TURMERIC PLANT LEAF DISEASES DETECTION AND CLASSIFICATION USING DEEP LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

K. LAKSHMI NIKITHA	(21UECS0254)	**(19521)**
P. MAHI PRANATHI	(21UECS0501)	**(20065)**
P. REKHA SHANMUKHI	(21UECS0454)	**(19523)**

Under the guidance of
Dr. S. SRIDEVI M.E., Ph.D.
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

TURMERIC PLANT LEAF DISEASES DETECTION AND CLASSIFICATION USING DEEP LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

K. LAKSHMI NIKITHA	(21UECS0254)	**(19521)**
P. MAHI PRANATHI	(21UECS0501)	**(20065)**
P. REKHA SHANMUKHI	(21UECS0454)	**(19523)**

Under the guidance of
Dr. S. SRIDEVI M.E., Ph.D.
PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled " TURMERIC PLANT LEAF DISEASES DETECTION AND CLASSIFICATION USING DEEP LEARNING " by " K. LAKSHMI NIKITHA (21UECS0254), P. MAHI PRANATHI (21UECS0501), P. REKHA SHANMUKHI (21UECS0454) " has been carried out under my supervision and that this work has not been submitted elsewhere for a degree."

Signature of Supervisor	**Signature of Professor In-charge**
Computer Science & Engineering	**Computer Science & Engineering**
School of Computing	**School of Computing**
Vel Tech Rangarajan Dr. Sagunthala R&D	**Vel Tech Rangarajan Dr. Sagunthala R&D**
Institute of Science & Technology	**Institute of Science & Technology**
May, 2024	**May, 2024**

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

K. LAKSHMI NIKITHA

Date: / /

P. MAHI PRANATHI

Date: / /

P. REKHA SHANMUKHI

Date: / /

APPROVAL SHEET

This project report entitled " TURMERIC PLANT LEAF DISEASES DETECTION AND CLASSIFICATION USING DEEP LEARNING " by " K.LAKSHMI NIKITHA (21UECS0254), P. MAHI PRANATHI (21UECS0501), P. REKHA SHANMUKHI (21UECS0454) " is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners	**Supervisor**
	Dr. S. Sridevi M.E., Ph.D. Professor.,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Dr.S.SRIDEVI M.E., Ph.D,PROFESSOR.,** for her cordial support, valuable information and guidance, she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Mrs. U.HEMAVATHI, M.E., Mrs. C. SHYAMALA KUMARI, M.E.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

K. LAKSHMI NIKITHA	**(21UECS0254)**
P. MAHI PRANATHI	**(21UECS0501)**
P. REKHA SHANMUKHI	**(21UECS0454)**

ABSTRACT

Turmeric is a widely cultivated crop with various medical and culinary uses. However, leaf diseases pose a significant threat to turmeric plants, leading to reduced yield and quality. Various conventional methods are available for disease detection and classification, but they often lack accuracy and efficiency. In recent years, deep learning techniques have emerged as a promising approach for plant disease detection and classification. In this study, we propose a deep learning-based approach for automatic detection and classification of turmeric leaf diseases using a convolutional neural network (CNN) model. We evaluated the proposed approach on a publicly available dataset of turmeric leaf images containing three different types of diseases and mployed a convolutional neural network (CNN) architecture for disease detection and classification. The CNN model was trained using a large dataset of labeled images to learn discriminative features for different turmeric leaf diseases.

Experimental results demonstrate the effectiveness of the proposed approach in accurately detecting and classifying turmeric leaf diseases. The developed model achieved high accuracy, sensitivity, and specificity in identifying various leaf diseases, thereby providing a valuable tool for early disease diagnosis and effective management strategies in turmeric cultivation. The result evaluates the superior performance of the proposed approach in terms of accuracy and efficiency. The proposed approach can assist farmers and agronomists in making informed decisions for disease management and improving the yield of turmeric crops. It can also serve as a basis for the development of an automated system for disease detection and classification in real-time. Overall, the proposed approach offers a promising solution for accurate and efficient detection and classification of turmeric leaf diseases using deep learning, compared to the existing systems.

Keywords: CNN, Deep Learning, Google Colab, Keras, Support Vector Machine, Visual Geometry Group

LIST OF FIGURES

4.1 **CNN Architecture** 10
4.2 **Data Flow Diagram for Input Images** 11
4.3 **Use Case Diagram for Test Image** 12
4.4 **Sequence Diagram for Disease Identification** 13
4.5 **Activity Diagram for CNN Classification** 14
4.6 **Image Acquisition** 17
4.7 **Image Pre-processing** 18
4.8 **Image Segmentation** 19

5.1 **Input Image of Dataset** 22
5.2 **Output Design of Disease Detected** 23
5.3 **Unit Testing** 24
5.4 **Integration Testing** 25
5.5 **System Testing** 26
5.6 **Test Image** 27

6.1 **Output for Disease Identification Accuracy Graph** 30

8.1 **Plagiarism Report** 32

9.1 **Poster Presentation** 34

LIST OF ACRONYMS AND ABBREVIATIONS

CNN	Convolutional Neural Network
GAN	Generative Adversarial Network
GOCOLAB	Google Colab
ICT	Information and Communication Technology
IPYNB	Ipython Notebook
MSVM	Multicategory Support Vector Machine
R-CNN	Reigon based Convolutional Neural Network
R-FCN	Region based Fully Convolutional Network
SSD	Single Shot Detector
SVM	Support Vector Machine
VGG	Visual Geometry Group
YOLO	You Only Look Once

TABLE OF CONTENTS

Page.No

ABSTRACT **v**

LIST OF FIGURES **vi**

LIST OF ACRONYMS AND ABBREVIATIONS **vii**

1 INTRODUCTION **1**
- 1.1 Introduction 1
- 1.2 Aim of the Project 2
- 1.3 Project Domain 2
- 1.4 Scope of the Project 2

2 LITERATURE REVIEW **3**

3 PROJECT DESCRIPTION **6**
- 3.1 Existing System 6
- 3.2 Proposed System 6
- 3.3 Feasibility Study 7
  - 3.3.1 Economic Feasibility 7
  - 3.3.2 Technical Feasibility 7
  - 3.3.3 Social Feasibility 8
- 3.4 System Specification 8
  - 3.4.1 Hardware Specification 8
  - 3.4.2 Software Specification 8
  - 3.4.3 Standards and Policies 8

4 METHODOLOGY **10**
- 4.1 General Architecture for CNN 10
- 4.2 Design Phase 11
  - 4.2.1 Data Flow Diagram for Input Images 11
  - 4.2.2 Use Case Diagram for Test Image 12
  - 4.2.3 Sequence Diagram for Disease Identification 13

4.2.4 Activity Diagram for CNN Classification 14
4.3 Algorithm & Pseudo Code 15
4.3.1 Enhanced Convolutional Neural Network 15
4.3.2 Pseudo Code 15
4.4 Module Description 17
4.4.1 Image Acquisition 17
4.4.2 Image Pre-processing 18
4.4.3 Image Segmentation 19
4.5 Steps to execute/run/implement the project 21
4.5.1 Upload the IPYNB code in the GOCOLAB 21
4.5.2 Create and upload the dataset 21
4.5.3 Run the code in their respective fields 21

5 IMPLEMENTATION AND TESTING 22
5.1 Input and Output 22
5.1.1 Input Design 22
5.1.2 Output Design 23
5.2 Types of Testing 24
5.2.1 Unit Testing 24
5.2.2 Integration Testing 25
5.2.3 System Testing 26
5.2.4 Test Result 27

6 RESULTS AND DISCUSSIONS 28
6.1 Efficiency of the Proposed System 28
6.2 VGG16 vs VGG19 Accuracy and Loss Comparison 28
6.3 Sample Code 29

7 CONCLUSION AND FUTURE ENHANCEMENTS 31
7.1 Conclusion 31
7.2 Future Enhancements 31

8 PLAGIARISM REPORT 32

9 SOURCE CODE & POSTER PRESENTATION 33
9.1 Source Code 33

9.2 Poster Presentation 34

References **35**

Chapter 1

INTRODUCTION

1.1 Introduction

In India, agriculture is the major source of the income. The agricultural sector employs the majority of the country's people, either directly or indirectly. As a result, creating high-quality agricultural yields is essential for the country's economic progress to continue. Farmers choose the proper products by monitoring and maintaining the necessary temperature, light, and humidity needs in order to generate harvests of higher quality and productivity. Furthermore, due to population expansion, weather fluctuations, and political instability, the agriculture business has begun to explore for new ways to enhance food production. Farmers are still grappling with issues such as early detection of plant diseases because observing this sort of disease on a plant's leaf with the naked eye is not always possible, an automated expert system that can help detect the disease in a timely manner would be very valuable. Advances in technology, particularly the use of image processing in conjunction with a machine learning technique, will aid farmers in detecting plant disease in its early stages.

Turmeric is the dried rhizome of Curcuma longa, a perennial herbaceous plant in the Zingiberaceae family. Turmeric contains up to 3 percent of curcumin, the most physiologically active phytochemical component. Its high curcumin concentration. Indian turmeric is regarded as the greatest in the world. It's extracted and studied for its well-known medicinal properties. The creation of an automated system for turmeric disease classification is motivated by the goal of integrating the Information and Communications Technology (ICT) with agriculture sector. Leaf spot,leaf blotch, and rhizome rot are the three diseases that damage turmeric leaves. Each of these diseases has its own cause and symptoms, which are listed below. Turmeric leaf spot is the most common disease that affects the plant. It has become a key stumbling block to successful turmeric cultivation. Brown dots of varied sizes emerge on the upper surface of young leaves as symptom. Colletotrichumcapsici is the fungus that causes it. Small, oval, rectangular or irregular brown spots emerge on either side of the leaves as a symptom of leaf blotch disease, which quickly turn filthy yellow or dark brown. The fungus Taphrinamaculans is the culprit. This disease is caused by the fungus Pythiumgraminicolum. Turmeric plant biproducts are utilised in food preparation and health care on a regular basis. The most frequent turmeric plant diseases are Leaf Spot and Leaf Blotch. VGG19 is a CNN (Convolutional Neural Network) architecture used for sick picture classification and detection.

1.2 Aim of the Project

The aim of the project is to detect and classify the diseases in turmeric plant through image processing using CNN architecture. The main aim is to identify diseases in turmeric plant.

1.3 Project Domain

The timely identification and early prevention of crop diseases are essential for improving production. In this paper, CNN models are implemented to identify and diagnose diseases in plants from their leaves, since CNNs have achieved impressive results in the field of machine vision. Standard CNN models require a large number of parameters and higher computation cost. In this paper, we replaced standard convolution with depth separable convolution, which reduces the parameter number and computation cost. The implemented models were trained with the open dataset that consisting of 14 different plant species with 38 different categorical disease classes and healthy plant leaves. To evaluate the performance of this models, various parameters such as batch size along with dropout and different numbers of the epochs were incorporated.

1.4 Scope of the Project

The scope of the project is to get the accurate values of the disease using the data set we use and find what type of disease that the leaf is having.

Chapter 2

LITERATURE REVIEW

Aravindhan Venkataramanan et al.,[1] addressed the problem of detecting and classifying plant diseases using deep neural networks (DNNs). This begins with an introduction to the importance of agriculture and the impact of plant diseases on crop yield and food security. Then it highlights the traditional methods of disease detection and classification, which are time-consuming, expensive, and require specialized expertise. Next they introduces the concept of using DNNs for plant disease detection and classification, which can significantly improve the efficiency and accuracy of the process.

F. Jakjoud et al.,[2] addressed the use of deep learning algorithms for the detection of plant diseases. The proposed method begins with an introduction to the importance of plant disease detection in agriculture and the challenges associated with traditional methods. It highlights the potential of deep learning algorithms to improve the efficiency and accuracy of disease detection. It also provides a comprehensive review of the existing literature on plant disease detection using deep learning algorithms, which includes the use of convolutional neural networks (CNNs) and transfer learning.

G. Rama Mohan Reddy et al.,[3] proposed method begins with an introduction to the importance of plant disease detection and its impact on agriculture. This method highlights the traditional methods of disease detection, which are manual, time-consuming, and require specialized knowledge. This method then introduces the concept of using deep learning techniques, particularly CNNs, for automated disease detection. The proposed method involves collecting plant leaf images using a Raspberry Pi camera and pre-processing the images to enhance their quality.

Hardik kumar et al.,[4] addressed the problem of detecting and classifying plant leaf diseases using conventional machine learning and deep learning techniques. The proposed method begins with an introduction to the importance of plant leaf disease detection and classification in agriculture and the impact of these diseases on crop yield and food security. It highlights the traditional methods of disease detection, which are time consuming, expensive, and require specialized expertise. The proposed system then introduces the concept of using machine learning and deep learning for plant leaf disease detection and classification, which can significantly improve the efficiency and accuracy of the process.

Marko Arsenovic et al.,[5] focused on the use of deep neural networks for the recognition and classification of plant diseases using leaf images. The proposed system begins with an introduction to the importance of plant disease recognition and classification in agriculture and the impact of these diseases on crop yield and food security. It highlights the traditional methods of disease detection, which are time consuming, expensive, and require specialized expertise. The system then introduces the concept of using deep neural networks for plant disease recognition and classification, which can significantly improve the efficiency and accuracy of the process.

Mrs. Kavita Krishnat Patil et al.[6] proposed a method for the automated detection of leaf diseases using convolutional neural networks (CNNs) and image processing techniques. The proposed method begins with an introduction to the importance of plant disease detection and its impact on agriculture. It highlights the traditional methods of disease detection, which are manual, time-consuming, and prone to errors. The proposed method then introduces the concept of using deep learning techniques, particularly CNNs, for automated disease detection. This provides a comprehensive review of the existing literature on plant disease detection using CNNs and image processing techniques.

Murk Chohan et al.,[7] proposed method focuses on the use of deep neural networks for the detection of plant diseases using leaf images. This begins with an introduction to the importance of plant disease detection in agriculture and the impact of these diseases on crop yield and food security. The method highlights the traditional methods of disease detection, which are time-consuming, expensive, and require specialized expertise. This method then introduces the concept of using deep neural networks for plant disease detection, which can significantly improve the efficiency and accuracy of the process.

S. Sreeja et al.,[8] proposed method focuses on the use of image processing techniques for the automated detection of turmeric plant diseases. This method begins with an introduction to the importance of turmeric as a medicinal plant and the impact of leaf diseases on its growth and yield. It highlights the traditional methods of disease diagnosis, which are time-consuming, labour-intensive, and require specialized expertise. It then introduces the concept of using image processing techniques for disease detection, which can significantly improve the efficiency and accuracy of the process. It provides a comprehensive review of the existing literature on image processing techniques for disease detection, which includes the use of various methods such as segmentation, feature extraction, and classification.

Srdjan Sladojevic et al.,[9] proposed system focuses on the use of deep neural networks for the recognition of plant diseases by leaf image classification. It begins with an introduction to the importance of plant disease detection and the impact of these diseases on crop yield and food security. It highlights the traditional methods of disease diagnosis, which are time-consuming and require specialized expertise. It then introduces the concept of using deep neural networks for disease detection and classification, which can significantly improve the efficiency and accuracy of the process.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

Turmeric is plagued by numerous diseases throughout its growth method. Not finding its diseases at early stages might result in a loss in production and even crop failure. The foremost necessary issue is to accurately establish diseases of the turmeric plant rather than mistreatment multiple steps like image pre-processing, feature extraction ,classification within the standard methodology and the single-phase detection model is adopted to alter recognizing turmeric plant leaf diseases. To enhance the detection accuracy of turmeric diseases, a deep learning-based technique known as the improved YOLOV3-Tiny model is planned. To improve detection accuracy than YOLOV3-Tiny, this methodology uses residual network structure supported the convolutional neural network specially layers. The results show that the detection accuracy is improved within the planned model compared to the YOLOV3-Tiny model. It allows anyone to perform quick and correct turmeric leaf diseases detection. In this, major turmeric diseases like leaf spot, leaf blotch, and rootstalk rot square measured. Coaching and testing pictures square measure captured throughout each day and night and compared with numerous YOLO ways and quicker R-CNN with the VGG16 model. Moreover, the experimental results show that the CycleGAN augmentation method on turmeric leaf dataset supports a lot of for improving detection accuracy for smaller datasets and also the planned model has associate in nursing advantage of high detection accuracy and fast recognition speed compared with existing traditional models.

3.2 Proposed System

Compared to alternative models like YOLOV5 and every one of the alternative models, the model we have a tendency to square measure proposing offers a best results and correct results. The model we have a tendency to propose is the model that worls on VGG19 and VGG16 design. This has a tendency to square measure victimization google colab because the setting and giving dataset as pictures like thirty pictures and that this processes the unwellness as leaf sopt and leaf blotch. The fifteen pictures contain leaf spot and another pictures contain.

3.3 Feasibility Study

3.3.1 Economic Feasibility

At present, the standard technique of visual scrutiny in humans by visual scrutiny makes it not possible to characterize plant diseases. Advances in pc vision models provide quick, normalized, and correct answers to these issues. Classifiers may be sent as attachments throughout preparation. All you wish could be a internet association and a camera-equipped cellular telephone. The well-known business applications "I Naturalist" and "Plant Snap" show however this can be attainable each apps stand out sharing skills with customers additionally as building intuitive online social communities. In recent years,deep learning has semiconductor diode to perform in varied fields like image recognition, speech recognition, and tongue process. The utilization of the Convolutional Neural Network within the drawback of Plant Disease Detection has excellent results. CNN is recognized because the best technique for seeing. We have a tendency to contemplate the Neural design specifically quicker Region-Based Convolutional Neural networks (Faster R-CNN), Regionbased Convolution Neural Networks(R-FCN), and single-shot Multi box detector (SSD) every of the Neural design ought to be able to be incorporate with any feature exactor relying on the applying. Pre-processing of information is extremely vital to models for correct performance. Several infections (viral or fungal) typically will be is may be arduous to differentiate often sharing overlap of symptoms.

3.3.2 Technical Feasibility

To prevent losses, little holder farmers and passionate about a timely and correct crop malady diagnosing. During this study, a pre-trained CNN was fine-tuned, and the model was deployed on-line. The ultimate result was a plant disease detection app. This service is free, simple to use and needs simply a sensible phone and web association. Thus, the user's wants as outlined during this paper are fulfilled. A thorough investigation exposes the capabilities and limitations of the model. Overall,once valid in an exceedingly controlled setting, an accuracy of ninety seven two share is given. This achieved accuracy depends on variety of things including the stage of malady, malady sort, background information and object composition because of this, a collection of user pointers would be needed for business use, to make sure the expressed accuracy is delivered.

3.3.3 Social Feasibility

Augmentation and transfer learning during this case, proved beneficial to the model, serving to the CNN to generalize a lot of reliability whereas this improved the model's ability to extract options,it had been not enough once the model was presented with 'in field' imaging. During this case, the classifier ranked associate degree accuracy of simply forty four proportion the importance of diversifying the coaching dataset to incorporate alternative background knowledge, extra plant anatomy and varying stages of disease.Overall, this study is conclusive in demonstrating however CNNs is also applied to empower small-holder farmers in their fight against disease within the future, work ought to be targeted on diversifying coaching datasets and additionally in testing similar internet applications in real world things. Without such developments, the struggle against plant disease can continue

3.4 System Specification

3.4.1 Hardware Specification

• Processor:Intel Core x64 processor
• Ram :4GB
• Storage :500GB HDD/SDD
•GPU :2GB

3.4.2 Software Specification

•Operating System : Windows

3.4.3 Standards and Policies

Google Colab

If a person have got used Jupyter notebook antecedently, that person can quickly learn to use Google Colab. To be precise, Colab could be a free Jupyter notebook surroundings that runs entirely within the cloud. Most significantly, it doesn't need a setup and therefore the notebooks that you just produce will be at the same time emended by your team members just the manner you edit documents in Google Docs. Colab supports several standard machine learning libraries which might be simply loaded in your notebook.

- As a programmer, the following can be performed using Google Colab.
- Write and execute code in Python
- Document the code that supports mathematical equations
- Create/Upload/Share notebooks
- Import/Save notebooks from/to Google Drive
- Import/Publish notebooks from GitHub
- Import external datasets e.g. from Kaggle
- Integrate PyTorch, TensorFlow, Keras, OpenCV
- Free Cloud service with free GPU

VGG Visual Geometry GROUP

AlexNet came move into 2012 and it improved on the normal convolutional neural networks, therefore VGG is perceived as a successor of the AlexNet, however it had been created by a unique cluster named as Visual pure mathematics cluster at Oxford‘s and hence the name VGG, it carries and uses some concepts from it's predecessors and improves on them and uses deep convolutional neural layers to boost accuracy. Let's explore what VGG19 is and compare it with a number of the opposite versions of the VGG design and additionally see some helpful and sensible applications of the VGG design.

Chapter 4

METHODOLOGY

4.1 General Architecture for CNN

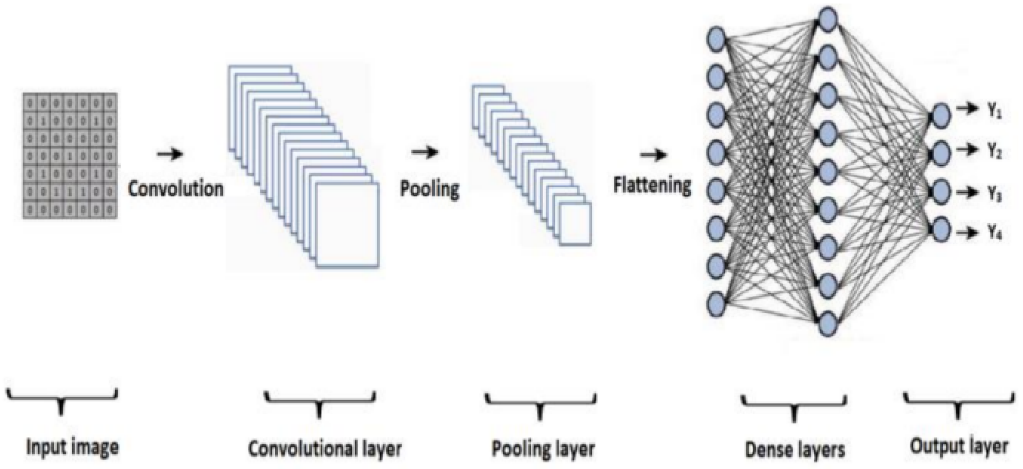


Figure 4.1: **CNN Architecture**

In Figure 4.1 displays the architecture of Convolutional Neural Networks. CNN stands for Convolutional Neural Network, a type of neural network widely used in image processing and computer vision applications. The architecture of a CNN typically consists of several layers, including convolutional layers, pooling layers, and fully connected layers.

4.2 Design Phase

4.2.1 Data Flow Diagram for Input Images

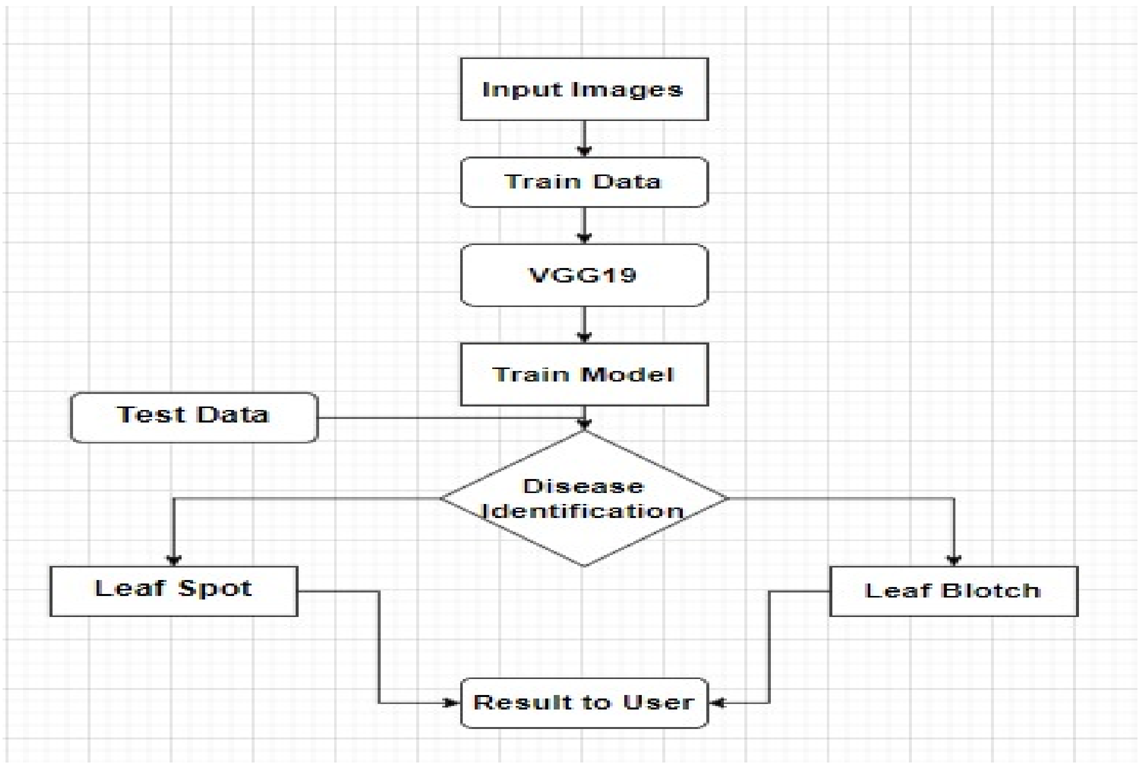


Figure 4.2: **Data Flow Diagram for Input Images**

In Figure 4.2, displays the data flow diagram. for this, the data is given in the format of images so they are input images. The images will get trained to the dataset, so the model gets trained by the dataset. The VGG architecture 19 helps the model in learning the dataset. The trained model gets the test data and it finds what type of disease the test image has got and it gives with the accurate values and results.

4.2.2 Use Case Diagram for Test Image

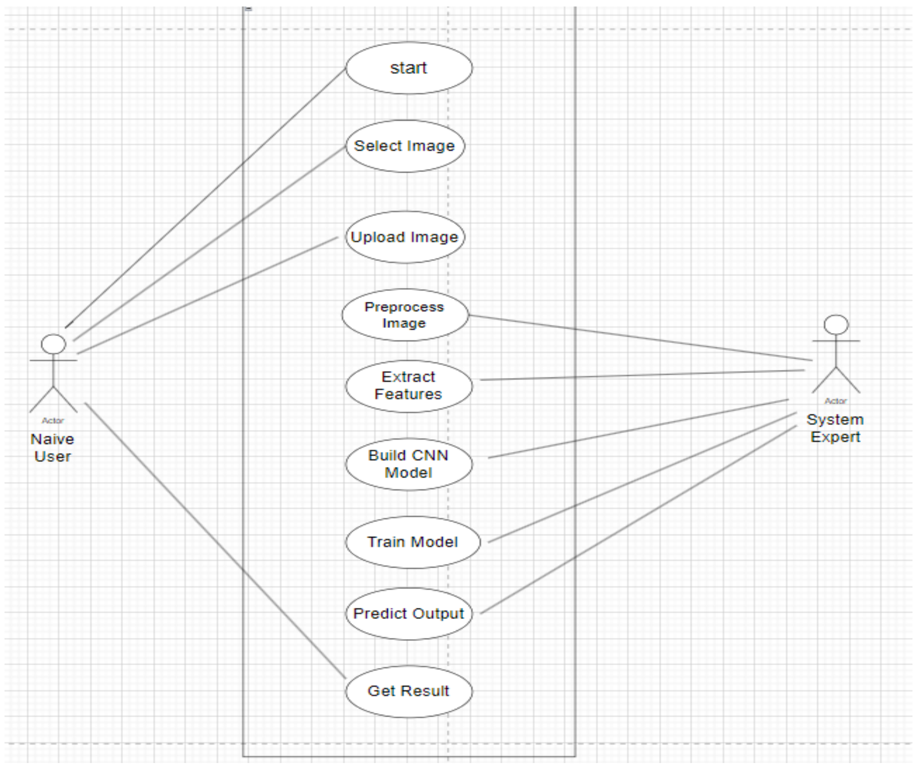


Figure 4.3: **Use Case Diagram for Test Image**

In Figure 4.3,displays the use case diagram. When we give a new input image first the module extracts the leaf features. Then it goes through the CNN model, it then compares the features with already trained dataset. Then it goes through dense CNN and the leaf features are extracted separately. Then the module will predict whether the plant leaf is affected by any disease or not. It shows the output from one of the 38 classes which are predetermined and trained. Then the output will be in a textual format.

4.2.3 Sequence Diagram for Disease Identification

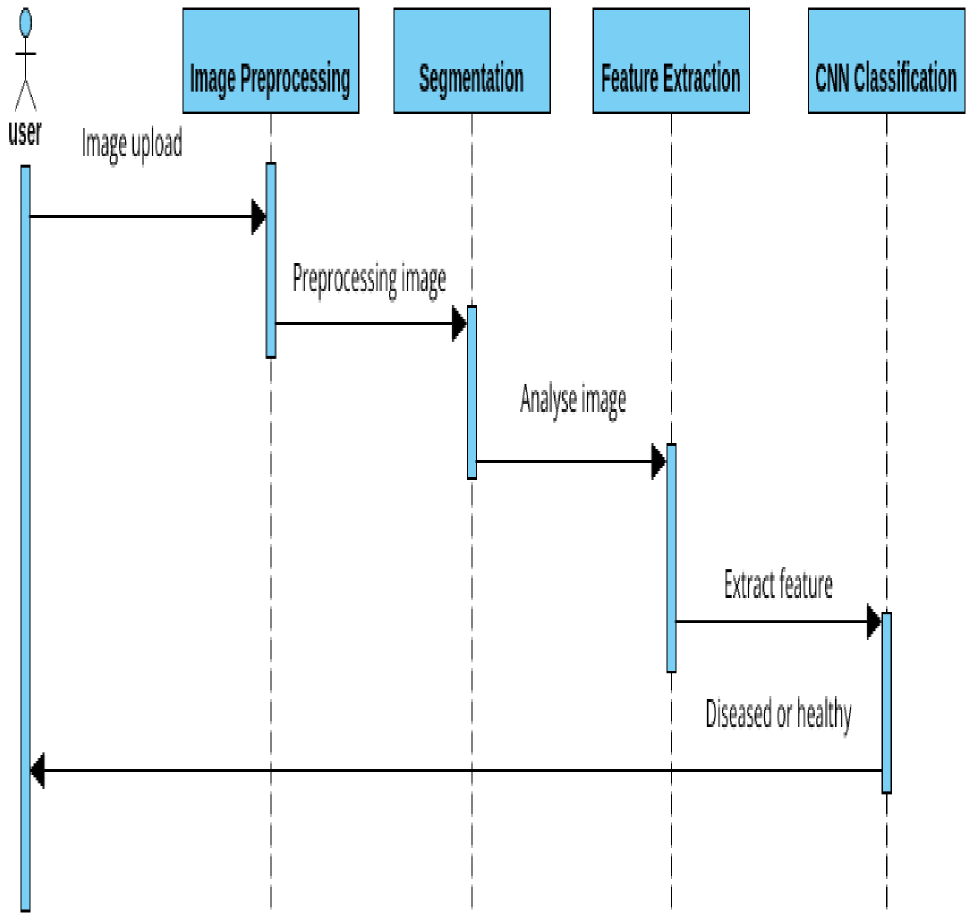


Figure 4.4: **Sequence Diagram for Disease Identification**

In Figure 4.4, displays the sequence diagram. For the model gives a perfect sequence of the data,this diagram gives a clear idea of how the data passes to the next step and explains what steps are linked together.

4.2.4 Activity Diagram for CNN Classification

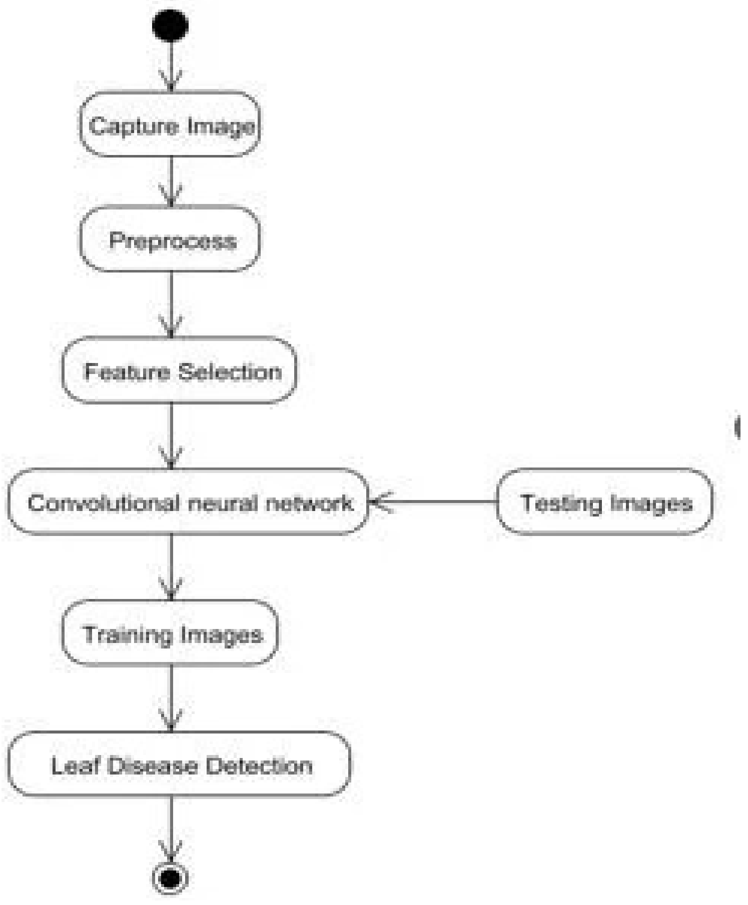


Figure 4.5: **Activity Diagram for CNN Classification**

In Figure 4.5, displays the Activity diagram which represets how the activity is been working in the model. When user intearacts with the VGG19 model, the user will upload the image of the disease affected plant and the image is been processed in multiple layers and then it shown the disease with accuracy percentage.

4.3 Algorithm & Pseudo Code

4.3.1 Enhanced Convolutional Neural Network

• **Step1:** Import TensorFlow and Keras libraries along with other required modules.
• **Step2:** Use the ImageDataGenerator function to set up generators for training, validation, and testing data, Set parameters such as zoom range, shear range, horizontal flip, and preprocessing function.
• **Step3:** Use the flow from directory function to create a generator for training data. Specify the target directory, target size, batch size, and class mode.
• **Step4:**‘ Use the same flow from directory function to create a generator for validation data. Use the same target directory and target size but specify a different batch size.
• **Step5:** Build a Sequential model. Add Convolutional layers, Pooling layers, Flatten layer, and Dense layers.
• **Step6:** Define the optimizer, loss function, and evaluation metric. Use the fit generator function to train the model.
• **Step7:** Visualize the training and validation accuracy and loss over epochs.
• **Step8:** Use the evaluate generator function to evaluate the model on the test data generator.

4.3.2 Pseudo Code

```
Import necessary libraries: numpy, pandas, matplotlib, os, keras, and relevant modules
from keras.preprocessing.image and keras.applications


 Set up image data generators for training and validation using the ImageDataGenerator function,with
      the zoom range, shear range, horizontal flip, and preprocessing function
Image Data Generator ( zoom range = 0.5,she arrange = 0.3 , horizontal
flip =True , pre processing function = pre process input)


 Create a training data generator using the flow from directory function, with
the target directory, target size, and batch size specified.
train = train.datagen flow from directory(directory =   / content / sample
data/train  d a t a  ,target size=(256,256),batch size =25)

 Create a validation data generator using the same flow from directory function, with the same
     target directory and target size but a different batch size.
val=train datagen.flow from directory( directory=  /content/ sample data/-
train  d a t a  ,target size = (256,256) batch size=25)
```

```

Obtain the first batch of images and their labels using the next() method of
the training data generator.
t img,label=train.next()

Define a function called plotimage that takes in an array of images and their
corresponding labels and plots them using matplotlib.
def plotimage(img arr,label);

Call the plotimage function with the first 10 images and labels from the
training set.
plot image(t img [:10],label [:10]
 The program should output a plot of the first 10 images and their labels.
test loss,test acc=model.evaluate(test generator)
```

4.4 Module Description

4.4.1 Image Acquisition

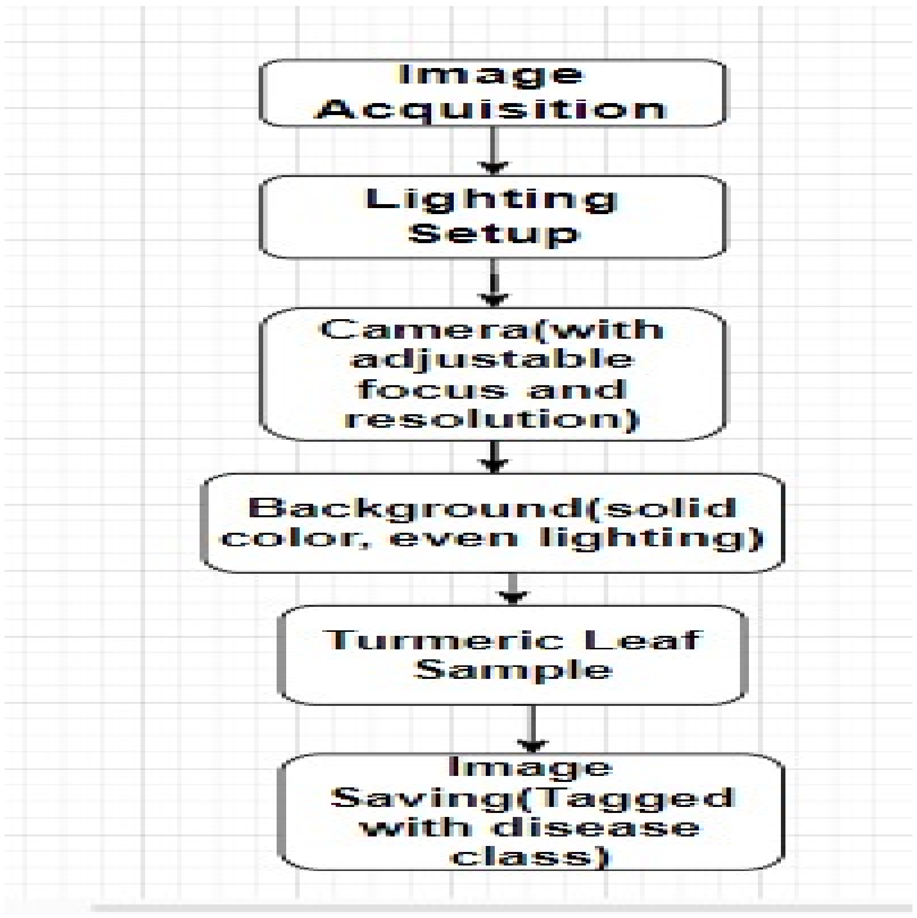


Image Acquisition
Lighting Setup
Camera(with adjustable focus and resolution)
Background(solid color, even lighting)
Turmeric Leaf Sample
Image Saving(Tagged with disease class)

Figure 4.6: **Image Acquisition**

In Figure 4.6, displays the Image Acquisition Process.
Lighting Setup: Use a diffused light source to minimize shadows on the leaf surface. A cloudy day outdoors or a softbox lighting setup indoors can achieve this. Avoid direct sunlight or harsh lighting that can cause uneven illumination.
Camera: A digital camera with adjustable focus and resolution is ideal. Set the camera to capture high-resolution images (around 12 megapixels or higher) to ensure sufficient detail for disease identification.
Background: Use a solid color background, preferably contrasting with the turmeric leaf color, for better image segmentation during pre-processing. White, green, or blue backgrounds are commonly used. Ensure even lighting on the background to avoid affecting the leaf image.
Turmeric Leaf Sample: Position the turmeric leaf sample on a flat surface in front

of the background. Ensure the entire leaf is in the frame and avoid overlapping leaves or obscuring parts of the leaf by stems or petioles.

Image Saving: Capture multiple images of the leaf sample from various angles and distances. It's helpful to capture close-up images of specific disease regions for better detection during training. Save the images in a common format like JPEG or PNG and label them with the corresponding disease class (e.g., healthy, leaf spot, etc.). This labeling is crucial for supervised deep learning models.

4.4.2 Image Pre-processing

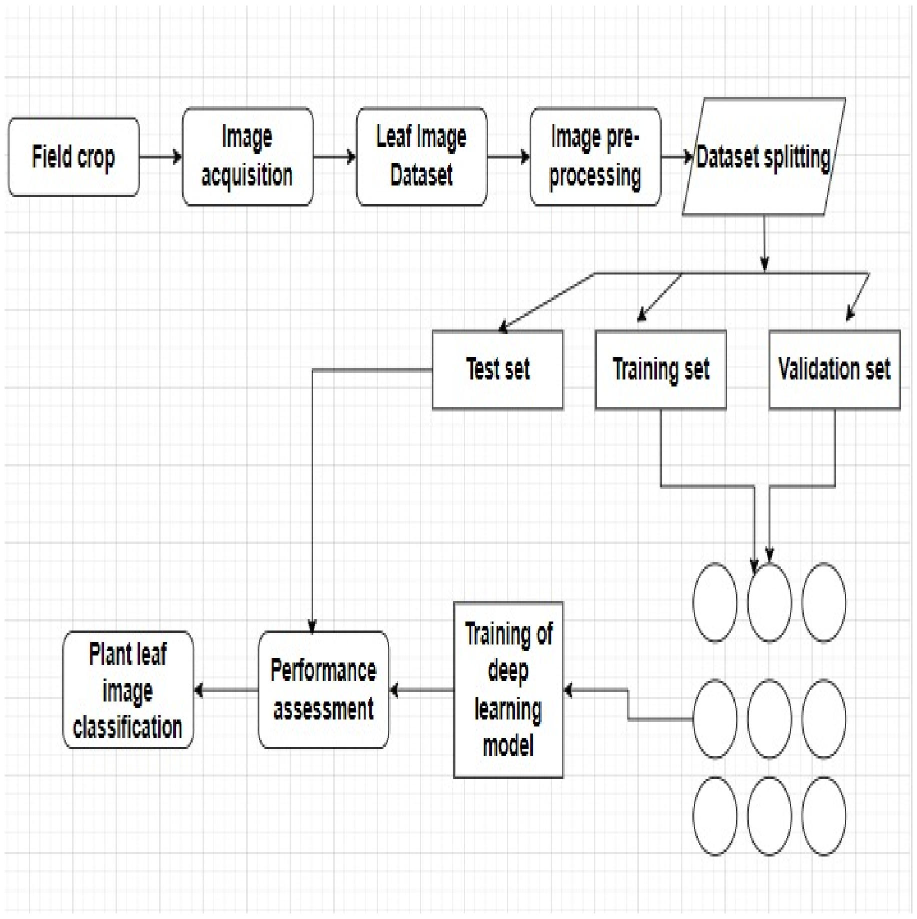


Figure 4.7: **Image Pre-processing**

In Figure 4.7, displays the Image Pre-processing.

Image Acquisition: Capture images of turmeric leaves under controlled lighting conditions (bright and even illumination) to minimize shadows and variations. Use a high-resolution camera to capture clear and detailed images of the leaves.

Resizing: Resize all images to a standard size to ensure compatibility with the deep learning model's input layer. This reduces computational complexity during training.

Color Normalization: Normalize the color channels of the images. This reduces the impact of lighting variations and camera inconsistencies on the model's performance.
Noise Reduction: Apply filtering techniques to remove noise from the images. Noise can be introduced during image acquisition or due to environmental factors.
Data Augmentation (Optional): Artificially create new images from existing ones through techniques like random cropping, flipping, rotation, and brightness adjustments. This helps increase the size and diversity of the training dataset, improving the model's robustness to variations in real-world images.
Normalization: Normalize the pixel values of the images to a specific range (often 0-1 or -1 to 1). This ensures all images have similar scales and prevents numerical issues during deep learning model training.

4.4.3 Image Segmentation

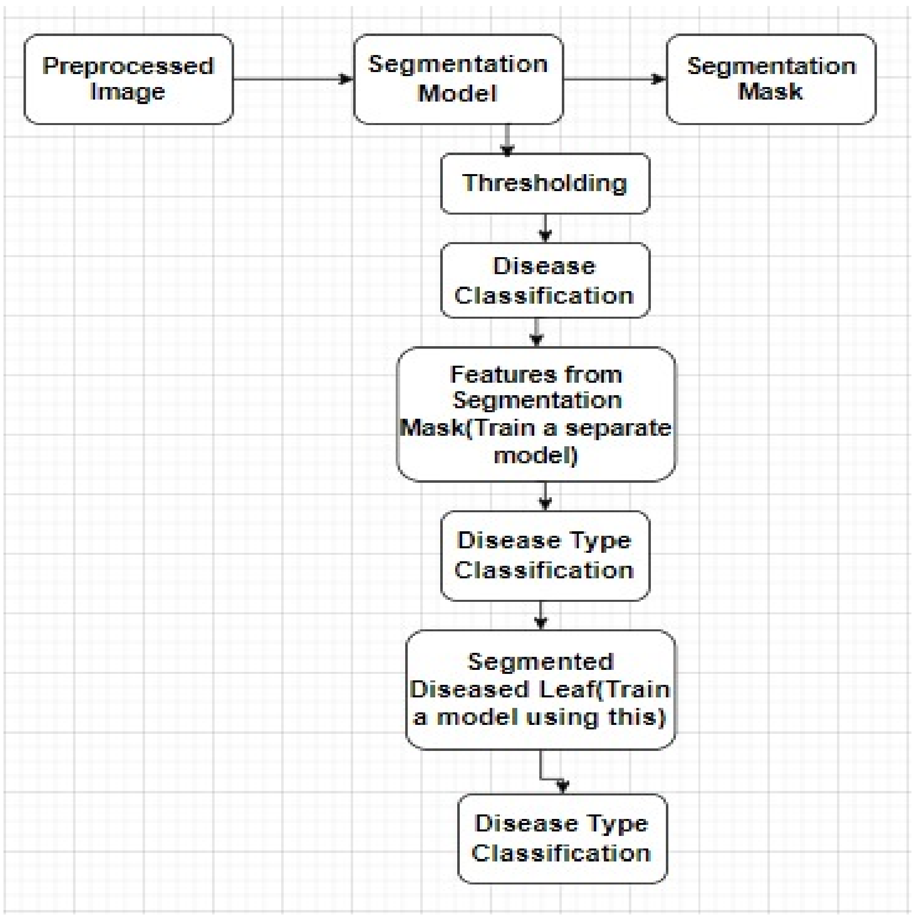

Preprocessed Image
Segmentation Model
Segmentation Mask
Thresholding
Disease Classification
Features from Segmentation Mask(Train a separate model)
Disease Type Classification
Segmented Diseased Leaf(Train a model using this)
Disease Type Classification

Figure 4.8: **Image Segmentation**

In Figure 4.8, displays the Image Segementation.
Preprocessed Image: This block represents the output from the image preprocessing

stage (resizing, normalization, etc.) as described before. **Segmentation Model:** This block represents a deep learning model specifically trained for image segmentation. Popular choices include U-Net, FCN (Fully Convolutional Network), or DeepLab. The model takes the preprocessed image as input and outputs a segmentation mask.
Segmentation Mask: This is a grayscale image where each pixel value represents the probability of that pixel belonging to a diseased region. High values indicate a higher likelihood of disease.
Thresholding: You can apply a threshold to the segmentation mask to convert it into a binary image (optional step). Pixels with values above the threshold are classified as diseased, while those below are classified as healthy.
Disease Classification: his is where the workflow splits into two possible paths:

Path 1 (Using Segmentation Mask): Train a separate deep learning model to classify the disease type based on the features extracted from the segmentation mask (probability of disease for each pixel). This path focuses on capturing specific information about the diseased regions.

Path 2 (Using Segmented Diseased Leaf): If you applied thresholding, the segmented diseased leaf can be used directly for disease classification. Train a model to classify the disease based on the image features of the segmented regions.
Disease Type Classification: This represents the final output, where the model predicts the specific disease affecting the turmeric plant leaf.
Choosing the Path: The choice between Path 1 and Path 2 depends on your specific needs and the complexity of the disease classification task.

Path 1might be beneficial if:

• You want to capture detailed information about the diseased regions.

• You're dealing with diseases with subtle variations that require a focus on specific areas.

Path 2might be a better option if:

• Computational efficiency is a concern, as training an additional model (Path 1) adds complexity.

• The disease classification task is simpler and the whole segmented region provides sufficient information.

4.5 Steps to execute/run/implement the project

4.5.1 Upload the IPYNB code in the GOCOLAB

once the google colab is on and upload the IPYNB code in the gocolab. Once the uploading is completed we can see the respective code that helps the model.

4.5.2 Create and upload the dataset

Create new folder in the sample data and name the foder as train data. Create two new folders in the train data and name the folders as LEAF SPOT and LEAF BLOTCH upload images of leaf spot and blotch in the respective folders and upload a new test image in the train data and name the image as test image.

4.5.3 Run the code in their respective fields

After uploading the training dataset, run the code in the respective fields.there are certain fields in the code that gives graphical output and some cases the output is the accuracy of the images. There is a code where we need to enter the path of the test image so that classifies the image and gives what type of disease the leaf has effected.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

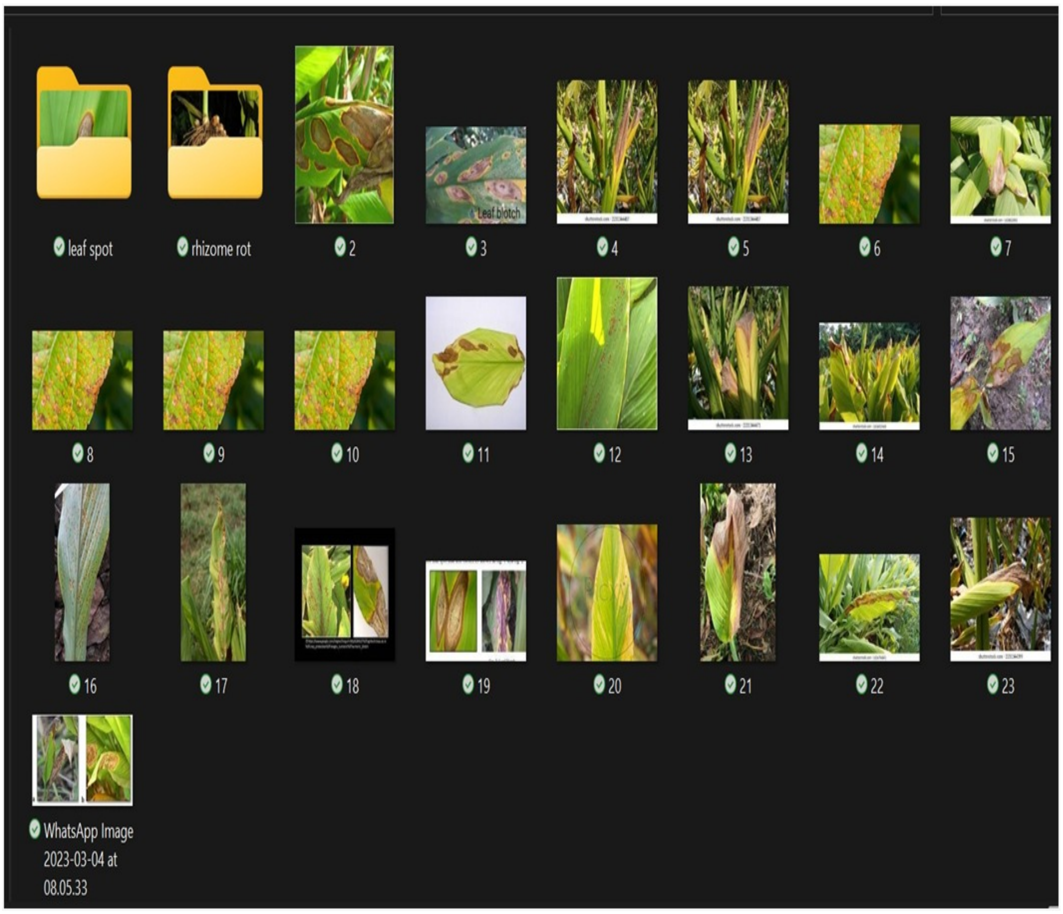
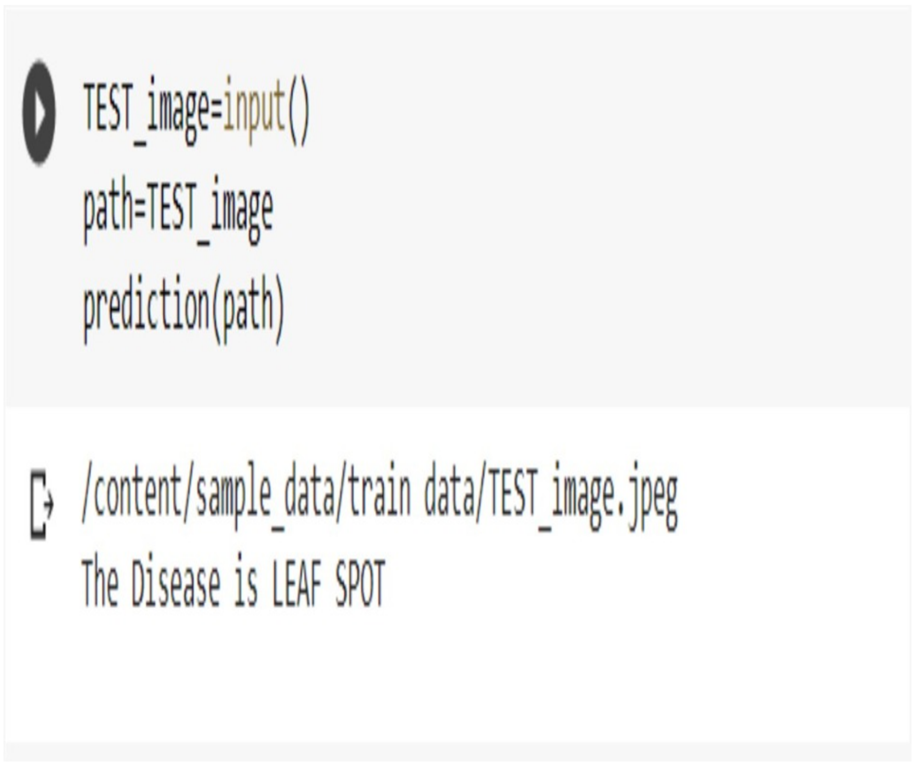


Figure 5.1: **Input Image of Dataset**

In Figure 5.1, displays the input images that will be used to train the dataset. In the given images there are thee types of diseases. Leaf spot, leaf blotch and rhizome rot. The leaf spot disease is caused by the fungus Colletotrichum capsici and is characterized by small, circular, brown spots on the leaves. The spots may coalesce and form larger patches, causing defoliation and reduced yield. The Turmeric leaf blotch disease is caused by the fungus Phaeoramularia curcuminis and is a serious foliar disease of turmeric. The Rhizome Rot disease is caused by the fungus Pythium aphanidermatum and is characterized by wilting, yellowing, and decay of the leaves. The disease can affect the rhizomes as well, leading to stunted growth and reduced yield.

5.1.2 Output Design



```
TEST_image=input()
path=TEST_image
prediction(path)
```

```
/content/sample_data/train data/TEST_image.jpeg
The Disease is LEAF SPOT
```

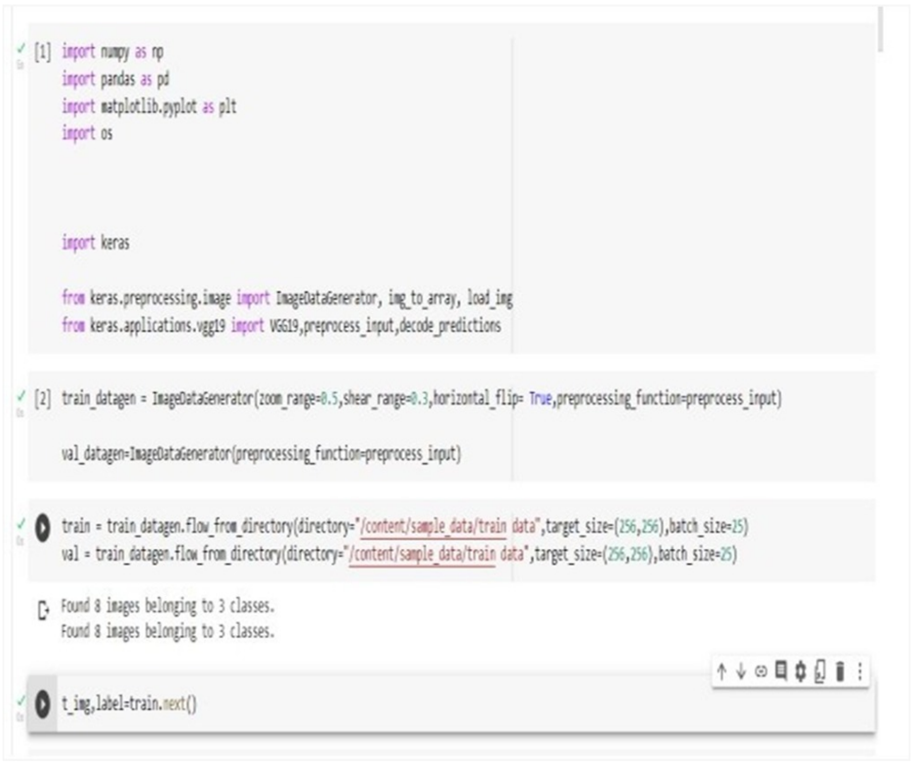
Figure 5.2: **Output Design of Disease Detected**

In Figure 5.2, displays the output. The output displays the output by showing what type of disease is detected. It uses the input dataset and gives a output based on the input that is given. Testing is a process of evaluating a software system or application to ensure that it meets its intended requirements, functions correctly, and performs as expected in different scenarios and conditions. The goal of testing is to identify any defects,errors, or issues that may affect the software's quality, reliability, and performance and to ensure that it meets the user's expectations. Testing can be performed at various stages of the software development life cycle, such as unit testing, integration testing, system testing, and acceptance testing. Each testing stage focuses on specific aspects of the software, such as functionality, performance, security, and usability, and uses different techniques and methods to evaluate the software's quality.

5.2 Types of Testing

5.2.1 Unit Testing

Input



```
[1] import numpy as np
    import pandas as pd
    import matplotlib.pyplot as plt
    import os



    import keras

    from keras.preprocessing.image import ImageDataGenerator, img_to_array, load_img
    from keras.applications.vgg19 import VGG19,preprocess_input,decode_predictions
```

```
[2] train_datagen = ImageDataGenerator(zoom_range=0.5,shear_range=0.3,horizontal_flip= True,preprocessing_function=preprocess_input)

    val_datagen=ImageDataGenerator(preprocessing_function=preprocess_input)
```

```
train = train_datagen.flow_from_directory(directory="/content/sample_data/train data",target_size=(256,256),batch_size=25)
val = train_datagen.flow_from_directory(directory="/content/sample_data/train data",target_size=(256,256),batch_size=25)
```

```
Found 8 images belonging to 3 classes.
Found 8 images belonging to 3 classes.
```

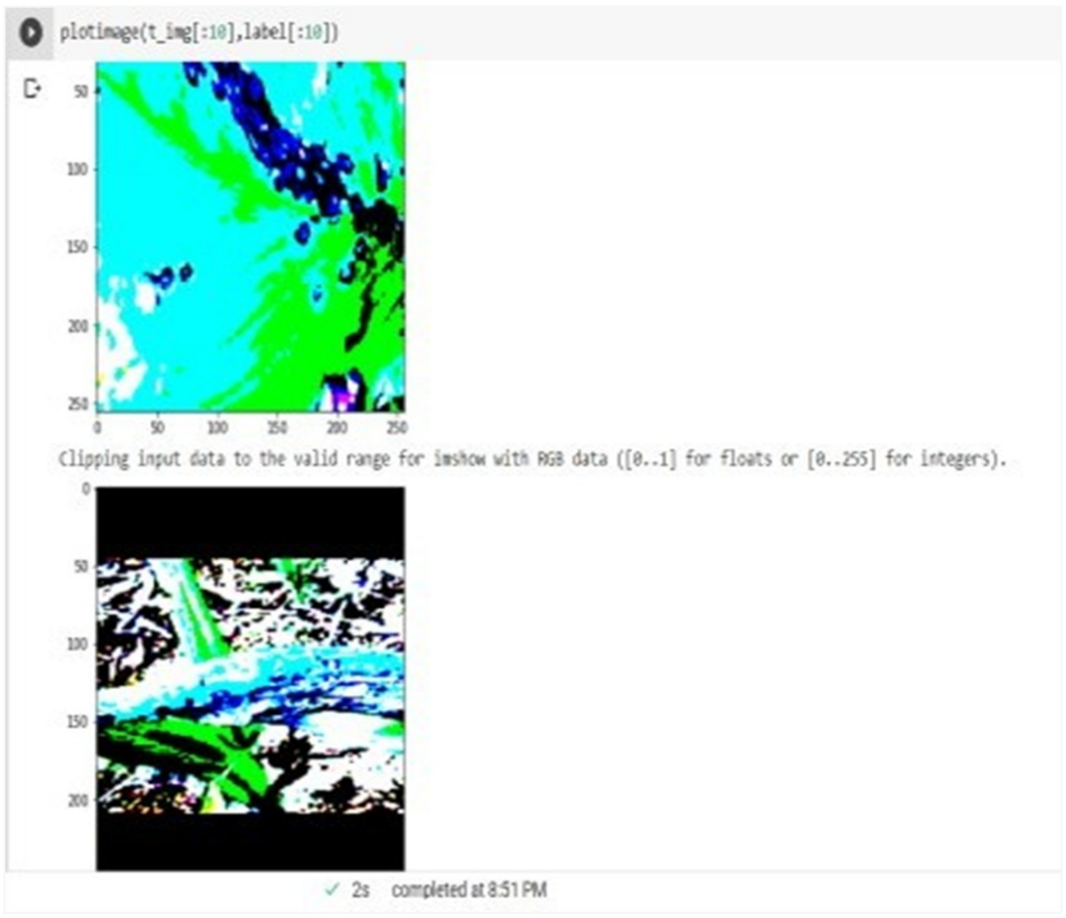
```
t_img,label=train.next()
```

Figure 5.3: **Unit Testing**

In Figure 5.3, displays unit testing result of our approach. Unit testing is a software testing method in which individual units or components of software are tested in isolation from the rest of the system to ensure that each unit is working as expected. A unit can be a function, method, or class. The goal of unit testing is to identify and fix defects early in the development cycle before the code is integrated into a larger system. Unit tests are typically automated and are run frequently during development to catch defects as soon as possible.

5.2.2 Integration Testing

Input



```
plotimage(t_img[:10],label[:10])
```

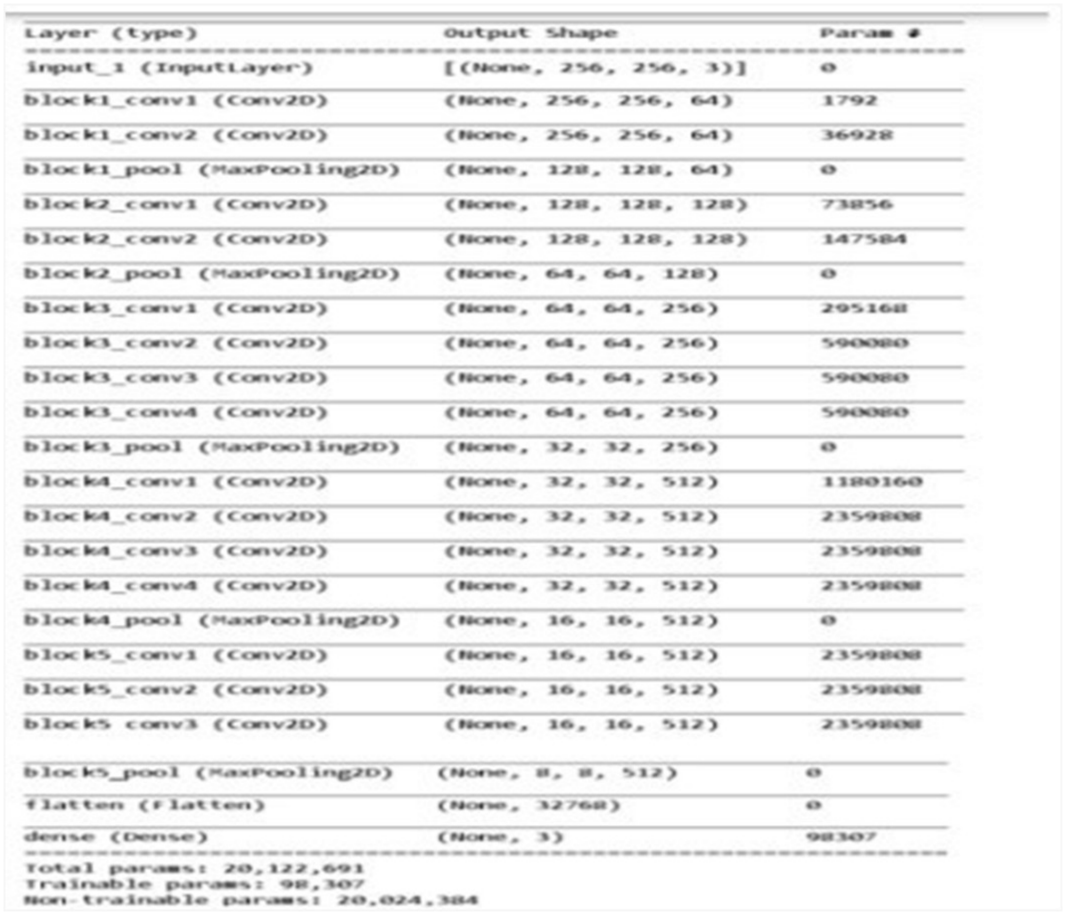
```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```

Figure 5.4: **Integration Testing**

In Figure 5.4, displays integration testing. Integration testing is a software testing method that tests the interactions between different modules or components of a system to ensure that they work together as expected. The goal of integration testing is to verify that the individual components of a system can work together and communicate with each other correctly. Integration testing can be done at different levels of the system, such as module, subsystem, or system level.

5.2.3 System Testing

Input

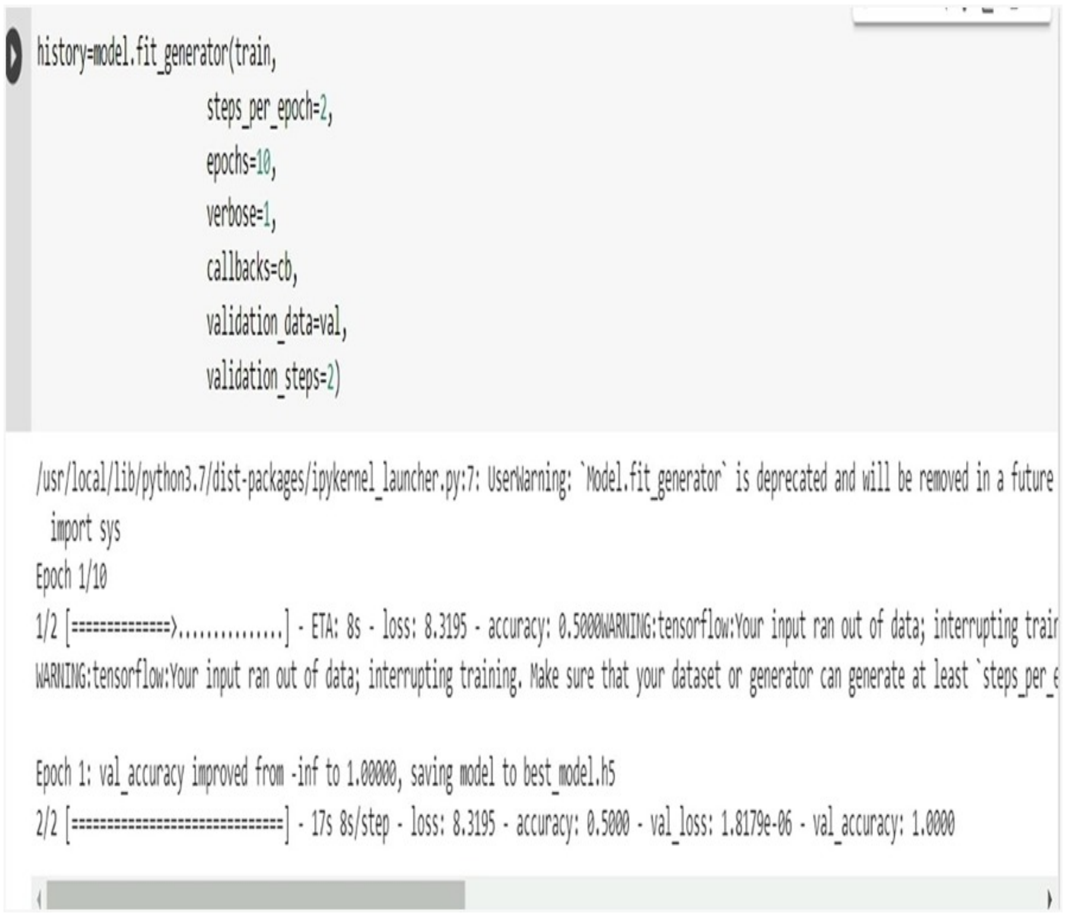


```
Layer (type)                    Output Shape              Param #
=================================================================
input_1 (InputLayer)            [(None, 256, 256, 3)]     0
block1_conv1 (Conv2D)           (None, 256, 256, 64)      1792
block1_conv2 (Conv2D)           (None, 256, 256, 64)      36928
block1_pool (MaxPooling2D)      (None, 128, 128, 64)      0
block2_conv1 (Conv2D)           (None, 128, 128, 128)     73856
block2_conv2 (Conv2D)           (None, 128, 128, 128)     147584
block2_pool (MaxPooling2D)      (None, 64, 64, 128)       0
block3_conv1 (Conv2D)           (None, 64, 64, 256)       295168
block3_conv2 (Conv2D)           (None, 64, 64, 256)       590080
block3_conv3 (Conv2D)           (None, 64, 64, 256)       590080
block3_conv4 (Conv2D)           (None, 64, 64, 256)       590080
block3_pool (MaxPooling2D)      (None, 32, 32, 256)       0
block4_conv1 (Conv2D)           (None, 32, 32, 512)       1180160
block4_conv2 (Conv2D)           (None, 32, 32, 512)       2359808
block4_conv3 (Conv2D)           (None, 32, 32, 512)       2359808
block4_conv4 (Conv2D)           (None, 32, 32, 512)       2359808
block4_pool (MaxPooling2D)      (None, 16, 16, 512)       0
block5_conv1 (Conv2D)           (None, 16, 16, 512)       2359808
block5_conv2 (Conv2D)           (None, 16, 16, 512)       2359808
block5 conv3 (Conv2D)           (None, 16, 16, 512)       2359808

block5_pool (MaxPooling2D)      (None, 8, 8, 512)         0
flatten (Flatten)               (None, 32768)             0
dense (Dense)                   (None, 3)                 98307
=================================================================
Total params: 20,122,691
Trainable params: 98,307
Non-trainable params: 20,024,384
```

Figure 5.5: **System Testing**

In Figure 5.5, displays system testing. System testing is a software testing method that tests the entire system as a whole, including all its components and their interactions. The goal of system testing is to ensure that the system meets all the functional and non-functional requirements.

5.2.4 Test Result



```
history=model.fit_generator(train,
                    steps_per_epoch=2,
                    epochs=10,
                    verbose=1,
                    callbacks=cb,
                    validation_data=val,
                    validation_steps=2)
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future
  import sys
Epoch 1/10
1/2 [==============>..............] - ETA: 8s - loss: 8.3195 - accuracy: 0.5000WARNING:tensorflow:Your input ran out of data; interrupting trair
WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps_per_e

Epoch 1: val_accuracy improved from -inf to 1.00000, saving model to best_model.h5
2/2 [==============================] - 17s 8s/step - loss: 8.3195 - accuracy: 0.5000 - val_loss: 1.8179e-06 - val_accuracy: 1.0000
```

Figure 5.6: **Test Image**

In Figure 5.6 ,displays the test image. The test image consists the data on how the image is tested and how the disease is identified and classified. A test image is a digital image used in image processing and computer vision applications for testing and evaluating algorithms, software, and hardware systems. Test images are designed to contain specific features or characteristics that allow the performance of the system to be measured and compared to a standard or benchmark.

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

Predicting the category for brand spanking new knowledge instances using our finalised classification model in Keras using the prediction path operate. Note that this operate is merely accessible on consecutive models, not those models developed victimization the practical API. There the output define are going to be The malady is. The trained model can analyze the input with the info pretrained and build a prediction. A ninety nine accuracy is achieved with the input. But with discomposed knowledge the model might not be correct. But the accuracy stays between 96percent

Existing System	Proposed System
VGG16	VGG19
Accuracy is 94 percent	Accuracy is 96 percent.
It has 16 Weight Layers	It has 19 Weight Layers
Image Input Size 224*224 pixel	Image Input Size 224*224 pixel
Size of layer is 41	Size of layer is 47

Comparision Between VGG16 VGG19

6.2 VGG16 vs VGG19 Accuracy and Loss Comparison

Existing system:(VGG 16))

In the Existing system, the implimentation to predict the category for brand spanking new information instances victimization the finalized classification model in Keras victimization the prediction(path) perform is done. Note that this performance is just offered on sequent models, not those models developed victimization the practical API. There the output define are going to be ”The unwellness is”. The trained model with vgg16 can analyze the input with the information pretrained and build a prediction. A 94.59459185600281 of accuracy is achieved with the input.

But with discomposed information the model may not be correct. But the accuracy stays between ninety 93 and 94.

Proposed system:(VGG 19)

The trained model with vgg19 can analyze the input with the information pretrained and build a prediction. The trained model with vgg19 can analyze the input with the information pretrained and build a prediction. A 97.222220897674 accuracy is achieved with the input. But with discomposed information the model may not be correct. But the accuracy stays between ninety six to ninety seven. The Base model had associate accuracy of 0.9614 with 0.81 loss wherever the created vgg16 model got a ninety four p.c accuracy with low loss. Here the vgg19 model has the whip hand with ninety seven p.c accuray with steady results as we tend to observe from graphs.

6.3 Sample Code

```
import numpy as np
import pandas as pd
importmat plotlib.py plot as plt
import os
import keras
from keras. preprocessing . image
i m p o r t ImageDataGenerator , img to array , loadimg
from keras.applications . vgg19
import VGG19 , preproces input , decode predictions traindatagen = Image Data Generator(zoom range =
      0.5, shear range = 0.3,horizontalflip= True , preprocessingfunction =preproessinput)
valdatagen = Image Data Generator(preproessingfunction= preprocess input)
```

Output for Disease Identification Accuracy Graph

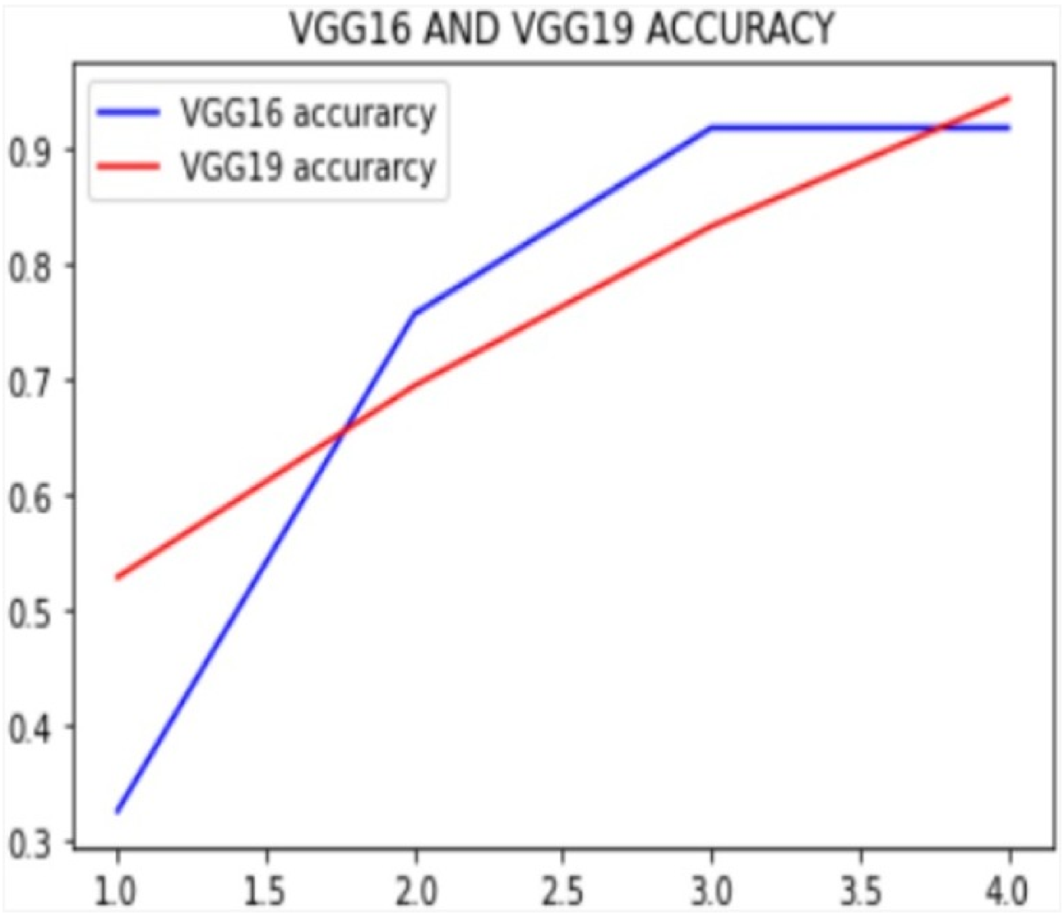


Figure 6.1: **Output for Disease Identification Accuracy Graph**

In Figure 6.1, displays the output of the code that is implemented and gives a graph as output giving out the accuracy in both VGG available.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

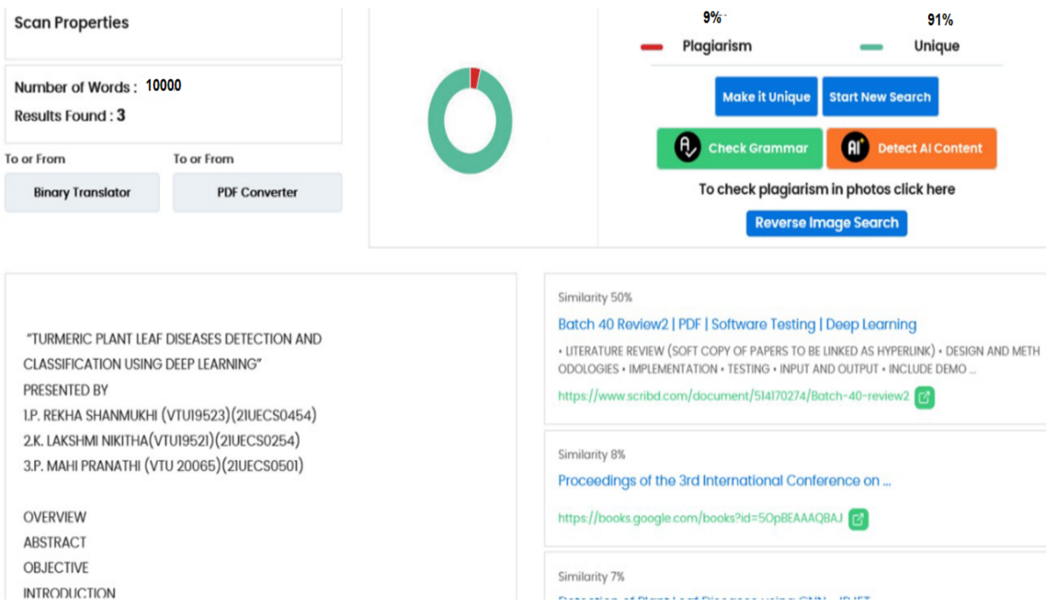
The annual lack of agricultural productiveness is as a result of intense plant sicknesses. As a result, diagnosing sicknesses in flora at an early level is vital for alerting such intense losses beforehand the future.The Monocotyledonous and Dicotyledonous plant families, in addition to the applicable method comprising the photograph processing phases had been included on this study. The maximum substantially used classification techniques for the identity and detection of sicknesses on plant leaves had been evaluated. The maximum current paintings has been tested and illustrated. It may be said that, some of the techniques utilised with in side the preceding paintings the deep studying principles, especially the CNN approach, have received the highest accuracy (96percent) .

7.2 Future Enhancements

In this have a look at a dataset created for 2 critical illnesses of turmeric plant the usage of pics received from the sector conditions, and some dataset for the crop became found. The pics from RGB colour area had been transformed into distinctive colour spaces. With the created dataset, a pre-educated deep studying version specifically VGG16 became used for education and validation. In addition, capabilities from the distinctive layers of VGG16 had been given to the MSVM for assessing the type efficiency. This have a look at has proved the prevalence of (RGB) discipline pics in which the type accuracy became maximum for the 2 illnesses. Surprisingly, it additionally furnished a competing accuracy withinside the case of pics educated with VGG16. VGG19 performed well than vgg16,SVM and ALEX net. Hence, VGG19 is proposed to this project.

Chapter 8

PLAGIARISM REPORT


Scan Properties

Number of Words : 10000

Results Found : 3

9% Plagiarism

91% Unique

To check plagiarism in photos click here

"TURMERIC PLANT LEAF DISEASES DETECTION AND CLASSIFICATION USING DEEP LEARNING"

PRESENTED BY

1.P. REKHA SHANMUKHI (VTU19523)(21UECS0454)

2.K. LAKSHMI NIKITHA(VTU19521)(21UECS0254)

3.P. MAHI PRANATHI (VTU 20065)(21UECS0501)

OVERVIEW

ABSTRACT

OBJECTIVE

INTRODUCTION

Similarity 50%

Batch 40 Review2 | PDF | Software Testing | Deep Learning

• LITERATURE REVIEW (SOFT COPY OF PAPERS TO BE LINKED AS HYPERLINK) • DESIGN AND METH ODOLOGIES • IMPLEMENTATION • TESTING • INPUT AND OUTPUT • INCLUDE DEMO ...

https://www.scribd.com/document/514170274/Batch-40-review2

Similarity 8%

Proceedings of the 3rd International Conference on ...

https://books.google.com/books?id=5OpBEAAAQBAJ

Similarity 7%

Figure 8.1: **Plagiarism Report**

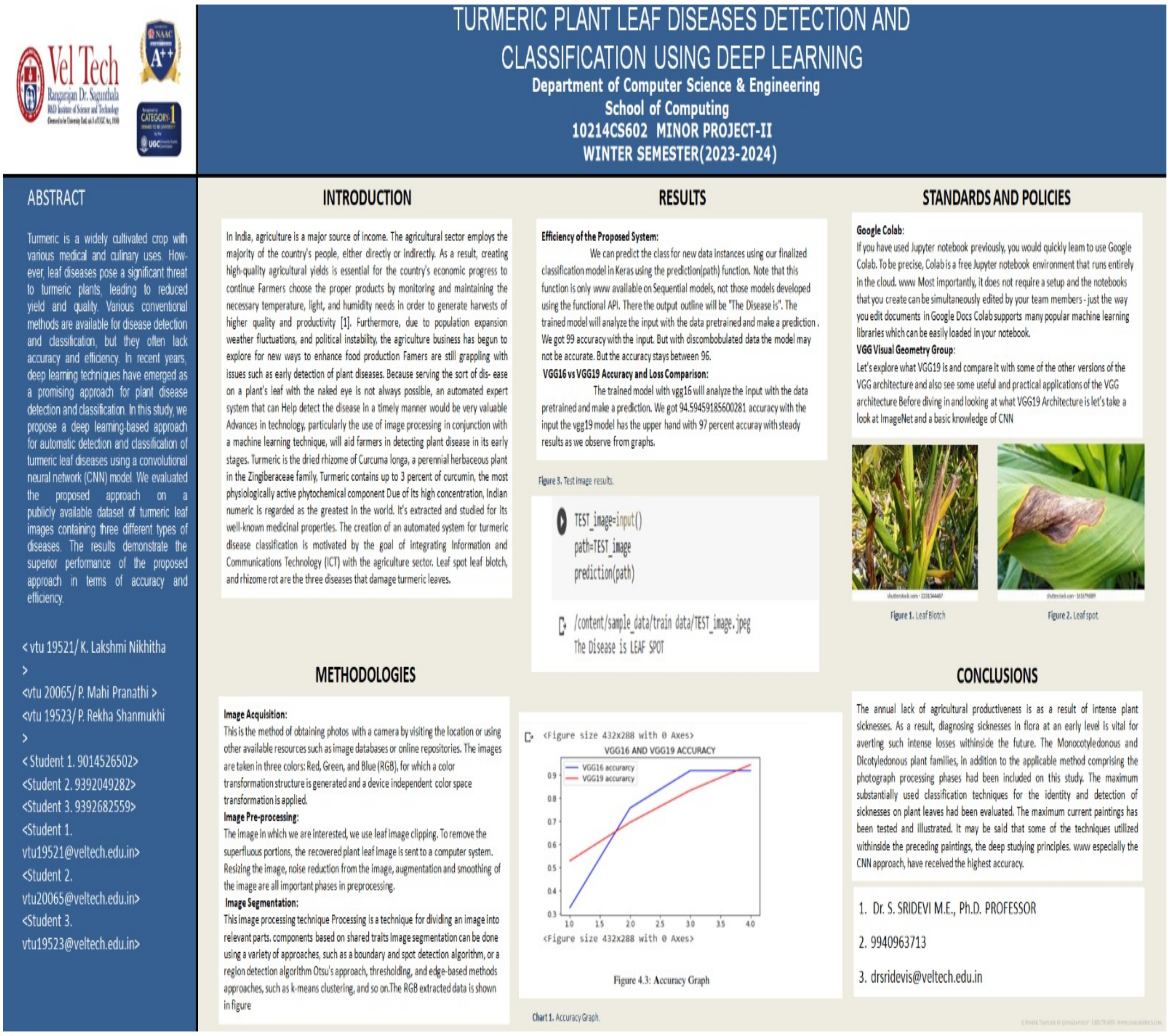
Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os
import keras
from keras.preprocessing.image import ImageDataGenerator, img_to_array, load_img
from keras.applications.vgg19 import VGG19, preprocess_input, decode_predictions

train_data_gen = ImageDataGenerator(zoom_range=0.5, shear_range=0.3, horizontal_flip=True,
    preprocessing_function=preprocess_input)
val_data_gen = ImageDataGenerator(preprocessing_function=preprocess_input)

train = train_data_gen.flow_from_directory(directory='/content/sample_data/train_data', target_size
    =(256, 256), batch_size=25)
val = train_data_gen.flow_from_directory(directory='/content/sample_data/train_data', target_size
    =(256, 256), batch_size=25)

def plot_image(img_arr, label):
    for im, l in zip(img_arr, label):
        plt.figure(figsize=(5, 5))
        plt.imshow(im)
        plt.show()

plot_image(train.next()[0][:10], train.next()[1][:10])train = train_data_gen.flow_from_directory(
    directory='/content/sample_data/train_data', target_size=(256, 256), batch_size=25)
val = train_data_gen.flow_from_directory(directory='/content/sample_data/train_data', target_size
    =(256, 256), batch_size=25)

def plot_image(img_arr, label):
    for im, l in zip(img_arr, label):
        plt.figure(figsize=(5, 5))
        plt.imshow(im)
        plt.show()
```

9.2 Poster Presentation



TURMERIC PLANT LEAF DISEASES DETECTION AND CLASSIFICATION USING DEEP LEARNING

Department of Computer Science & Engineering
School of Computing
10214CS602 MINOR PROJECT-II
WINTER SEMESTER(2023-2024)

Vel Tech
Rangarajan Dr. Sagunthala
R&D Institute of Science and Technology

ABSTRACT

Turmeric is a widely cultivated crop with various medical and culinary uses. However, leaf diseases pose a significant threat to turmeric plants, leading to reduced yield and quality. Various conventional methods are available for disease detection and classification, but they often lack accuracy and efficiency. In recent years, deep learning techniques have emerged as a promising approach for plant disease detection and classification. In this study, we propose a deep learning-based approach for automatic detection and classification of turmeric leaf diseases using a convolutional neural network (CNN) model. We evaluated the proposed approach on a publicly available dataset of turmeric leaf images containing three different types of diseases. The results demonstrate the superior performance of the proposed approach in terms of accuracy and efficiency.

< vtu 19521/ K. Lakshmi Nikhitha >
<vtu 20065/ P. Mahi Pranathi >
<vtu 19523/ P. Rekha Shanmukhi >
< Student 1. 9014526502>
<Student 2. 9392049282>
<Student 3. 9392682559>
<Student 1. vtu19521@veltech.edu.in>
<Student 2. vtu20065@veltech.edu.in>
<Student 3. vtu19523@veltech.edu.in>

INTRODUCTION

In India, agriculture is a major source of income. The agricultural sector employs the majority of the country's people, either directly or indirectly. As a result, creating high-quality agricultural yields is essential for the country's economic progress to continue Farmers choose the proper products by monitoring and maintaining the necessary temperature, light, and humidity needs in order to generate harvests of higher quality and productivity [1]. Furthermore, due to population expansion weather fluctuations, and political instability, the agriculture business has begun to explore for new ways to enhance food production Famers are still grappling with issues such as early detection of plant diseases. Because serving the sort of dis- ease on a plant's leaf with the naked eye is not always possible, an automated expert system that can Help detect the disease in a timely manner would be very valuable Advances in technology, particularly the use of image processing in conjunction with a machine learning technique, will aid farmers in detecting plant disease in its early stages. Turmeric is the dried rhizome of Curcuma longa, a perennial herbaceous plant in the Zingiberaceae family. Turmeric contains up to 3 percent of curcumin, the most physiologically active phytochemical component Due of its high concentration, Indian numeric is regarded as the greatest in the world. It's extracted and studied for its well-known medicinal properties. The creation of an automated system for turmeric disease classification is motivated by the goal of integrating Information and Communications Technology (ICT) with the agriculture sector. Leaf spot leaf blotch, and rhizome rot are the three diseases that damage turmeric leaves.

METHODOLOGIES

Image Acquisition:
This is the method of obtaining photos with a camera by visiting the location or using other available resources such as image databases or online repositories. The images are taken in three colors: Red, Green, and Blue (RGB), for which a color transformation structure is generated and a device independent color space transformation is applied.
Image Pre-processing:
The image in which we are interested, we use leaf image clipping. To remove the superfluous portions, the recovered plant leaf Image is sent to a computer system. Resizing the image, noise reduction from the image, augmentation and smoothing of the image are all important phases in preprocessing.
Image Segmentation:
This image processing technique Processing is a technique for dividing an image into relevant parts. components based on shared traits Image segmentation can be done using a variety of approaches, such as a boundary and spot detection algorithm, or a region detection algorithm Otsu's approach, thresholding, and edge-based methods approaches, such as k-means clustering, and so on.The RGB extracted data is shown in figure

RESULTS

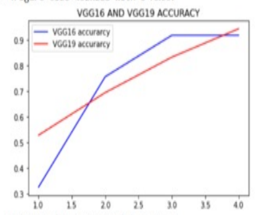
Efficiency of the Proposed System:
We can predict the class for new data instances using our finalized classification model in Keras using the prediction(path) function. Note that this function is only www available on Sequential models, not those models developed using the functional API. There the output outline will be "The Disease is". The trained model will analyze the input with the data pretrained and make a prediction . We got 99 accuracy with the input. But with discombobulated data the model may not be accurate. But the accuracy stays between 96.
VGG16 vs VGG19 Accuracy and Loss Comparison:
The trained model with vgg16 will analyze the input with the data pretrained and make a prediction. We got 94.59459185600281 accuracy with the input the vgg19 model has the upper hand with 97 percent accuray with steady results as we observe from graphs.

Figure 3. Test image results.

```
TEST_image=input()
path=TEST_image
prediction(path)
```

```
/content/sample_data/train data/TEST_image.jpeg
The Disease is LEAF SPOT
```

```
<Figure size 432x288 with 0 Axes>
```



```
<Figure size 432x288 with 0 Axes>
```

Figure 4.3: Accuracy Graph

Chart 1. Accuracy Graph.

STANDARDS AND POLICIES

Google Colab:
If you have used Jupyter notebook previously, you would quickly learn to use Google Colab. To be precise, Colab is a free Jupyter notebook environment that runs entirely in the cloud. www Most importantly, it does not require a setup and the notebooks that you create can be simultaneously edited by your team members - just the way you edit documents in Google Docs Colab supports many popular machine learning libraries which can be easily loaded in your notebook.
VGG Visual Geometry Group:
Let's explore what VGG19 is and compare it with some of the other versions of the VGG architecture and also see some useful and practical applications of the VGG architecture Before diving in and looking at what VGG19 Architecture is let's take a look at ImageNet and a basic knowledge of CNN

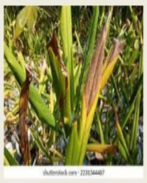


Figure 1. Leaf Blotch

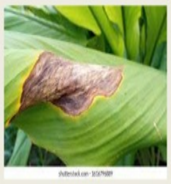


Figure 2. Leaf spot.

CONCLUSIONS

The annual lack of agricultural productiveness is as a result of intense plant sicknesses. As a result, diagnosing sicknesses in flora at an early level is vital for averting such intense losses withinside the future. The Monocotyledonous and Dicotyledonous plant families, in addition to the applicable method comprising the photograph processing phases had been included on this study. The maximum substantially used classification techniques for the identity and detection of sicknesses on plant leaves had been evaluated. The maximum current paintings has been tested and illustrated. It may be said that some of the techniques utilized withinside the preceding paintings, the deep studying principles. www especially the CNN approach, have received the highest accuracy.

1. Dr. S. SRIDEVI M.E., Ph.D. PROFESSOR
2. 9940963713
3. drsridevis@veltech.edu.in

Figure 9.1: **Poster Presentation**

References

[1] Aravindhan Venkataramanan, Deepak Kumar P Honakeri and Pooja Agarwal, “Plant Disease Detection and Classification Using Deep Neural Networks”, In- ternational Journal on Computer Science and Engineering (IJCSE), August 2019.

[2] F. JAKJOUD, A. HATIM and A. BOUAADDI, “Deep Learning application for plant diseases detection”, ITEE’19, El Jadida, Morocco, November 2019.

[3] G. Rama Mohan Reddy, Nettam Sai Sumanth and N. Sai Preetham Kumar, ”Plant Leaf Disease Detection Using CNN And Raspberry Pi”, IJASRET, Volume 5, Issue 2, February 2020

[4] Hardikkumar S. Jayswal and Jitendra P. Chaudhari,”Plant Leaf Disease Detection and Classification using Conventional Machine Learning and Deep Learn- ing”, International Journal on Emerging Technologies 11(3): 1094-1102, January 2020.

[5] Marko Arsenovic, “Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification”, Volume 2019, Article ID 3289801, June 2019.

[6] Mrs. Kavita Krishnat Patil , ”Leaf Disease Detection Using Image Processing By CNN”, IJIERT, VOLUME 8, ISSUE 8, August 2021.

[7] Murk Chohan, Rozina Chohan, Saif Hassan Katpar and Muhammad Saleem Mahar,” Plant Disease Detection using Deep Learning”, International Journal of Recent Technology and Engineering (IJRTE), Volume-9 Issue-1, May 2020.

[8] S. Sreeja, “ Automated detection of turmeric plant diseases using image process- ing techniques”, IJCSE, 2020.

[9] Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic, “Deep Neural Networks Based Recognition of Plant Diseases by Leaf Image Classification”, Computational Intelligence and Neuroscience, June 2020