always a convolutional layer for which the input is an image, in case of first network layer or feature map from the

previous layer. The input is convolved with the filters, called kernels in order to produce the output feature maps. Theconvolution output can be denoted as, convolution output can be denoted as,

$$x_3^d = f\left(\sum_{uM_j}^{x_i^{t-1}} = x_{x_j}^t \div b_j^l\right)$$

where, x_{j} represents the set of output Feature maps, Mf represents the sat of input mups, K4y represents the kernel for comviution. by represents the bias term. The size: of the output feafure map is given by; The set of input maps, K_{ij} represents the kernel for convolutio b_j represents the bias term. The size of the output feature map given by,

$$0 = \frac{w - k + 2p}{s} + 1$$

where, O is the ontput height/length, W is the input height/length, K is the filter size, P is the padding, and S is the stride. In order to preserve the size of the outpat, padding of zeros can be employed on the edges of the input. The amount of padding, P can be determined as follows,

$$P = \frac{K - 1}{2}$$

where, K is the filter size.

3. ReLU Layer The Rectified Linear Unit layer also called as the activation layer is used to introduce some non-linearity to the system, since it performs linear operations during the convolution process. So, it is introduced after every convolution layer. This layer simply changes all the negative activation values to 0. The ReLU layer performs a thresholding operation to each element given by, f(x) = max(0,x) This layer plays a significant role in alleviating the vanishing gradient problem and helps to train the system faster. The ReLU suits well for multiclass classification. For binary classification, the sigmoid function can be

used. instead of ReLU.

4. Max-Pooling Layer In this layer, the input is divided into multiple non-overlapping blocks and outputs the maximum among the elements in each block to form an output of reduced size while preserving the important information in the input. It is also capable of controlling the over fitting problem. There is no learning process in this layer.

5. Dropout Layer The basic idea of the dropout layer is that, the input elements with a certain probability are deactivated or dropped out such that the individual neurons are able to learn the features that are less dependent on its surroundings. This process takes place only during the training phase.

6. Batch Normalization Layer The Batch Normalization layer is usually present between the convolution layer and the ReLU layer. It increases the training speed and reduces the sensitivity of network initialization. In this layer the activations of each channel are normalized by subtracting the mini-batch mean and dividing by the mini-batch standard deviation. This is followed by shifting the input by an offset and then scaling it by a factor . These two parameters are updated during the training phase. The batch normalized output, yi is given by,

$$y_i = BN_{\gamma,\beta}(x_i) \equiv \gamma \hat{x}_i + B$$

where,^xi is the normalization of activation xi which is given by equation

$$\hat{x}_i = \frac{x_i + \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}$$

nstant, μ_B is the mini-batch mean and σ_B^2 nce given by,

$$ma = \frac{1}{m}\sum_{i=1}^{m} x_n$$
$$\sigma_{11}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - m)^n$$

where, m is the mini-batch size. In this way network training can be made faster by making the optimization problem easier.

7. Fully Connected Layer In the fully connected layer all the neurons of this layer are connected to all the neurons in the previous layer, thereby combining all the features learned by the previous layer to facilitate classification. This layer produces an N-dimensional vector at the output, where N is the number of classes.

8. Output Layer The output layer consists of the softmax layer followed by the classification layer. The softmax layer outputs a probability distribution based on which, the network model classifies an instance as a class that has the maximum probability value. The Softmax function also called Normalized Exponential is given by equation

$$P\left(c_r \mid x, \theta\right) = \frac{P\left(x, \theta \mid c_j\right) P\left(c_r\right)}{\sum_{j=1}^{k} P\left(x, \theta \mid c_j\right) P\left(c_j\right)}$$

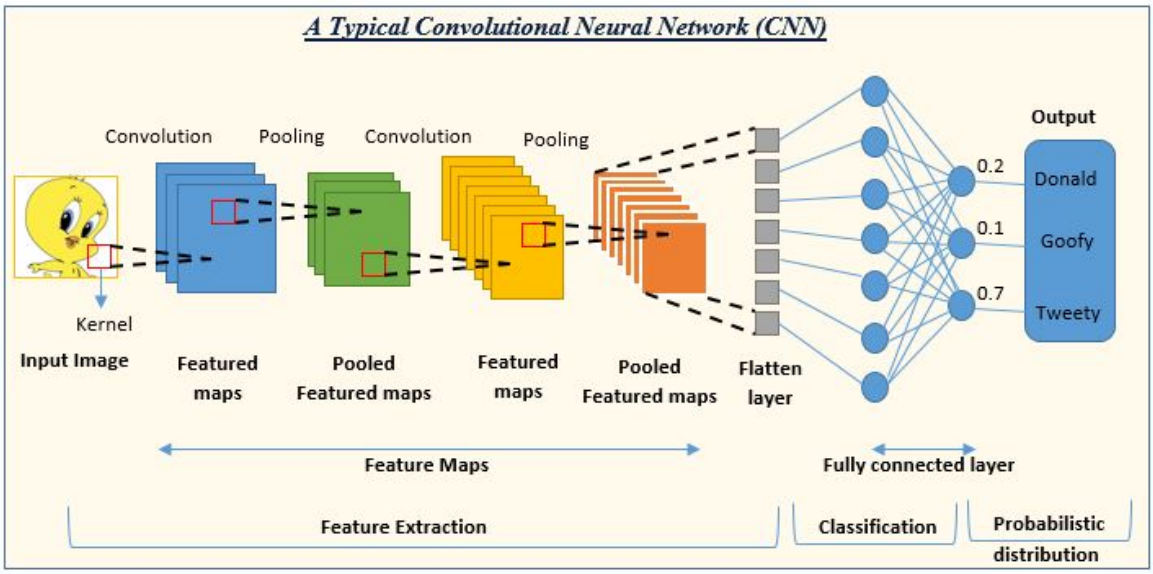

Figure 3.2: A Typical CNN Network

3.3 SOLUTION APPROACH

The solution approach for plant disease detection using deep learning on PyTorch can be summarized into the following steps:

(a) Data Collection and Preprocessing: Collect plant images that show symptoms of disease, and separate them into classes based on the type of disease they show. we may also need to collect images of healthy plants to serve as negative examples. Preprocess the images by resizing them to a common size, normalizing their pixel values, and augmenting them with techniques such as rotation, flipping, and cropping.

(b) Data Splitting: Split the dataset into training, validation, and testing sets. The training set will be used to train the deep learning model, the validation set will be used to tune hyperparameters and avoid overfitting, and the testing set will be used to evaluate the model's performance.

(c) Model Architecture: Choose a pre-trained model such as ResNet or VGG as the backbone of the model, and add custom layers on top of it to adapt it to your specific problem. You can also choose to train a model from scratch. Define the loss function, optimizer, and learning rate schedule.

(d) Model Training: Train the model on the training set using PyTorch's DataLoader to load the images and labels. Use a suitable optimizer such as SGD or Adam, and adjust the learning rate based on the validation loss. Use techniques such as early stopping and model checkpointing to avoid overfitting.

(e) Model Validation: Evaluate the model on the validation set to tune the hyperparameters and avoid overfitting. Monitor the validation loss and accuracy and adjust the model accordingly. Model Testing: Evaluate the final trained model on the testing set and report the performance metrics such as accuracy, precision, recall, and F1 score.

(f) Model Deployment: Once the model is trained and tested, deploy it on web application or mobile app to make predictions on new images.

Chapter - 4

DATASET DESCRIPTION

4.1 Dataset Detail

The Plant leaf diseases dataset with augmentation dataset, in this data-set, 39 different classes of plant leaf and background images are available. The data-set containing 61,486 images. We used six different augmentation techniques for increasing the data-set size. The techniques are image flipping, Gamma correction, noise injection, PCA color augmentation, rotation, and Scaling.

The apple label namely: Apple scab, Black rot, apple rust, and healthy. Corn label namely: Corn Cercospora spot Gray spot, Corn rust, Corn healthy, Corn Northern Blight. Grape label namely: Black rot, Esca, healthy, and Leaf blight. Potato label namely: Early blight, healthy, and Late blight. Tomato label namely: bacterial spot, early blight, healthy, late blight, leaf mold, septoria leaf spot, spider mite, target sport, mosaic virus, yellow leaf curl virus.

4.2 Tools & Technologies

4.2.1 Python

Python as a language has a vast community behind it. Any problem which may be faced is simply resolved with a visit to Stack Overflow. Python is among the

S.No.	Classes	No. of Images
1	Apple_scab	1000
2	Apple_black_rot	1000
3	Apple_cedar_apple_rust	1000
4	Apple_healthy	1645
5	Background_without_leaves	1143
6	Blueberry_healthy	1502
7	Cherry_powdery_mildew	1000
8	Cherry_healthy	1052
9	Corn_gray_leaf_spot	1000
10	Corn_common_rust	1192
11	Corn_northern_leaf_blight	1000
12	Corn_healthy	1000
13	Grape_black_rot	1180
14	Grape_black_measles	1383
15	Grape_leaf_blight	1000
16	Grape_healthy	1076
17	Orange_haunglongbing	5507
18	Peach_bacterial_spot	2297
19	Peach_healthy	1000
20	Pepper_bacterial_spot	1000
21	Pepper_healthy	1478
22	Potato_early_blight	1000
23	Potato_healthy	1000
24	Potato_late_blight	1000
25	Raspberry_healthy	1000
26	Soybean_healthy	5090
27	Squash_powdery_mildew	1835
28	Strawberry_healthy	1100
29	Strawberry_leaf_scorch	1109
30	Tomato_bacterial_spot	2127
31	Tomato_early_blight	1000
32	Tomato_healthy	1591
33	Tomato_late_blight	1909
34	Tomato_leaf_mold	1062
35	Tomato_septoria_leaf_spot	1771
36	Tomato_spider_mites_twospotted_spider_mite	1676
37	Tomato_target_spot	1404
38	Tomato_mosaic_virus	1000
39	Tomato_yellow_leaf_curl_virus	5357

TOTAL IMAGES = 61486

USED IMAGES = 61486

Figure 4.1: Leaf Disease Dataset

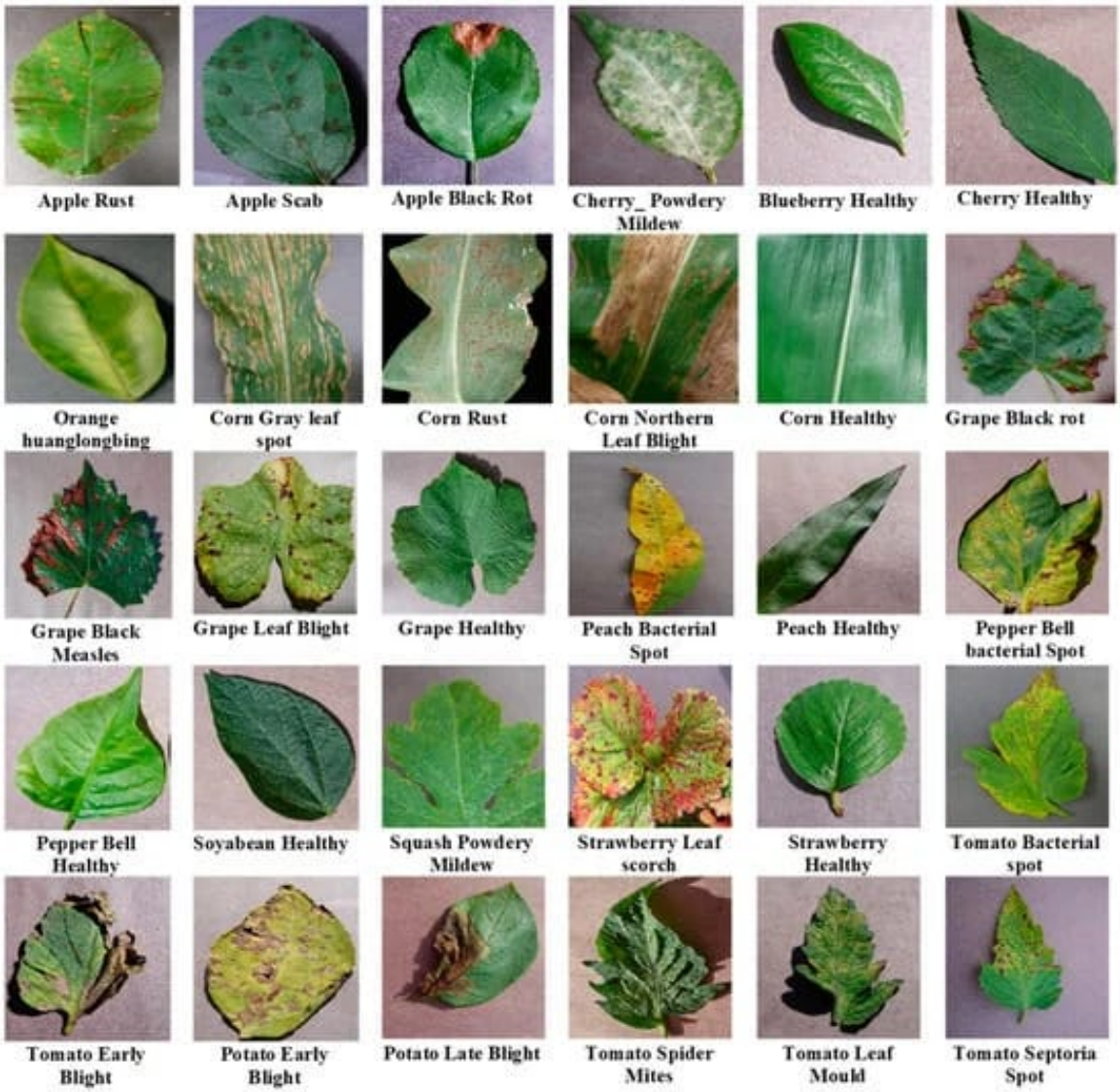

Figure 4.2: Leaf Disease Image Dataset

foremost standard language on the positioning that makes it very likely there will be straight answer to any question Python has an abundance of powerful tools prepared for scientific computing Packages like NumPy, Pandas and SciPy area unit freely available and well documented. Packages like these will dramatically scale back, and change the code required to write a given program. This makes iteration fast. Python as a language is forgiving and permits for program that appear as if pseudo code. This can be helpful once pseudo code given in tutorial papers must be enforced and tested. Using python this step is sometimes fairly trivial. However, Python is not without its errors. The language is dynamically written and packages are area unit infamous for Duck writing. This may be frustrating once a package technique returns one thing that, for instance, looks like an array instead of being an actual array. Plus the actual fact that standard Python documentation does not clearly state the return type of a method, this can lead to a lot of trials and error testing that will not otherwise happen i a powerfully written language. This is a problem that produces learning to use a replacement Python package or library more difficult than it otherwise may be.

4.2.2 Jupyter Notebook

The Jupyter Notebook is an open-source web application that enables you to make and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

4.2.3 Compiler Option

Anaconda is also a premium open-source distribution of the Python and R programming languages for large-scale process, predictive analytics, and scientific computing, that aims to modify package managing and deployment.

4.2.4 Pytorch

PyTorch is a popular deep learning framework that has gained significant traction in recent years due to its ease of use and flexibility. Here are some reasons why PyTorch can be a good choice for plant disease detection using deep learning.

(a) Easy to use: PyTorch has a simple and intuitive API that makes it easy to define and train deep learning models. Its dynamic computational graph also makes it easy to debug and modify models during development.

(b) Flexible: PyTorch provides a lot of flexibility in defining and training deep learning models. It allows for custom loss functions, optimizers, and layers, and supports various data formats and preprocessing techniques.

(c) GPU acceleration: PyTorch can take advantage of GPUs to accelerate training and inference of deep learning models, which can significantly reduce the training time.

(d) Pretrained models: PyTorch provides a range of pretrained deep learning models, such as ResNet and VGG, that can be used as a starting point for custom models, which can save time and computational resources.

(e) Large community: PyTorch has a large and active community of developers and researchers, which means there are plenty of resources and tutorials available to help you get started with deep learning and PyTorch.

4.2.5 Numpy

Numpy is python package which provide scientific and higher level mathematical abstractions wrapped in python. It is the core library for scientific computing, that contains a strong ndimensional array object, provide tools for integrating C, C++ etc. It is additionally useful in linear algebra, random number capability etc.Numpy's array type augments the Python language with an efficient data

structure used for numerical work. Numpy additionally provides basic numerical routines, like tools for locating Eigenvector.

4.2.6 Flask

Flask is a lightweight web framework written in Python that provides a simple and flexible way to build web applications. It is often used for building small to medium-sized web applications, RESTful APIs, and web services.

Here are some key features of Flask:Routing,Templating,WSGI compliance,Extension support,Debugging Testing.

4.2.7 Convolutional Neural Network

A Convolutional Neural Network (CNN) is a type of Deep Learning neural network architecture commonly used in Computer Vision. Computer vision is a field of Artificial Intelligence that enables a computer to understand and interpret the image or visual data. When it comes to Machine Learning, Artificial Neural Networks perform really well. Neural Networks are used in various datasets like images, audio, and text. Different types of Neural Networks are used for different purposes, for example for predicting the sequence of words we use Recurrent Neural Networks more precisely an LSTM, similarly for image classification we use Convolution Neural networks.

Convolutional neural networks (CNN) can be used for the computational model creation that works on the unstructured image inputs and converts to output labels of corresponding classification. They belong to the category of multi-layer neural networks which can be trained to learn the required features for classification purposes.Less pre-processing is required in comparison to traditional approaches and automatic feature extraction is performed for better performance. For the purpose of leaf disease detection, the best results could be seen

with the use of a variation of the LeNet architecture. LeNet consists of convolutional, activation, max-pooling and fully connected layer also LeNet is simple CNN model. This architecture used for the classification of the leaf diseases in LeNet model. It consists of an additional block of convolution, activation and pooling layers in comparison to the original LeNet architecture. The model used in this paper been shown in Each block consists of a convolution, activation and a max pooling layer. Three such blocks followed by fully connected layers and soft-max activation are used in this architecture. Convolution and pooling layers are used for feature extraction whereas the fully connected layers are used for classification. Activation layers are used for introducing non-linearity into the network. Convolution layer applies convolution operation for extraction of features. With the increase in depth, the complexity of the extracted features increases. The size of the filter is fixed to 5×5 whereas number of filters is increased progressively as we move from one block to another. The number of filters is 20 in the first convolution block while it is increased to 50 in the second and 80 in the third. This increase in the number of filters is necessary to compensate for the reduction in the size of the feature maps caused by the use of pooling layers in each of the blocks. After the application of the convolution operation feature maps are zero padded, in order to preserve the size of the image. The max pooling layer is used for reduction in size of the feature maps, speeding up the training process, and making the model less variant to minor changes in input. The kernel size for max pooling is 2×2. Re-LU activation layer is used in each of the blocks for the introduction of non-linearity.Dropout regularization randomly drops neurons in the network duringiteration of training in order to reduce the variance of the model and simplify the network which aids in prevention of over fitting. Finally, the classification block consists of two sets fully connected neural network layers each with 500 and 10 neurons respectively. The second dense layer is followed by a soft max activation function to compute the probability scores for the ten classes.

Chapter - 5

IMPLEMENTATION

In the automation of multiple processes, machine learning plays a critical role. The proposed architecture was designed with that goal in mind, and it is based on machine learning methodologies. Especially in the case of detecting and categorizing images into various disease categories. This section has been structured such that the topic begins with the device specifications and data acquisition for the data used in the currently suggested approach.The second point of discussion would be image segmentation. Feature extraction would be the focus of the debate. The fourth point of discussion would be the classification method. CNN models are best suited for object recognition and classification with image databases. Despite the advantages of CNNs, challenges still exist, such as the long duration of training and the requirement for large datasets. To extract the low-level and complex features from the images, deep CNN models are required; this increases the complexity of the model training. Transfer learning approaches are capable of addressing the aforementioned challenges. Transfer learning uses pre-trained networks, in which model parameters learned on a particular dataset can be used for other problems.

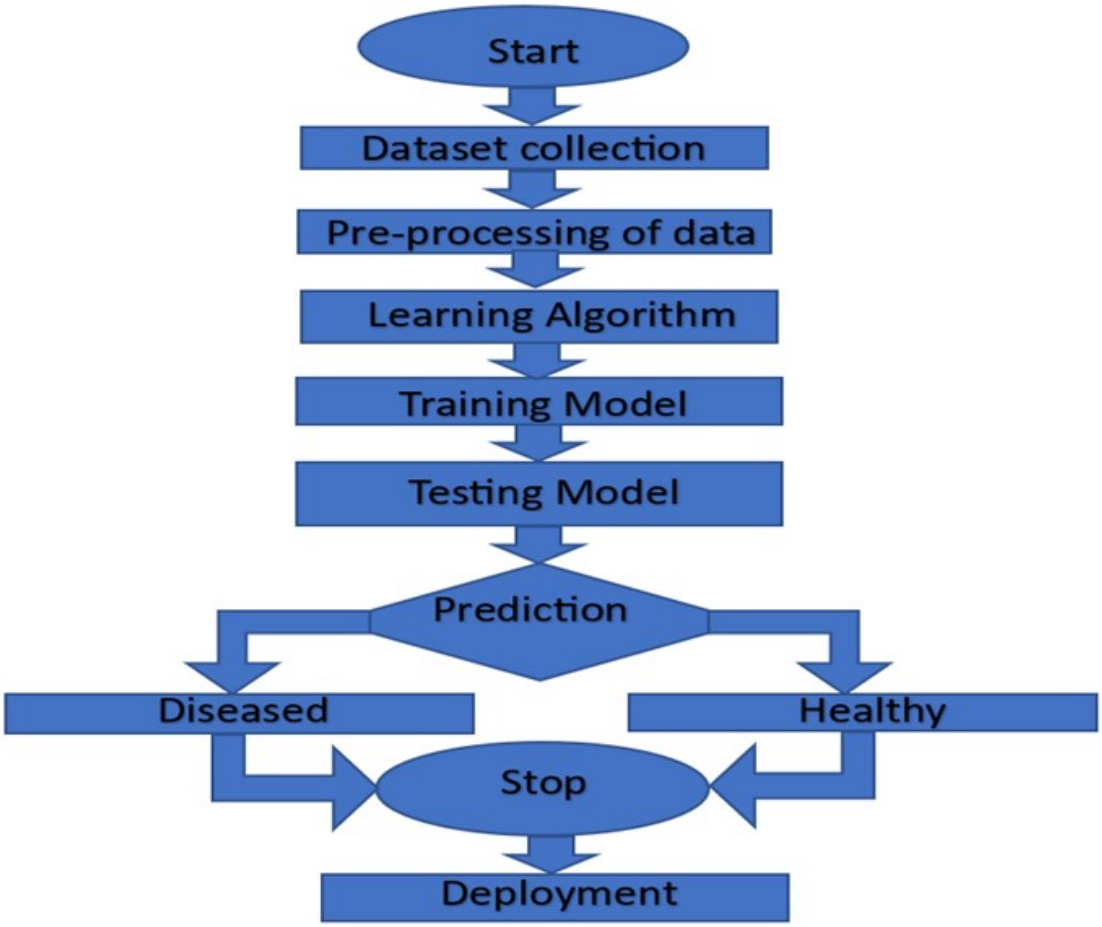

Figure 5.1: Dataflow Diagram of Project Methodology

5.1 Dataset Collection

Dataset collection: The first step is to collect a large dataset of plant leaf images that contain affected as well as unaffected images of plant leaf from disease. The dataset we are taking is labeled so that model can learn to differentiate between healthy and diseased plants. We collect these dataset images from online plant image dataset.

Figure 5.2: Source Of The Data

5.2 Pre-processing

Data preprocessing: Once we have collected the dataset, preprocess the image to make them suitable for deep learning algorithms. This may involves resizing, cropping, normalizing , enhancing ,reducing noise from image.

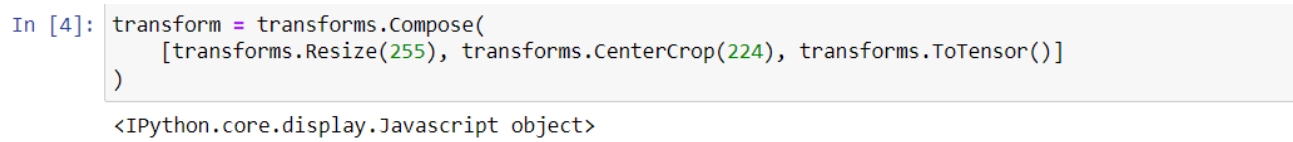

Figure 5.3: Data Preprocessing Step

5.3 Dataset Loading

```
In [5]: dataset = datasets.ImageFolder("Dataset2", transform=transform)

        <IPython.core.display.Javascript object>

In [6]: dataset

Out[6]: Dataset ImageFolder
            Number of datapoints: 61488
            Root location: Dataset2
            StandardTransform
        Transform: Compose(
                       Resize(size=255, interpolation=bilinear, max_size=None, antialias=warn)
                       CenterCrop(size=(224, 224))
                       ToTensor()
                   )

        <IPython.core.display.Javascript object>
```

Figure 5.4: Dataset Loading in model

5.4 Data Splitting

```
In [13]: train_indices, validation_indices, test_indices = (
             indices[:validation],
             indices[validation:split],
             indices[split:],
         )

         <IPython.core.display.Javascript object>

In [14]: train_sampler = SubsetRandomSampler(train_indices)
         validation_sampler = SubsetRandomSampler(validation_indices)
         test_sampler = SubsetRandomSampler(test_indices)

         <IPython.core.display.Javascript object>
```

Figure 5.5: Dataset Splitting Into Test & Training Data

5.5 CNN Modeling Code

Model selection: The next step is to choose a suitable deep learning model for the project. We use the model for image classification task is convolution neural network(CNN).

```
In [22]: class CNN(nn.Module):
             def __init__(self, K):
                 super(CNN, self).__init__()
                 self.conv_layers = nn.Sequential(
                     # conv1
                     nn.Conv2d(in_channels=3, out_channels=32, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(32),
                     nn.Conv2d(in_channels=32, out_channels=32, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(32),
                     nn.MaxPool2d(2),
                     # conv2
                     nn.Conv2d(in_channels=32, out_channels=64, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(64),
                     nn.Conv2d(in_channels=64, out_channels=64, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(64),
                     nn.MaxPool2d(2),
                     # conv3
                     nn.Conv2d(in_channels=64, out_channels=128, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(128),
                     nn.Conv2d(in_channels=128, out_channels=128, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(128),
                     nn.MaxPool2d(2),
                     # conv4
                     nn.Conv2d(in_channels=128, out_channels=256, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(256),
                     nn.Conv2d(in_channels=256, out_channels=256, kernel_size=3, padding=1),
                     nn.ReLU(),
                     nn.BatchNorm2d(256),
                     nn.MaxPool2d(2),
                 )
```

Figure 5.6: Model Preparing Step

5.6 Training Code

Model Training : Once we have selected a model, we train it using the preprocessed dataset. This involves splitting the dataset into training and testing sets, and then using an optimizer and loss function to train the model on the training set.

```
In [36]: batch_size = 64
train_loader = torch.utils.data.DataLoader(
    dataset, batch_size=batch_size, sampler=train_sampler
)
test_loader = torch.utils.data.DataLoader(
    dataset, batch_size=batch_size, sampler=test_sampler
)
validation_loader = torch.utils.data.DataLoader(
    dataset, batch_size=batch_size, sampler=validation_sampler
)
```

```
In [37]: train_losses, validation_losses = batch_gd(model, criterion, train_loader, valida
```

```
Epoch : 1/5 Train_loss:1.158 Test_loss:0.974 Duration:2:07:23.036070
Epoch : 2/5 Train_loss:0.896 Test_loss:0.780 Duration:2:10:21.588655
Epoch : 3/5 Train_loss:0.734 Test_loss:0.667 Duration:12:04:14.439644
Epoch : 4/5 Train_loss:0.582 Test_loss:0.703 Duration:2:17:11.741300
Epoch : 5/5 Train_loss:0.493 Test_loss:0.557 Duration:1:52:20.861600
```

Figure 5.7: Training Phase Of The Model

5.7 Model Accuracy

Model Evaluation: To check the accuracy of a plant disease detection model using CNN algorithm, we can use the testing dataset that we have created during the implementation stage. We can feed the testing images into the trained model and compare the predicted labels with the true labels of the images. Additionally, we can also use other evaluation metrics such as precision, recall, and F1-score to get a more detailed understanding of the model's performance. These metrics can be calculated using the confusion matrix, which is a table that shows the number of true positives, false positives, true negatives, and false negatives for each class.

Overall, checking the model's accuracy is an important step in assessing the quality of the model and identifying any issues that need to be addressed.

```
In [35]: def accuracy(loader):
             n_correct = 0
             n_total = 0

             for inputs, targets in loader:
                 inputs, targets = inputs.to(device), targets.to(device)

                 outputs = model(inputs)

                 _, predictions = torch.max(outputs, 1)

                 n_correct += (predictions == targets).sum().item()
                 n_total += targets.shape[0]

             acc = n_correct / n_total
             return acc

         <IPython.core.display.Javascript object>

In [ ]: train_acc = accuracy(train_loader)
        test_acc = accuracy(test_loader)
        validation_acc = accuracy(validation_loader)

In [38]: print(
             f"Train Accuracy : {train_acc}\nTest Accuracy : {test_acc}\nValidation Accuracy : {validation_acc}"
         )

         Train Accuracy : 96.7
         Test Accuracy : 98.9
         Validation Accuracy : 98.7

         <IPython.core.display.Javascript object>
```

Figure 5.8: Model's Testing Accuracy : 98.6%

5.8 Model Deployment

Deployment: Once we are satisfied with the performance of the model, we deploy the model on a web application .User can upload images of plant leaf to the web app which will then be analyzed by the CNN model to detect the diseases and provide information about how to treat it.

```
import os
from flask import Flask, redirect, render_template, request, url_for
from PIL import Image
import torchvision.transforms.functional as TF
import torchvision.transforms as transform
import CNN
import numpy as np
import torch
import pandas as pd


disease_info = pd.read_csv('disease_info.csv' , encoding='cp1252')
supplement_info = pd.read_csv('supplement_info.csv',encoding='cp1252')



model = CNN.CNN(39)
model.load_state_dict(torch.load("plant_disease_model.pt"))
model.eval()
```

Figure 5.9: Model Deployment

Deployment through Pytorch : Convolutional Neural Networks (CNNs) are a powerful type of neural network that is particularly well-suited for image

classification, object detection, and other computer vision tasks. PyTorch is a popular open-source machine learning framework that provides powerful tools for building and deploying deep learning models, including CNNs.

The deployment of a CNN model through PyTorch involves several steps. First, you need to train your CNN model using PyTorch's high-level abstractions for building neural networks, such as the nn.Module and nn.Sequential classes. Once your model is trained and you are satisfied with its accuracy, you can save the model's state to a file using PyTorch's torch.save() function.

When it's time to deploy your model, you can use PyTorch's torch.load() function to load the saved model state into memory. Then, you can create a new instance of your CNN model and set its state to the loaded state using the model.loadstatedict() method. This will recreate the trained model in memory, ready to make predictions on new data.

To make predictions using your deployed CNN model, you can pass new images through the model using PyTorch's torch.Tensor and torchvision.transforms classes. First, you can convert your image data into a PyTorch tensor using the torch.Tensor() function. Then, you can apply any necessary data transformations using torchvision.transforms. Finally, you can pass the transformed tensor through your CNN model to get a prediction.

To optimize the performance of your deployed CNN model, you can also use PyTorch's torch.jit module to compile your model's code into a high-performance, optimized executable. This can significantly reduce the model's inference time and improve its overall performance.

In summary, deploying a CNN model through PyTorch involves training the model using PyTorch's high-level abstractions, saving the model's state to a file, loading the saved state into memory, creating a new instance of the model, passing new data through the model, and optionally using PyTorch's torch.jit module to compile the model's code for improved performance.

Chapter - 6

RESULT ANALYSIS

Plant diseases can have a significant impact on crop yields, leading to global economic losses of 220 billion Dollar and food insecurity. So, rapid and accurate disease detection is essential for early intervention and prevention of crops.

In this project, we propose plant disease detection system developed using the PyTorch framework. The proposed system employs a convolutional neural network (CNN) architecture for detecting plant diseases from images of plant leaves. The dataset used for training and testing the model includes images of healthy and diseased leaves of several plants, such as tomato, potato, apple, etc.

The CNN model consists of several convolutional layers, followed by max-pooling layers, and a fully connected layer for classification. The model achieved an accuracy of 96diseases.

This system has potential applications in the field of precision agriculture and can aid farmers in detecting and managing plant diseases more efficiently and also suggest the appropriate pest used to decrease the effect of disease if any disease found. Overall, It has the potential to revolutionize the agricultural industry by improving crop yields, reducing costs, and increasing efficiency.

Figure 6.1: Home Page of Project

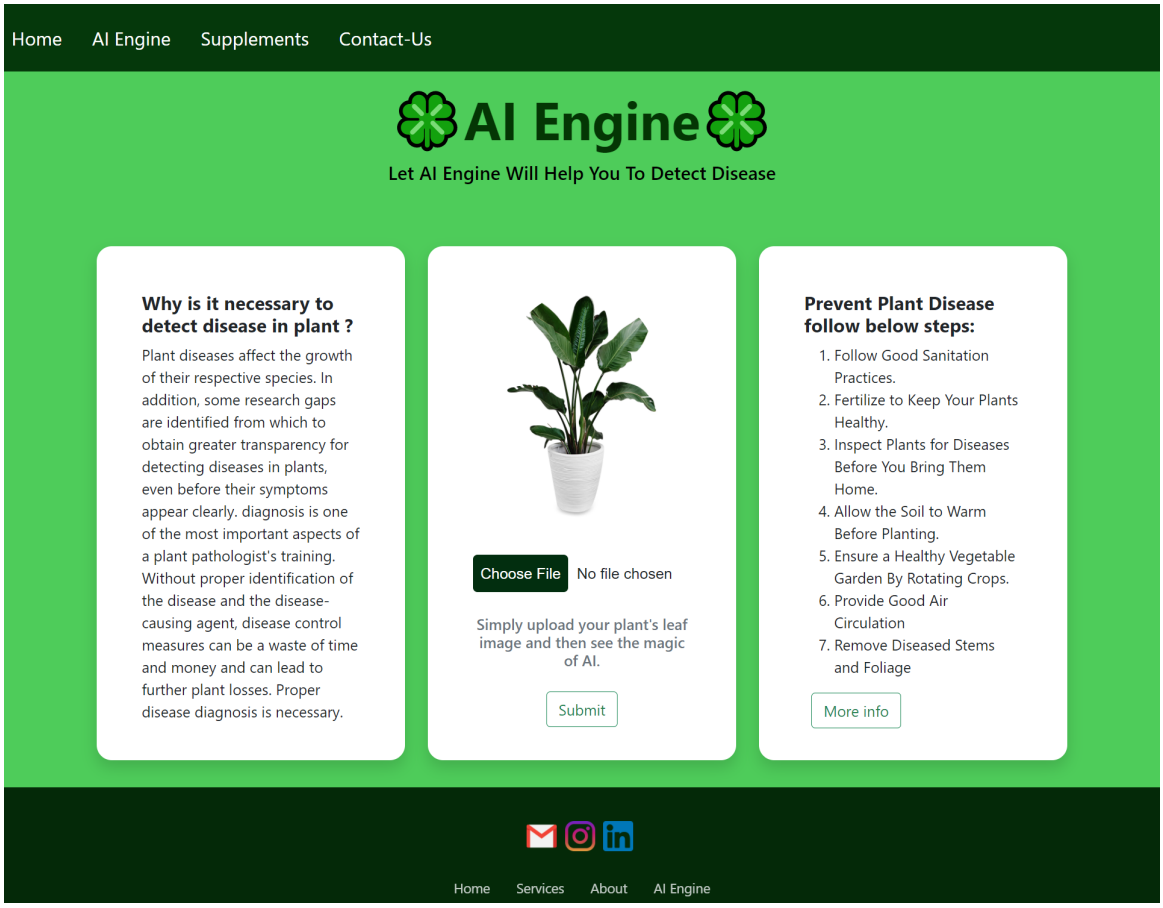

Figure 6.2: Input Page Of Project

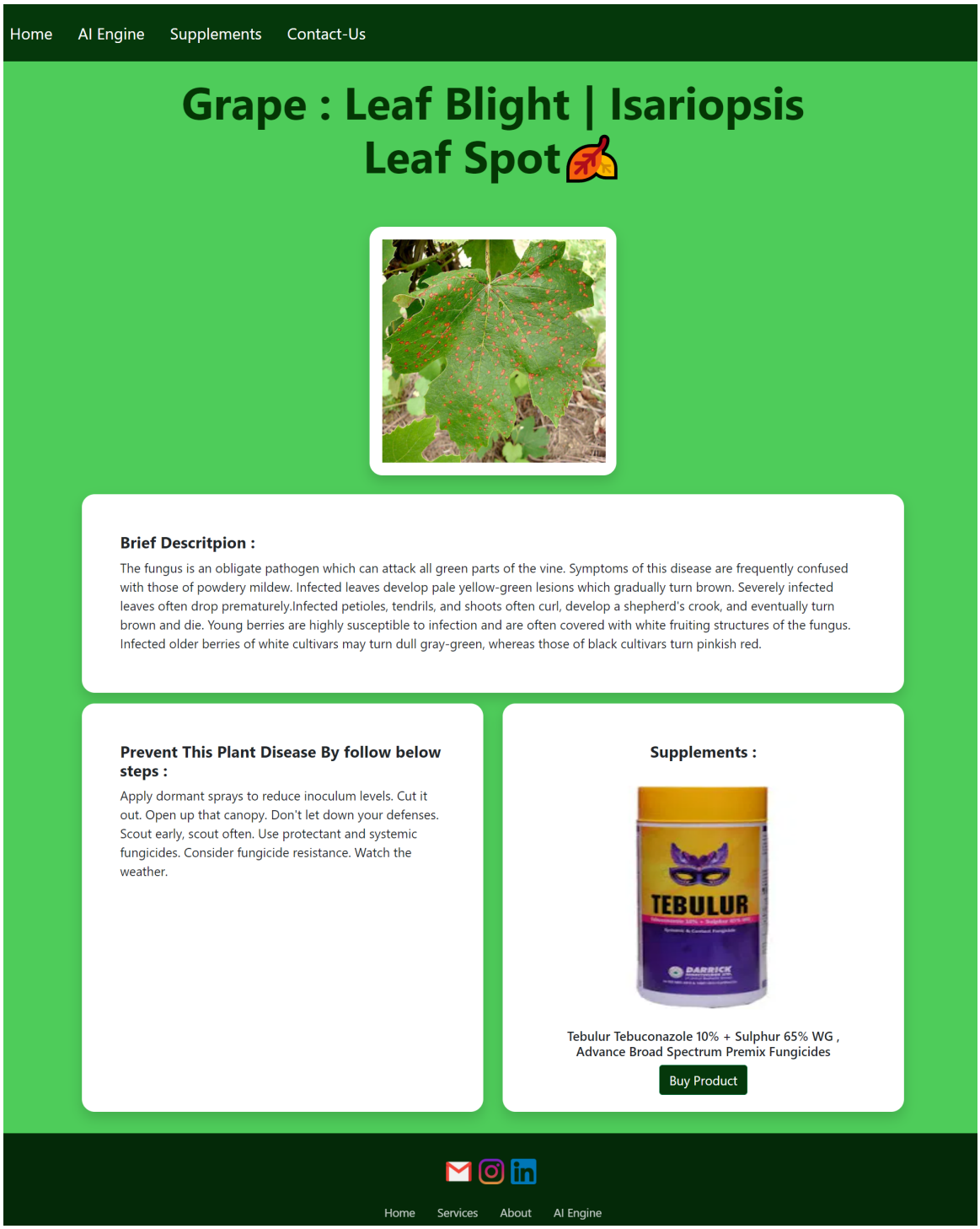

Figure 6.3: Result Page

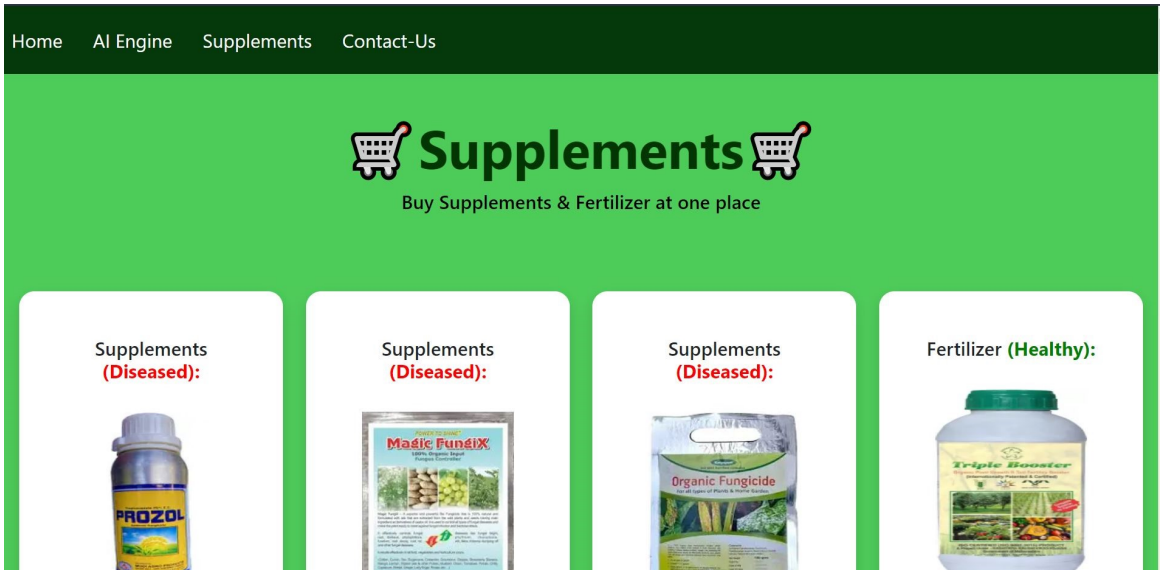

Figure 6.4: Supplements store

6.1 LIMITATIONS

(a) These most crucial points in the selection of any deep learning model are:

- Limited dataset: The accuracy of the model depends heavily on the quality and quantity of the dataset used for training. If the dataset is limited, the model may not be able to accurately identify all the diseases, leading to false positives or false negatives.
- Model overfitting: Overfitting occurs when the model becomes too complex and starts memorizing the training data instead of learning general patterns. This can lead to poor performance on new data.
- Difficulty in generalization: The model may perform well on the specific crops and diseases it was trained on, but may not be able to generalize to other crops or diseases.
- Sensitivity to environmental factors: Environmental factors such as lighting and weather conditions can affect the quality and consistency of the images, which can impact the performance of the model.

(b) Machine learning models are not very efficient in predicting diseases from leaf images when the no of categories is increased.

Chapter - 7

CONCLUSION & FUTURE WORK

7.1 Conclusion

The project on plant disease detection using deep learning has shown promising results in accurately identifying and classifying various diseases that affect plants. Deep learning algorithms, specifically convolutional neural networks (CNNs), were trained on a large dataset of images of healthy and diseased plants to learn the patterns and features that differentiate between them. The project has highlighted the potential of deep learning in assisting farmers and researchers in early detection and management of plant diseases. The use of deep learning models can help in reducing the economic losses caused by plant diseases and improving food security.

7.2 FUTURE WORK

(a) Developing more comprehensive datasets: Deep learning models rely on large and diverse datasets for training. Therefore, there is a need to develop more comprehensive datasets that include a wide range of plant species, diseases, and environmental conditions.

(b) Improving the accuracy of disease identification: Although deep learning models have shown great promise in identifying plant diseases, they are

still prone to errors. Future work should focus on developing more accurate models that can accurately identify the disease and differentiate it from other abnormalities such as nutritional deficiencies, pest damage, or physical damage.

(c) Transfer learning: Transfer learning is a powerful technique that enables the transfer of knowledge learned from one task to another. Future work should investigate the use of transfer learning to improve the accuracy and generalization ability of deep learning models for plant disease detection.

(d) Deployment and Adoption: Finally, there is a need to deploy and adopt the developed models for practical use. Future work should focus on developing user-friendly interfaces, tools, and applications to facilitate the adoption of these technologies by farmers, agronomists, and other stakeholders in the agriculture industry.

References

[1] S Arivazhagan and S Vineth Ligi. Mango leaf diseases identification using convolutional neural network. *International Journal of Pure and Applied Mathematics*, 120(6):11067–11079, 2018.

[2] Tanvi Mehra, Vinay Kumar, and Pragya Gupta. Maturity and disease detection in tomato using computer vision. In *2016 Fourth international conference on parallel, distributed and grid computing (PDGC)*, pages 399–403. IEEE, 2016.

[3] Dhiman Mondal, Aruna Chakraborty, Dipak Kumar Kole, and D Dutta Majumder. Detection and classification technique of yellow vein mosaic virus disease in okra leaf images using leaf vein extraction and naive bayesian classifier. In *2015 international conference on soft computing techniques and implementations (ICSCTI)*, pages 166–171. IEEE, 2015.

[4] RP Narmadha and G Arulvadivu. Detection and measurement of paddy leaf disease symptoms using image processing. In *2017 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–4. IEEE, 2017.

[5] Pranjali B Padol and Anjali A Yadav. Svm classifier based grape leaf disease detection. In *2016 Conference on advances in signal processing (CASP)*, pages 175–179. IEEE, 2016.

[6] Pooja Pawar, Varsha Turkar, and Pravin Patil. Cucumber disease detection using artificial neural network. In *2016 international conference on*

inventive computation technologies (ICICT), volume 3, pages 1–5. IEEE, 2016.

[7] R Meena Prakash, GP Saraswathy, G Ramalakshmi, KH Mangaleswari, and T Kaviya. Detection of leaf diseases and classification using digital image processing. In *2017 international conference on innovations in information, embedded and communication systems (ICIIECS)*, pages 1–4. IEEE, 2017.

[8] Zarreen Naowal Reza, Faiza Nuzhat, Nuzhat Ashraf Mahsa, and Md Haider Ali. Detecting jute plant disease using image processing and machine learning. In *2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT)*, pages 1–6. IEEE, 2016.

[9] MR Tejonidhi, BR Nanjesh, Jagadeesh Gujanuru Math, and Ashwin Geet D'sa. Plant disease analysis using histogram matching based on bhattacharya's distance calculation. In *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pages 1546–1549. IEEE, 2016.

[10] Mukesh Kumar Tripathi and Dhananjay D Maktedar. Recent machine learning based approaches for disease detection and classification of agricultural products. In *2016 International Conference on Computing Communication Control and automation (ICCUBEA)*, pages 1–6. IEEE, 2016.