DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING DIFFERENT TYPES OF PLANT DISEASES

Minor project-1 report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Design

By

J. VENKATESWARAO	(21UEDL0013)	**(VTU19860)**
K. GOPI CHAND	(21UEDL0017)	**(VTU20694)**
K. KASI REDDY	(21UECE0029)	**(VTU20592)**

Under the guidance of
Dr .K. SEETHALAKSHMI, M.E., Ph.D.,
ASSOCIATE PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & DESIGN
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

January, 2024

DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING DIFFERENT TYPES OF PLANT DISEASES

Minor project-1 report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science and Design

By

J. VENKATESWARAO	(21UEDL0013)	**(VTU19860)**
K. GOPI CHAND	(21UEDL0017)	**(VTU20694)**
K. KASI REDDY	(21UECE0029	**(VTU20592)**

Under the guidance of
Dr. K. SEETHALAKSHMI, M.E., Ph.D.,
ASSOCIATE PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & DESIGN
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

January, 2024

CERTIFICATE

It is certified that the work contained in the project report titled "DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING DIFFERENT TYPES OF PLANT DISEASES " by "J. VENKATESWARAO (21UEDL0013), K. GOPI CHAND (21UEDL0017), K. KASI REDDY (21UECE0029)" has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Dr. K. Seehtalakshmi
Associate Professor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
January, 2024

Signature of Head of the Department
Computer Science & Design
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
January, 2024

Signature of the Dean
Dr. V. Srinivasa Rao
Professor & Dean
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
January, 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.

J. VENKATESWARAO

Date: / /

K. GOPI CHAND

Date: / /

K. KASI REDDY

Date: / /

APPROVAL SHEET

This project report entitled “DEEP TRANSFER LEARNING MODEL FOR CLASSIFYING DIFFERENT TYPES OF PLANT DISEASES ” by J. VENKATESWARAO (21UEDL0013), K. GOPI CHAND (21UEDL0017), K. KASI REDDY (21UECEOO29) are approved for the degree of B.Tech in Computer Science and Design.

Examiners **Supervisor**

Dr. K. Seethalakshmi, M.E., Ph.D.,

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 & Design, Dr. R. PARTHASARATHY, 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. K. SEETHALAKSHMI, M.E., Ph.D., 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., Ms. 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.

J. VENKATESWARAO (21UEDL0013)
K. GOPI CHAND (21UEDL0017)
K. KASIREDDY (21UECE0029)

ABSTRACT

Increasing agricultural production is a critical requirement to meet demand due to the rapid population expansion. To prevent productivity loss, diseases must be discovered early. Manually monitoring leaf diseases is a laborious activity that involves extensive knowledge of plant pathogens, a lot of work, and long processing times. Our team have created and analysed a four different types of algorithms based on image processing, deep learning for these goals, and frequently produce meaningful findings. Using a paddy disease detection dataset having a comparative study of deep transfer learning (VGG16, InceptionV2, ResNet50, Mobile Net) models in terms of accuracy and disease name.

Keywords: Convolution neural network, Deep learning, Image processing, Pathogens, ResNet, Transfer learning.

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

CNN	Convolutional Neural Network
DL	Deep Learning
ResNet	Residual Neural Network
VGG	Visual Geometry Group

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Deep transfer learning models have emerged as powerful tools for the image classification tasks, including disease detection in plants. These models make use of neural networks that have already been trained, which enables them to quickly learn characteristics from huge datasets and perform well on more specific datasets. This comparison study will examine how, well various deep transfer learning models categorise various crop illnesses. The accuracy of various models, including VGG16, InceptionV2, MobileNet and ResNet will be assessed and compared. Also will also look into how other factors, such learning rate, batch size, and optimisation algorithms, affect the performance of the models. The study's findings will help identify the most suitable deep transfer learning model for disease classification in paddy crops and provide insights into optimizing the models performance. Ultimately, this research aims to contribute to the development of effective disease management strategies, leading to improved crop yields and better food security.

1.2 Aim of the Project

The aim of the deep transfer learning models for classifying different types of disease is identify the most accurate and effective model for disease detection and classification in crops.
The project aims to achieve the following specific objectives:
Collection of a comprehensive dataset of disease detection.Preprocessing of the dataset: Removal of noise.Normalization of image sizes.Division of data into training, validation, and testing sets. Development and training of multiple convolutional neural networks (CNNs).Evaluation and comparison of the models accuracy on the testing set.

1.3 Project Domain

The project domain for deep transfer learning models for classifying different types of disease in agriculture and computer vision.Agriculture is an important sector that feeds the growing population, and a crucial staple crop that contributes significantly to global food security. However, paddy crops are vulnerable to various diseases, which can have a significant impact on their productivity and quality.
By utilising computer vision techniques, the study intends to address this problem by reliably detecting and classifying the various crop illnesses. The study involves in collecting and preprocessing a large dataset of plant images, developing,training multiple deep transfer learning models, evaluating and comparing their performance, and identifying the most accurate and effective model. The method that use in this specific case is the CNN Algorithm with VGG16, InceptionV2 and MobileNet, ResNet for the prediction of the disease.

1.4 Scope of the Project

The project's goal is to help increase yields by preventing crop devastation. Many farmers wish to accept advanced agriculture, but many are unable to do so variety of reasons, including lack of knowledge about cutting edge technology and significant technical costs. It has frequently been noted that the plant diseases are challenging to manage because their prevalence varies depends on the environment. Many types of different diseases affect plants, such as leaf spot, dryness, colour changes, etc. Some of them might ruin the entire crop if not identified and treated in a timely manner. This could result in a farmer suffering a significant loss and lowering the yield of staple crops. On the basis of photographs of the leaves, people utilise the process of image processing algorithms to automatically identify the disease of the plant. This destruction of the crops or lower yields result from a lack of knowledge about diseases and inappropriate pesticide usage, among other factors. Moreover, find crop diseases before the spread to the crop's outer layers. The scope of the project is limited to the use of deep transfer learning models for disease classification in crops. The study does not cover other factors that may affect disease management, such as cultural practices, environmental conditions, or crop varieties.

Chapter 2

LITERATURE REVIEW

Houda Orchi et al., [1] proposed about the machine learning models,by stated the classification at their highest accuracy level is 98.01%.A dataset is drawn on village plants where increasing agricultural production is a critical requirement to meet demand due to the rapid population expansion as early detection of agricultural diseases is critical to preventing production loss.

Ahmed Samit Hatem et al., [2] develpoed a method by using CNN models that are GoogleNet, AlexNet, ResNet50 and VGG16. GoogleNet, AlexNet, VGG16, and ResNet50 have been determined to have accuracy of 98.57%, 98.81%, 99.05%, and 99.36% respectively. A data set is taken which contains maize or corn leaf data set. The classification and analysis of images now heavily rely on deep learning on the diseases in maize reduce productivity, which is a significant source of financial losses in the global agriculture industry. In the past, researchers have used hand crafted traits to categorise photos and spot leaf diseases in maize plants.

Manzhou Li et al., [3] studied by comparing the CNN, KNN and SVM and their best accuracy score in CNN is 99.6%. The most crucial part of agricultural production is safeguarding crop yields, and one of the most crucial steps in doing so is controlling crop pests and diseases. For this reason, it is essential to identify crop pests and diseases. With computer vision technology maturing in recent years, additional opportunities have been presented for applying plant disease detection.

Afsana Mimi et al., [4] implemented by the method of CNN, SVM and MobileNet comparitive study by using nearly 4000 images in a data set. Among these three models the highest accuracy is 97.35%. Monitoring a plant's health and well being visually involves keeping an eye on the state of the plant's leaves. Symptoms of serious underlying disorders including a lack of essential components like nitrogen, an excess of chloride particles.

Vinay Gautam et al., [5] proposed a methods of CNN based DL architecture, TL,

for agricultural research to help farmers. Disease detection in paddy plants was considered using InceptionV3, VGG16, ResNet, SqueezeNet, and VGG19. The suggested model outperformed modern models with various TL architectural variants, with an accuracy rate of 96.4%. The most important and consumable agricultural crop is the paddy crop. Paddy crops quality and productivity are impacted by leaf disease. Consequently, it is essential to address this problem as soon as possible in order to lessen its effects. As a result, diagnosing and categorising leaf disease has become increasingly important in recent years thanks to deep learning techniques.

Sunu Jatmika et al., [6] developed a method which is designed by using two approaches,modelling a convolutional neural network from scratch and modelling with transfer learning using the Inception v3 architecture. Based on the results of the testing, it is concluded that the system with the model created using the transfer learning approach produces good accuracy, with an accuracy of 90%. Meanwhile, the system with the other model achieves a 62% accuracy. This study examines the use of deep learning in mobile applications to categorise or locate illnesses in rice leaves. By presenting diagnostic results in the form of the disease name along with its taxonomy, disease description, and prescription suggestions for disease treatments, this system will make it simple for users to diagnose diseases.

Andrew J et al., [7] developed a method by using the pre trained models based on CNN for effective plant disease diagnosis. Reasearchers concentrated on optimising the hyper parameters of popular pre trained models as DenseNet-121, ResNet-50, VGG-16, and Inception V4.The experiments demonstrated that DenseNet-121 outperformed state of the art models in classification accuracy by 99.81%.The agriculture industry is crucial in delivering high quality food and contributes most to expanding economies and populations.Plant diseases have the potential to significantly reduce food output and wipe out species variety.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

The major downside in leaf identification is the need to describe leaf pattern. To describe the leaf form, numerous form choices have so far been retrieved. Unfortunately, once the leaf has been photographed and its characteristics have been determined, there is no accurate application to categorise.

The leaf disease prediction at an inopportune moment is explored by using the current system paper. To divide the contaminated area into its appropriate classes and segment it, a segmentation model based on colour was developed. For the example photos, experimental analysis has been performed in accordance with or in light of the time complexity and the affected region area.

3.2 Proposed System

The major goal of the proposed system is to identify plant leaf diseases using an feature extraction techniques that take form, colour, and texture into account. CNN is a method for categorising plant leaves as healthy or diseased, and if it's a diseased plant leaf, CNN will identify the specific disease. It is made to suggest treatments for certain diseases that will aid in the growth of healthy plants and also to determine the best and highest accuracy rate.

3.3 Feasibility Study

The project's economic potential is evaluated in this phase. Due to the connection between the research and agriculture, some information must be considered, such as the potential harm to the pest control methods could do to the environment, Economic Feasibility, Technical Feasibility, Social Feasibility.

3.3.1 Economic Feasibility

To determine the economic feasibility of the deep transfer learning models for classifying different types of disease in paddy, there are several factors to consider. These include, the cost of collecting and preparing the data to train and validate the deep learning models is an important factor to consider.

Cost of computing resources, deep learning models require significant computing resources, including high performance GPUs and cloud computing services developing deep learning models requires special expertise and time. The cost of hiring machine learning experts or dedicating staff time to model development should be considered.

3.3.2 Technical Feasibility

The use of various approaches and strategies to stop or slow the spread of diseases in paddy crops is technically feasible for several types of diseases. The viability of these methods depends on a number of variables, such as the type of disease, its strength, the crop's stage, and the availability of resources, to prevent and control the spread of illnesses in crops, a comprehensive strategy that combines cultural, chemical, and genetic methods can be useful.

3.3.3 Social Feasibility

The term social feasibility for various diseases refers to the ability to really put different disease control techniques into reality in a way that is acceptable to the community and has minimal detrimental effects on their social and cultural activities. While adopting disease management methods, social feasibility is an important factor to consider because the success of these strategies frequently rely on community participation and support and involvement of the community in the application of disease management techniques in crops determines their social viability. As a result, it's critical to involve the community in decision making and to offer the instruction and training on the advantages of these techniques.

3.4 System Specification

3.4.1 Hardware Specification

- Processor : I5

• Storage : 15GB.
• RAM : 4GB.
• Internet connectivity.

3.4.2 Software Specification

• Operating System - Windows 10 or 11.
• TensorFlow - 2.4.1.
• numpy - 1.24.2.

3.4.3 Standards and Policies

Anaconda Prompt

Anaconda prompt is a type of command line interface which explicitly deals with the ML(MachineLearning) modules. And navigator is available in all the Windows,Linux and MacOS. The anaconda prompt has many number of IDE's which make the coding easier. The UI can also be implemented in python.

Standard Used: ISO/IEC 27001

Jupyter

It's like an open source web application that allows us to share and create an the documents which contains the live code, equations, visualizations and narrative text. It can be used for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning.

Standard Used: ISO/IEC 27001

Chapter 4

METHODOLOGY

4.1 General Architecture

Figure 4.1: **Architecture**

In the above Figure 4.1 shows about the architecture is the conceptual model that defines the structure, behavior, and more view of the system. The first step is to take the input leaf, preprocess the entire set of input photos, extract the leaf's features, and then categorise the leaves using those features. And collect the data of the training images from the database and compare the input leaf with the training data and classify the name of the disease.

4.2 Design Phase

4.2.1 Data Flow Diagram

Figure 4.2: **Data Flow Diagram**

In the above Figure 4.2 shows the above information is being gathered from websites and agriculture organisations, according to the data flow diagram. To produce an ideal data set, preprocessing is done after data collection. After training the model, users can test the input data leaf, and the testing result will classify the leaf disease.

4.2.2 Use Case Diagram

Figure 4.3: **Use Case Diagram**

In the above Figure 4.3 illustrate the use case diagrams the first step is to collect the data set and preprocess the whole data of input images and extract the features of the leaf and classify the leaves according to the features. And collect the data of the training images and testing images from the database and compare the input leaf and then testing result outcome will be categorized and classify the name of the disease .

4.2.3 Class Diagram

Figure 4.4: **Class Diagram**

In the above Figure 4.4 shows the class diagram the first step is going to collect the data from data set, and then collection of data is trained and testing with multiple images later it will predict the disease name and shows the output based on input leaf.

4.2.4 Sequence Diagram

Figure 4.5: **Sequence Diagram**

In the above Figure 4.5 shows the sequence diagram so first upload source image into image processing, which is followed by image segmentation before that can extract leaf's feature data. Following feature extraction, classification can be done using the name of the leaf disease.

4.2.5 Collaboration diagram

Figure 4.6: **Collaboration Diagram**

In the above Figure 4.6 shows the collaboration figure explain how the user can provide the model with the input image, after which the data is sent for image processing, where the features of the leaf can be extracted and then classified into various classes. After categorization, check the image, provide the model with the necessary data, and have the model output results for the user.

4.2.6 Activity Diagram

Figure 4.7: **Activity Diagram**

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm

Step1 : Start
Step2 : Firstly according to the project collect and prepare the dataset.
Step3 : By using the dataset divide the dataset into training set, validation set and testing set.
Step4 : Select a set of deep transfer learning models that are pre-trained on a large image dataset such as VGG16, InceptionV2, MobileNet, ResNet.
Step5 : Later remove the final layers of the pre-trained model.
Step6 : Replace them with new layers that are customized for the paddy disease classification task.
Step7 : Evaluate the performance of the selected on the testing set and compare the accuracy.
Step8 : Monitor the performance of the deployed model and update to improve its accuracy.
Step9 : Stop

4.3.2 Pseudo Code

```
    train set, val set, test set = load data( path to data directory )
num classes = len(train set.classes)
models = [VGG16, InceptionV2, MobileNet, ResNet]
for model in models:
model = load pretrained weights(model)
model = replace last layer(model, num classes)
train model(model, train set, val set)
best model = None
best accuracy = 0.0
for model in models:
accuracy = evaluate model(model, test set)
if accuracy greater than best accuracy:
best model = model
best accuracy = accuracy
test accuracy = evaluate model(best model, test set)
18
precision, recall, f1 score = calculate metrics(best model, test set)
for model in models:
accuracy = evaluate model(model, test set)
print( Model : , Accuracy: .format( model.name, accuracy.))
best model = fine tune model(best model, train set, val set)
deploy model(best model)
monitor model(best model)
```

4.4 Module Description

4.4.1 Pre processing

The VGG model, commonly known as VGGNet, is referred to as VGG16. Very Deep Convolutional Networks for Large-Scale Image Recognition is a CNN model with 16 layers. In ImageNet, a dataset that contains more than 14 million training photos over 1000 item classes, the VGG16 model can reach a test accuracy of 92.7 percent. VGG16 enhances AlexNet by substituting sequences of smaller 33 filters for the big filters. For the first convolutional layer in AlexNet, the kernel size is 11, while for the second layer, it is 5.

The smaller convolutional filter used by VGG optimises the likelihood that the network would over-fit during training activities. The best size for a filter is 33, as lower sizes can't catch information from the left, right,up and down. Therefore,

VGG is the simplest model that may be used to comprehend the spatial properties of a picture. The network is simple to control thanks to consistent 33 convolutions.

• Input - An image of size 224x224 is fed into VGG16. By removing a 224*224 square from the centre of each image submitted for the ImageNet competition, the model's developers were able to maintain a constant image input size.
• Convolutional layers - The smallest 3*3 receptive field is used by the convolutional filters of the VGG algorithm. A 1*1 convolution filter is also used by VGG to linearly transform the input.

• ReLu activation - The Rectified Linear Unit Activation Function (ReLU) compone nt, a key improvement made by AlexNet to shorten training times, comes next. ReLU is a linear function that outputs zero for negative inputs and a matching output for positive inputs.

• Hidden layers - ReLU is used throughout the whole VGG network's hidden layers in place of AlexNet's Local Response Normalisation. With no improvement in overall accuracy, the latter method lengthens training sessions and uses more memory.

• Pooling layers - The number of parameters and dimensionality of the feature maps produced by each convolution phase are decreased by adding a pooling layer after a series of convolutional layers. Given the quick increase in the number of possible filters from 64 to 128, 256, and finally 512 in the last layers, pooling is essential.

• Fully connected layers - Three completely interconnected layers make up VGG16.Each of the first two layers contains 4096 channels, while the third layer contains 1000 channels one for each class.

4.4.2 Segmentation

The CNN architecture is used in the Inception V2 model. The original Inception model, which was created to increase the effectiveness and precision of image recognition tasks, has been upgraded using this model. Input Layer, the input

layer receives the image information, which is typically 299x299x3 width x height x channels in size. Stem layer, following batch normalisation and a ReLU activation function, the stem layer is composed of a 3x3 convolutional layer. After that, a further 3x3 convolutional layer, batch normalisation, and max pooling are applied.Inception modules, the foundation of the Inception V2 architecture are the Inception modules. Each module is made up of a number of parallel branches, each of which conducts a unique convolutional operation. The output of the module is created by joining these branches together. Max pooling operations are combined with 1x1, 3x3, and 5x5 convolutions in the network's first Inception module. The structure of succeeding modules is the same, but the number of branches and convolutional filters varies.Reduction modules,the Inception V2 architecture also has a number of reduction modules that are used to shrink the spatial size of the feature maps in addition to the conventional Inception modules. These modules often combine max pooling techniques with 3x3 convolutions.Classification layer, a fully connected layer with softmax activation makes up the network's top layer, which generates the output probabilities for all potential image classifications.

4.4.3 Furthur Extraction

A deep learning model called Residual Network is utilised in computer vision applications. It is a CNN architecture made to accommodate a large number of convolutional layers, possibly thousands. Previous CNN architectures could only support a few number of layers, which had a negative impact on performance.

The majority of ResNet models skip two or three layers at once, with batch normalisation and non linearity in between. Highway Nets, a type of more sophisticated ResNet architecture, can learn skip weights, which dynamically decide how many layers to skip.

The construction block is similar to the Inception network, which supports distinct 1*1 Conv, 3*3 Conv, 5*5 Conv, and MaxPooling transformations. The ResNeXt model adds and integrates these modifications, whereas the Inception model stacks them on top of one another.

Two 3x3 convolutional layers with batch normalisation and a ReLU activation function are applied to the input data, the input of the block shortcut connection is increased by the output of the second convolutional layer, another ReLU activation function receives the sum, the following residual block or the output layer receives the ReLU's output.

Figure 4.8: **ResNet Architecture**

4.4.4 Furthur Resolution

A deep convolutional neural network architecture called MobileNet was created for mobile devices and other contexts with limited resources. A set of depth wise separable convolutions make up the MobileNet architecture. A factorised convolution that separates the spatial and channel wise convolutions is called a depthwise separable convolution.

Additionally, MobileNet employs a method known as bottleneck layers, which lowers the number of channels in the network's intermediary layers. This lowers the network's computational expense while still enabling the network to collect crucial information. MobileNet is a well-liked option for embedded and mobile vision applications because it strikes a fair mix between accuracy and processing efficiency.

This continues for a while until the original image is reduced to 1024 channels but still pixels. After that, an average-pooling layer applies to the entire image and creates what appears to be an image but is actually only a vector with 1024 components.One filter is applied to each input channel using the depth-wise convolutions. Contrary to a standard convolution, in which the filters are applied to each input channel individually, this differs.
It does not combine them to create new features because depthwise convolution merely filters the input channel. In order to combine the output of depthwise convolution using a 1*1 convolution, a new layer is created and given the name pointwise convolution layer.

4.4.5 Classification

Data collection from agriculture resources, and web sources, then categorized them for training and testing.

4.5 Steps to execute/run/implement the project

4.5.1 Data Collection

•Import all the required modules.
• Load the dataset and define data transforms.
• Next pre-trained the deep learning models like VGG16, InceptionV2, MobileNet, ResNet are used as the base models.
• All the layers except the last layer are set to non-trainable.
• The weights of the base model are then fine-tuned by training the entire model with a higher learning rate for several epochs.

4.5.2 Testing The Model

Testing is done for the model whether the model is predicting the output correctly or not.
• Import all the required packages.

• Load the saved models.
• Load the test set and the same data transforms used in training are applied to preprocess the images.
• For each image in test set the model predicts the class label based on the output of the final fully connected layer to compare the metrics ike accuracy.
• If the performance of the best model is not satisfactory the model can be fine-tuned by adjusting the parameters like learning rate, number of epochs of the model.
• After fine-tuning the model is re-evaluated on the test set.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

Figure 5.1: **Input Design**

The above Figure 5.1 represents the input to the comparative analysis comprises of dataset, sizing 2.63GB. These datasets are divided appropriately between 4 models and reciprocated alternatively. After the preliminary process of data cleaning and clearing, the datasets are fed to the model Of these files 10,407 files are labelled paddy leaf images across 10 classes. The rest of the files are separated for testing the model for each of the diseases and final testing of an iteration.

5.1.2 Output Design

Figure 5.2: **Output Design**

The above Figure 5.2 represents a single picture is fed to the trained model, the model in turn would analyze and classify the batch and return the result in the form of a matrix. The use of accuracy matrix allows high efficiency detection and classification of variant diseases. The following table shows the comparative analysis of various Algorithm models that were chosen and experimented on respectively.

5.2 Testing

5.3 Types of Testing

5.3.1 Unit Testing

Each module is individually tested during unit testing, then the results are integrated with the total system. Unit testing concentrates verification efforts on themodule's tiniest unit of software design. Also known as module testing, this is. Each system module is tested independently. This testing is done right together with the code. Each module is found to function satisfactorily in the testing phase with relation to the module's anticipated output. There are also certain field validation tests.

5.3.2 Integration Testing

Data loss between interfaces is a possibility, and one module's negative impact on other subfunctions can make it so that, when combined, they don't provide the expected major functions. Systematic testing is known as integrated testing, and it is used to design and uncover interface flaws. With sample data, the testing was conducted. For these test data, the developed system has performed well. Finding the system's overall performance necessitates integrated testing.

5.3.3 System Testing

Each system module's integration has been tested through system testing. It is used to recognize differences between the system and its initial goal. It is discovered that the system documentation and current requirements are in accordance.

5.3.4 Test Result

Figure 5.3: **Test Image**

The above Figure 5.3 represents while the comparative analysis is the final stage is to research objective, the test result is the verifier.It depicts the real time results of the experiment and exploitative analysis of the 4 selected models.

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

The positive aspect of being able to identify any type of plant disease with the proposed system is due to the CNN accuracy, which can be trained with any type of plant image and identify the plant disease specific to that plant. In the proposed system, ourselves trained the input image data with an average accuracy of 95% using a convolution neural network. As pests continue to adjust their systems, the data for the remedies can be updated at any moment.

6.2 Comparison of Existing and Proposed System

6.2.1 Existing System

CNN is an algorithm that takes consideration of several characteristics such as leaf size and pixel intensity and groups them into a matrix, which has a certain value based on which the image is split into separate classes.

6.2.2 Proposed System

To compare the accuracy among these 4 models VGG16, Inception V2, Mobile Net, ResNet. A convolutional neural network with 16 layers is called VGG-16. Convolutional neural network Inception V2 was developed using training data from the ImageNet collection, which contains more than one million images. MobileNet creates a light weight deep neural network by using depthwise convolutions to drastically lower the amount of parameters compared to other networks.

6.3 Sample Code

```
1
2  VGG16 :
3  i m p o r t os
4  i m p o r t numpy as np
5  i m p o r t pandas as pd
6  i m p o r t m a t p l o t l i b . p y p l o t as p l t
7  i m p o r t s e a b o r n as s n s
8  i m p o r t t e n s o r f l o w as t f
9  from t e n s o r f l o w i m p o r t k e r a s
10 from t e n s o r f l o w . k e r a s i m p o r t l a y e r s
11 from t e n s o r f l o w . k e r a s . c a l l b a c k s i m p o r t ModelCheckpoint , E a r l y S
       t o p p i n g , ReduceLROnPlateau
12 from s k l e a r n . m e t r i c s i m p o r t a c c u r a c y s c o r e , p r e c i s i o n s c o
       r e , r e c a l l s c o r e , f 1 s c o r e
13 from s k l e a r n . m e t r i c s i m p o r t c l a s s i f i c a t i o n r e p o r t , c o n f u
       s i o n m a t r i x
14 i m p o r t random
15 d e f s e e d e v e r y t h i n g ( s e e d = 0) :
16 random . s e e d ( s e e d )
17 os . e n v i r o n [    PYTHONHASHSEED    ] = s t r ( s e e d )
18 np . random . s e e d ( s e e d )
19 t f . random . s e t s e e d ( s e e d )
20 s e e d = 0
21 s e e d e v e r y t h i n g ( s e e d )
22 BATCH SIZE = 32
23 IMG SIZE = 224
24 i n p u t p a t h =     . . / i n p u t / paddy     d i s e a s e     c l a s s i f i c a t i o n /

25 t r a i n d a t a d i r = i n p u t p a t h +     t r a i n i m a g e s /
26 t e s t d a t a d i r = i n p u t p a t h +     t e s t i m a g e s /
27 epochs = 50
28 h i s t o r y = model . f i t ( t r a i n d s ,
29 epochs = epochs ,
30 v a l i d a t i o n d a t a = v a l d s ,
31 c a l l b a c k s = [ t l c h e c k p o i n t 1 , e a r l y s t o p , r l r o p ] ,
32 v e r b o s e = 2)
33 acc = h i s t o r y . h i s t o r y [     a c c u r a c y     ]
34 p l t . p l o t ( e p o c h s r a n g e , l o s s , l a b e l =     T r a i n i n g Loss     )
35 p l t . p l o t ( e p o c h s r a n g e , v a l l o s s , l a b e l =     V a l i d a t i o n Loss
       )
36 p l t . l e g e n d ( l o c =     upper r i g h t     )
37 p l t . t i t l e (     T r a i n i n g and V a l i d a t i o n Loss     )
38 p l t . show ( )
39
40 s u b m i s s i o n . head ( )
41 I n c e p t i o n V2 :
42 i m p o r t numpy as np
```

```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sb
import cv2
from torch.utils.data import Dataset, DataLoader
from sklearn.model_selection import train_test_split
train, valid= train_test_split(traindata, test_size=0.05, random_state=0)
print(len(train))
print(len(valid))
train['label'].value_counts()
testdataset= PaddyTestDataset(testdataframe, root_dir='../input/paddy-disease-classification/
test_images',transform=transformvalid)
testloader= torch.utils.data.DataLoader(testdataset, batch_size=16, num_workers=2)
dfpreds= pd.DataFrame({'image_id':filenames, 'label':finalpredictions})
dfpreds.head()
testdf= pd.read_csv('../input/paddy-disease-classification/sample_submission.csv')
submissiondf= pd.merge(testdf[['image_id']], dfpreds, how='inner', on='image_id')
submissiondf.to_csv('submission.csv', index=False)
submissiondf.head()
print(len(submissiondf))
ResNet :

import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sb
import os
import warnings
warnings.filterwarnings('ignore')
from glob import glob
import tensorflow as tf
from tensorflow.keras.layers.experimental.preprocessing import RandomFlip
import tensorflow.keras.layers as tfl
from tensorflow.keras.preprocessing import image_dataset_from_directory
df= pd.read_csv('../input/paddy-disease-classification/train.csv')
df.shape
df.head()
df.groupby('variety').mean()
plt.figure(figsize=(10,10))
sb.countplot(data=df, x='variety')
def get_model(base, preprocessor, img_size):
```

```
inputs=tf.keras.Input(shape=(imgsize,imgsize,3))
x = RandomFlip(    horizontal    )(inputs)
x=preprocessor(x)
x=base(x)
x=tfl.Flatten()(x)
x=tfl.Dense(1024,activation=    relu    )(x)
x=tfl.BatchNormalization()(x)
x=tfl.BatchNormalization()(x)
x=tfl.Dense(64,activation=    relu    )(x)
x=tfl.Dropout(0.3)(x)
x=tfl.BatchNormalization()(x)
outputs=tfl.Dense(10,activation=    softmax    )(x)
model=tf.keras.Model(inputs,outputs)
return model
testdataset=imagedatasetfromdirectory(
    ../input/paddy    disease    classification/testimages
        ,
imagesize=(imgsize,imgsize),
batchsize=64,
shuffle=False,
labels=None,
classnames=None,
labelmode=None,
colormode=    rgb    ,
)
predictions=modeltuned.predict(testdataset)
predictions.shape
ss=pd.readcsv(    ../input/paddy    disease    classificat
    ion/samplesubmission.csv    )
ss[    label    ]=np.argmax(predictions,axis=1)
ss[    label    ]=ss[    label    ].replace([0,1,2,3,4,5,6
    ,7,8,9],labels)
ss.tocsv(    Submission.csv    ,index=False)
ss.head()
```

Output

Figure 6.1: **Comparision of 4 Deep Learning Models and the system**

The above Figure 6.1 represents the comparing of 4 CNN models that are VGG16, InceptionV2, MobileNet, ResNet. Among these 4 models have been determined to have accuracy rates of 91.5 %, 90.09 %, 96.10 %, 95.6 % respectively.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

There are numerous methods for identifying plant diseases and offering treatments. Each offers advantages as well as drawbacks. While visual analysis is the simplest and least expensive method, it is less effective and trustworthy. The technology of image processing is most frequently mentioned for its extremely high accuracy and minimal time commitment. The suggested methods primary objective is to accurately and efficiently identify the disease. The experimental findings show that the suggested strategy is a worthwhile strategy, capable of considerably assisting a precise diagnosis of leaf diseases with minimum computational work. The farmers also require access to reliable information that can utilise for effective crop management, and there is no better way to meet their needs than by offering them a service that can access through software. So by comparing the 4 models are VGG16, Inception V2, ResNet, Monile Net the best and highest accuracy is ResNet.

7.2 Future Enhancements

As new diseases emerge in the future and farmers develop the ability to recognise them but not their treatments, one can can test the disease, update all the diseases in the data set, train the model, and update the treatments. Once there discovered, everyone can update the diseases of the various crops to assist farmers in identifying the numerous diseases that affect their harvests. Later can improve by training many data sets at once and updating the model at predetermined intervals to incorporate fresh characteristics.

Chapter 8

PLAGIARISM REPORT

Figure 8.1: **Plagiarism Report**

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
VGG16 :
import os
import numpy as np
import pandas as pd
importmatplotlib.pyplot as plt
importseaborn as sns
importtensorflow as tf
from tensorflowimportkeras
from tensorflow.kerasimportlayers
from tensorflow.keras.callbacksimport ModelCheckpoint , EarlyS
    topping , ReduceLROnPlateau
from sklearn.metricsimportaccuracyscore , precisionsco
    re , recalllscore , f1score
from sklearn.metricsimportclassificationreport , confu
    sionmatrix
import random
def seedeverything ( seed = 0) :
random . seed ( seed )
os . environ [ PYTHONHASHSEED ] = str ( seed )
np . random . seed ( seed )
tf . random . settseed ( seed )
see d = 0
seedeverything ( seed )
BATCH SIZE = 32
IMG SIZE = 224
inputpath = ../input/paddy disease classification/

traindatadir=inputpath+ trainimages/
testdatadir=inputpath+ testimages/
epochs = 50
history = model . fit ( trainds ,
epochs = epochs ,
validationdata=valds ,
callbacks=[ tlcheckpoint1 , earlystop , rlrop ] ,
verbose = 2)
```

```
acc = history.history[ accuracy ]
plt.plot(epochsrange, loss, label= Training Loss )
plt.plot(epochsrange, valloss, label= Validation Loss )
plt.legend(loc= upper right )
plt.title( Training and Validation Loss )
plt.show()
model.loadweights( ./vgg16bestweights.hdf5 ) #initialize the best trained weights
ypred = model.predict(testds, batchsize = 32, verbose = 1)
ypred.shape
ypredclasses = ypred.argmax(axis = 1)
ypredclasses.shape
yclassesnames = [classnames[x] for x in ypredclasses]
predictions = pd.readcsv( /kaggle/input/paddy disease classification/samplesubmission.csv )
predictions[ label ] = yclassesnames
predictions.tocsv( vgg16submission.csv , index = False, header = True )
submission.head()

Inception V2 :
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sb
import cv2
from torch.utils.data import Dataset, DataLoader
import io
import os
from PIL import Image
from future import print_function
from future import division
import torch
import torch.nn as nn
import torch.optim as optim
import numpy as np
import torchvision
from torchvision import datasets, models, transforms
import matplotlib.pyplot as plt
import time
import os
import copy
print( PyTorch Version: , torch.version)
print( Torchvision Version: , torchvision.version)
traindir = ../input/paddy disease classification/train images
testdir = ../input/paddy disease classification/test images
```

```
traindir= ../input/paddy disease classification/trainimages
testdir= ../input/paddy disease classification/testimages
labelArr=traindata[ label ].unique()
label2id={}
id2label={}
index=0
print(label2id)
from sklearn.modelselectionimporttraintestsplit
train,valid=traintestsplit(traindata,testsize=0.05,randomstate=0)
print(len(train))
print(len(valid))
train[ label ].valuecounts()
testdataset=PaddyTestDataset(testdataframe,rootdir= ../input/paddy disease classification/
testimages ,transform=transformvalid)
testloader=torch.utils.data.DataLoader(testdataset,batchsize=16,numworkers=2)
dfpreds=pd.DataFrame({ imageid :filenames, label :finalpredictions})
dfpreds.head()
testdf=pd.readcsv( ../input/paddy disease classification/samplesubmission.csv )
submissiondf=pd.merge(testdf[[ imageid ]],dfpreds,how= inner ,on= imageid )
submissiondf.tocsv( submission.csv ,index=False)
submissiondf.head()
print(len(submissiondf))

ResNet :
import os
import numpy as np
import pandas as pd
importseabornas sns
importmatplotlib.pyplotas plt
from sklearnimportpreprocessing
from sklearn.modelselectionimport KFold
import cv2 as cv
importplotly.expressas px
from tqdm.notebook import tqdm
importtorch
importtorchvision.transformsastransforms
importtorch.optim as optim
importtorch.nn.functionalas F
importtorch.nn as nn
from torch.utils.dataimportDataset, DataLoader
importalbumentations as A
from albumentations.pytorch.transformsimport ToTensorV2
```

```
df = pd.readcsv( ../input/paddy disease classificati
    on/train.csv )
df.sample(7)
traindir= ../input/paddy disease classification/t
    rainimages/
df[ pathjpeg ] = df.apply(lambda row: traindir+row[ labe
    l ]+ /+row[ imageid ],axis=1)
df.sample(7)
plt.rcParams[ font.family ]= serif
fig,axes=plt.subplots(nrows=3,ncols=4,figsize=(18,15),su
    bplotkw={ xticks :[], yticks :
[]})
for ax in axes.flat:
i=np.random.randint(df.shape[0])
ax.imshow(plt.imread(df[ pathjpeg ]
from albumentations.pytorch.transformsimport ToTensorV2
df = pd.readcsv( ../input/paddy disease classificati
    on/train.csv )
df.sample(7)
traindir= ../input/paddy disease classification/t
    rainimages/
df[ pathjpeg ] = df.apply(lambda row: traindir+row[ labe
    l ]+ /+row[ imageid ],axis=1)
df.sample(7)
plt.rcParams[ font.family ]= serif
fig,axes=plt.subplots(nrows=3,ncols=4,figsize=(18,15),su
    bplotkw={ xticks :[], yticks :
[]})
for ax in axes.flat:
i=np.random.randint(df.shape[0])
ax.imshow(plt.imread(df[ pathjpeg ][i]))
ax.settitle(df[ label ][i],fontsize=16)
plt.show()
integermapping
i=np.random.randint(df.shape[0])
image=cv.imread(df[ pathjpeg ][i])
image=cv.cvtColor(image,cv.COLOR BGR2RGB)
transform=transformsmethod(image)
133 transformedimage=transform(image=image)[ image ]
# PyTorch Tensor
plt.imshow(transformedimage.permute(1,2,0));
model=model.to(device)
loss=torch.nn.CrossEntropyLoss()
optimizer=torch.optim.Adam(model.parameters(),lr=10**(-3)
    )
epochs=5
batchsize=37
submissiondir= ../input/paddy disease classificat
    ion/testimages/
submission=pd.DataFrame({
```

```
    imageid    : imageids,
    label      : labels,
})
submission.tocsv(    submission.csv    ,index=False,header=True)
submission.head()

MobileNet :
from PIL import Image
importmatplotlib.pyplot as plt
import pandas as pd
import numpy as np
importseaborn as sb
import os
importwarnings
warnings.filterwarnings(    ignore    )
from glob import glob
importtensorflow as tf
from tensorflow.keras.layers.experimental.preprocessingimport RandomFlip
importtensorflow.keras.layers as tfl
from tensorflow.keras.preprocessingimportimagedatasetfromdirectory
df = pd.readcsv(    ../input/paddy    disease    classification/train.csv    )
df.shape
df.head()
df.groupby(    variety    ).mean()
plt.figure(figsize=(10,10))
sb.countplot(data=df,x=    variety    )
defgetmodel(base,preprocessor,imgsize):
inputs=tf.keras.Input(shape=(imgsize,imgsize,3))
x = RandomFlip(    horizontal    )(inputs)
x=preprocessor(x)
x=base(x)
x=tfl.Flatten()(x)
x=tfl.Dense(1024,activation=    relu    )(x)
x=tfl.BatchNormalization()(x)
x=tfl.BatchNormalization()(x)
x=tfl.Dense(64,activation=    relu    )(x)
x=tfl.Dropout(0.3)(x)
x=tfl.BatchNormalization()(x)
outputs=tfl.Dense(10,activation=    softmax    )(x)
model=tf.keras.Model(inputs,outputs)
return model

testdataset=imagedatasetfromdirectory(
    ../input/paddy    disease    classification/testimages
        ,
imagesize=(imgsize,imgsize),
```

```
batch_size = 64,
shuffle = False,
labels =None,
class_names = None,
label_mode = None,
color_mode = rgb,
)
predictions=model_tuned.predict(test_dataset)
predictions.shape
ss = pd.read_csv('../input/paddy disease classification/sample_submission.csv')
ss['label'] = np.argmax(predictions, axis= 1)
ss['label'] = ss['label'].replace([0,1,2,3,4,5,6,7,8,9], labels)
ss.to_csv('Submission.csv', index=False)
ss.head()
```

9.2 Poster Presentation

Figure 9.1: **Poster Presentation**

References

[1] Houda Orchi et al, Mohamed Sadik, Mohammed Khaldoun ,and Essaid Sabir.Automation of Crop Disease Detection through Conventional Machine Learning and Deep Transfer Learning Approaches,vol.13(2), pp.352,2023.

[2] Ahmed Samit Hatem , Maha Sabri Altememe and Mohammed Abdulraheem Fadhel.Identifying corn leaves diseases by extensive use of transfer learning,Vol. 29(2), pp. 1030-1038, February 2023.

[3] Manzhou Li,Siyu Cheng, Jingyi Cui, Changxiang Li, Zeyu Li, Chang Zhou and Chunli Lv.High-Performance Plant Pest and Disease Detection Based on Model Ensemble with Inception Module and Cluster Algorithm,vol.12(1),pp.200,2023.

[4] Afsana Mimi, Sayeda Fatema Tuj Zohura, Muhammad Ibrahim, Riddho Ridwanul Haque, Omar Farrok, Taskeed Jabid and Md Sawkat Ali.Identifying Selected Diseases of Leaves using Deep Learning and Transfer Learning Models,vol.32(1),pp.55-71,2023.

[5] Vinay Gautam, Naresh K.Trivedi, Aman Singh, Heba G. Mohamed, Irene Delgado Noya, Preet Kaur and Nitin Goyal.A Transfer Learning-Based Artificial Intelligence Model for Leaf Disease Assessment,vol.14(20), no.13610,2022.

[6] Sunu Jatmika, Danang Eka Saputra. Rice Plants Disease Identification Using Deep Learning with Convolutional Neural Network Method,Vol. 7(2), July 2022.

[7] Andrew J, Jennifer Eunice, Daniela Elena Popescu, M. Kalpana Chowdary and Jude Hemanth. Deep Learning-Based Leaf Disease Detection in Crops Using Images for Agricultural Applications“.vol.12(10),pp.2395,2022.

[8] Gugan Kathiresan, Anirudh M, Nagharjun M and Karthik R.Identifying corn leaves diseases by extensive use of transfer learning.vol.1911,No.1,pp.012004

January 2021.

[9] SK Mahmudul Hassan, Arnab Kumar Maji, Michal Jasi nski, Zbigniew Leonowicz and Elzbieta Jasi nska. Identification of Plant-Leaf Diseases Using CNN and Transfer-Learning Approach.vol.10(12), pp.1388,2021.

[10] V.Malathi, M.P.Gopinath.Classification of pest detection in paddy crop based on transfer learning approach.vol.71(7),pp.552-559,2021.

[11] Lili Li, Shujuan And Bin Wang.Plant Disease Detection and Classification by Deep Learning.vol.9,pp.56683-56698,2021.

[12] Siddharth Swarup Rautaray, Manjusha Kumar Pandey, Mahendra Kumar Gourisaria, Ritesh Sharma and Sujay Das.Paddy Crop Disease Prediction A Transfer Learning Technique.Vol.8(6),pp.1490-1495,2020.

[13] Nour Eldeen M.Khalifa, Mohamed Loey, Mohamed Hamed N. Taha.Insects Pests Recognition Based On Deep Transfer Learning Models,vol.98(1),2020.

[14] Vijai Singh, Namita Sharma, Shikha Singh.A review of imaging techniques for plant disease detection".vol.4,pp.229-242.2021.

[15] Vimal K. Shrivastava, Monoj K. Pradhan, Sonajharia Minz, Mahesh P. Thakur.Rice Plant Disease Classification Using Transfer Learning Of Deep Convolution Neural Network.Vol.3(6),pp.631-635, 2019.